Mourad Elloumi Josef Küng
Michal Linial Robert F. Murphy
Kristan Schneider Cristian Toma (Eds.)

Communications in Computer and Information Science 13

Bioinformatics Research and Development

Second International Conference, BIRD 2008
Vienna, Austria, July 2008
Proceedings

Springer

Communications
in Computer and Information Science 13

Mourad Elloumi Josef Küng
Michal Linial Robert F. Murphy
Kristan Schneider Cristian Toma (Eds.)

Bioinformatics Research and Development

Second International Conference, BIRD 2008
Vienna, Austria, July 7-9, 2008
Proceedings

Springer

Volume Editors

Mourad Elloumi
UTIC, University of Tunis, Tunisia
E-mail: Mourad.Elloumi@fsegt.rnu.tn

Josef Küng
FAW, University of Linz, Austria
E-mail: jkueng@faw.at

Michal Linial
The Hebrew University of Jerusalem, Israel
E-mail: michall@cc.huji.ac.il

Robert F. Murphy
Carnegie Mellon University, PA, USA
E-mail: murphy@cmu.edu

Kristan Schneider
University of Vienna, Austria
E-mail: kristan.schneider@univie.ac.at

Cristian Toma
Politehnica University of Bucharest, Romania
E-mail: cgtoma@physics.pub.ro

Library of Congress Control Number: 2008930437

CR Subject Classification (1998): J.3

ISSN	1865-0929
ISBN-10	3-540-70598-8 Springer Berlin Heidelberg New York
ISBN-13	978-3-540-70598-7 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12324439 06/3180 5 4 3 2 1 0

Preface

This volume contains the papers which were selected for presentation at the second Bioinformatics Research and Development (BIRD) conference held in Vienna, Austria during July 7–9, 2008. BIRD covers a wide range of topics related to bioinformatics. This year sequence analysis and alignment, pathways, networks, systems biology, protein and RNA structure and function, gene expression/regulation and microarrays, databases and data integration, machine learning and data analysis were the subjects of main interest.

The decisions of the Program Committee are based on the recommendations of at least three, up to five, reviews for each paper. As a result, 30 of the 61 submitted contributions could be accepted for the conference.

We were happy to have three invited talks presented by experienced researchers providing visitors with a good overview but also some very important insights into the fascinating domain of bioinformatics. Abstracts and more information on these talks are provided in the conference program as well as at the conference site.

In the second part of this volume the selected contributions of the two workshops which were held in parallel to the main conference are presented: Workshop on Dynamical Aspects of Perturbation, Intervention and Transition in Biological Systems – PETRIN 2008 and Workshop on Algorithms in Molecular Biology – ALBIO 2008

Poster presentations of the BIRD conference are in the companion proceedings published by the Trauner Verlag, Linz.

The second BIRD conference was a successful continuation of this new conference series, which started last year in Berlin. First of all the editors want to thank the authors, whose work made this volume possible. Then we want to thank the invited speakers, Rudolf Freund, Peter Schuster, Tom Slezak.

We especially thank the Program Committee members, who ensured the high quality of the conference and the publications in this volume, provided their experience and expert knowledge, and invested time to select the best submissions. We thank the Web administrators (A. Anjomshoaa, A. Dreiling, C. Teuschel), who took care of the online submission process and registration.

Most of all we wish to express our gratitude to Gabriela Wagner (DEXA Society), who managed the conference, which includes organizing the submission and the review process, setting up and coordinating the decisions of the Program Committee, being responsible for the final paper versions, planning and supervising the technical program schedule including the banquet, taking care of the editorial and printing issues, and much more.

July 2008

Robert Murphy
Roland Wagner
Josef Küng
Michal Linial
Kristan Schneider

Organization

Program Committee

Honorary Co-chairs

Heinz Engl, Austrian Academy of Sciences and University of Vienna, Austria
Rudolf Freund, Vienna University of Technology, Austria
Amarnath Gupta, SDSC of the University of California San Diego, USA
Vladimir Marik, Czech Technical University, Czech Republic
Chris Sander, MSKCC - Computational Biology Center, USA
A Min Tjoa, Vienna University of Technology, Austria

General Co-chairs

Robert Murphy, Carnegie Mellon University, USA
Roland Wagner, University of Linz, Austria

Organizing Chair

Gabriela Wagner, DEXA Society, Austria

Poster Session Co-chairs

Djork-Arné Clevert, Signature Diagnostics, Germany
Roland Wagner, University of Linz, Austria

Workshop Co-chairs

Roland R. Wagner, University of Linz, Austria
Gerald Quirchmayr, University of South Australia, Australia

Program Committee Co-chairs

Josef Küng, University of Linz, Austria
Michal Linial, The Hebrew University of Jerusalem, Israel
Kristan Schneider, University of Vienna, Austria

Program Committee

Werner Aigner, FAW, Austria
Fuat Akal, University of Basel, Switzerland
Tatsuya Akutsu, Kyoto University, Japan
Vasco Amaral, Universidade Nova de Lisboa, Portugal
Walid G. Aref, Purdue University, USA

Rubén Armañanzas Arnedillo, University of the Basque Country, Spain
Santosh Atanur, C-DAC, India
Brian Aufderheide, Keck Graduate Institute of Applied Life Sciences, USA
Rolf Backofen, Universität Freiburg, Germany
Luis Bagatolli, University of Southern Denmark, Denmark
Peter Baumann, Jacobs University Bremen, Germany
Khalid Benabdeslem, University of Lyon1 - LIESP, France
Christian Blaschke, Bioalma Madrid, Spain
Jacek Blazewicz, Poznan University of Technology, Poland
Brigitte Boeckmann, Swiss Institute of Bioinformatics, Switzerland
Andreas M. Boehm, Rudolf Virchow Center for Experimental Biomedicine, Germany
Veselka Boeva, Technical University of Plovdiv, Bulgaria
Erik Bongcam-Rudloff, Uppsala University, Sweden
Anthony Bonner, University of Toronto, Canada
Roberta Bosotti, Nerviano Medical Science s.r.l., Italy
Timo Breit, University of Amsterdam, The Netherlands
Bruno Buchberger, University of Linz, Austria
Philipp Bucher, Swiss Institute of Bioinformatics, Switzerland
Rita Casadio, University of Bologna, Italy
Sònia Casillas, Universitat Autonoma de Barcelona, Spain
Silvana Castano, Universita' degli Studi di Milano, Italy
Bulbul Chakravarti, Keck Graduate Institute of Applied Life Sciences, USA
Belinda Chang, University of Toronto, Canada
Kun-Mao Chao, National Taiwan University, Taiwan
Rosana Chehín, CONICET-Argentine, Argentina
Phoebe Chen, Deakin University, Australia
Cindy Chen, University of Massachusetts Lowell, USA
Francis Y.L. Chin, The University of Hong Kong, Hong Kong
Bin Cui, Peking University, China
Coral del Val Muñoz, University of Granada, Spain
Sabine Dietmann, GSF - National Research Center for Environment and Health, Germany
Zhihong Ding, University of California, Davis, USA
Pierre Dönnes, F. Hoffmann-La Roche Ltd., Switzerland
Arjan Durresi, Louisiana State University, USA
Silke Eckstein, Technical University of Braunschweig, Germany
Hans-Dieter Ehrich, Technical University of Braunschweig, Germany
Ingvar Eidhammer, University of Bergen, Norway
Domenec Farre, Center for Genomic Regulation, Spain
Pedro Fernandes, Inst. Gulbenkian de Ciência, Portugal
Jorge H. Fernandez, EMBRAPA, Brazil
Christoph M. Flamm, University of Vienna, Austria
Javier Forment Millet, Universidad Politécnica de Valencia, Spain
Christoph M. Friedrich, Fraunhofer SCAI, Germany
Cornelius Frömmel, Georg-August-Universität Göttingen, Germany
Michal J. Gajda, International Institute of Molecular and Cell Biology, Poland
Rachelle Gaudet, Harvard University, USA

See Kiong Ng, Institute for Infocomm Research, Singapore
Vit Novacek, National University of Ireland, Galway, Ireland
Boris Novikov, University of St. Petersburg, Russia
Klaus Obermayer, Technical University of Berlin, Germany
Bjorn Olsson, University of Skovde, Sweden
Allan Orozco, School of Medicine, UCR, Costa Rica
Jean Peccoud, Virginia Polytechnic Institute and State University, USA
Jose M. Peña, Linköping University , Sweden
Zhiyong Peng, Wuhan University, China
Francisco Pinto, ITQB, University of Lisbon, Portugal
Uwe Plikat, Novartis Pharma AG, Switzerland
Thomas Ploetz, University of Dortmund, Germany
Adam Podhorski, CEIT, Spain
Meikel Poess, Oracle Corporation, USA
C.V.S. Siva Prasad, Indian Institute of Information Technology, India
Steve Qin, University of Michigan, USA
Shoba Ranganathan, Macquarie University, Australia
Axel Rasche, Max Planck Institute for Molecular Genetics, Germany
Dietrich Rebholz, European Bioinformatics Institute, UK
Peter Robinson, Humboldt-Universität, Germany
David Rocke, University of California, USA
Paolo Romano, National Cancer Research Institute (IST), Italy
Angel Rubio, CEIT, Spain
Cristina Rubio-Escudero, University of Granada, Spain
Victor Sabbia, Laboratorio de Organización y Evolución del Genoma, Uruguay
Hershel Safer, Weizmann Institute of Science, Israel
Nick Sahinidis, Carnegie Mellon University, USA
Yasubumi Sakakibara, Keio University, Japan
Meena K. Sakharkar, National University of Singapore, Singapore
Francisca Sánchez Jiménez, University of Málaga, Spain
Guido Sanguinetti, University of Sheffield, UK
Clare Sansom, Birkbeck College, London, UK
Roberto Santana, University of the Basque Country, Spain
Kengo Sato, Computational Biology Research Center, AIST, Japan
Kenji Satou, JAIST, Japan
Alexander Schliep, Max Planck Institute for Molecular Genetics, Germany
Wolfgang Schreiner, Medical University of Vienna, Austria
Torsten Schwede, University of Basel, Switzerland
Angel Sevilla Camins, Universität Stuttgart, Germany
Denis Shestakov, University of Turku, Finland
Florian Sieker, Jacobs University Bremen, Germany
Tom Slezak, Lawrence Livermore National Lab L-174, USA
Sagi Snir, University of California, Berkeley, USA
Dimitri Soshnikov, Moscow Aviation Technical University, Microsoft Russia, Russia
Peter F. Stadler, University of Leipzig, Germany
Stefan Stanczyk, Oxford Brookes University, UK
Boris Steipe, University of Toronto, Canada

Workshop on Algorithms in Molecular Biology – ALBIO 2008

Computational molecular biology has emerged from the Human Genome Project as an important discipline for academic research and industrial application. The exponential growth of the size of biological databases, the complexity of biological problems and the necessity to deal with errors in biological sequences result in time efficiency and memory requirements. The development of fast, low-memory requirements and high-performances algorithms is thus increasingly important in computational molecular biology.

Papers presented in this workshop deal with algorithms that solve fundamental and/or applied problems in molecular biology, that are computationally efficient, that have been implemented and experimented on simulated and/or on real biological sequences, and that provide interesting new results.

Mourad Elloumi

Program Committee

Workshop on Dynamical Aspects of Perturbation, Intervention and Transition in Biological Systems – PETRIN 2008

Important aspects in information theory and control theory appeared by studying the behavior of biological systems. The classical control loop is a good example presenting the methods used by biological entities for controlling certain functional parameters in different circumstances. The further development of control theory and of dynamical models led to important achievements in the study of evolutionary processes. However, some modern aspects in physics (quantum theory) and mathematics (wavelets, fractal theory) imply a more profound approach of transitions and short-range phenomena both in materials science and in natural (biological) systems.

The mathematical formalism of impulsive systems tries to use the rigorous aspects from continuous systems formalism as well as the wide range of applications of discrete systems formalism. They were introduced due to the fact that many evolution processes are characterized by the fact that at certain moments of time they are subject to short-term perturbations (having the form of external impulses). It is known, for example, that many biological phenomena involving thresholds, bursting rhythm models in medicine and biology, optimal control models in economics, and frequency modulated systems present impulsive effects. Thus impulsive differential equations (involving impulse effects) can describe the evolution of many scientific and technical phenomena.

Yet the study of such abrupt changes must be completed with logical, mathematical and technical aspects connected with the moment of action of such impulsive external commands.

To model such changes (transitions) in an accurate manner, specific dynamics on limited time interval is required. It must be taken into account that bioinformatics should join together both statistical aspects (well known in natural sciences from thermodynamics theory and quantum theory) and deterministic aspects (describing the evolution of systems by differential equations, similar to classical mechanics). As particular aspects for biological systems, aspects connected with the so-called free-choice (for human systems) and external intervention (for human and biological systems) should be added . This implies the use of accurate dynamics of perturbations, intervention and transition in multi-scale systems, for deterministic aspects and stochastic aspects to be merged into a unitary model of the Proper Time of Intervention (a very useful concept for human action).

The Workshop on Dynamical Aspects of Perturbation, Intervention and Transition in Biological Systems is intended to emphasize the necessity of joining together deterministic and stochastic methods in an accurate multi-scale approach for modeling perturbations, transitions and interventions in biological systems.

The PETRIN 2008 Workshop represented a major scientific event organized by the Group for Interdisciplinary Science, Romanian Commission for UNESCO.

Cristian Toma

Program Committee

Sterian Paul - Politehnica University, Bucharest, Romania
Sterian Andreea- Hyperion University, Bucharest, Romania
Song Wanqing - University of Shanghai, China
Sanchez Ruiz Luis - University of Valencia, Spain
Majak Juri - University of Talinn, Estonia
Iordache Dan - Politehnica University, Bucharest, Romania

Table of Contents

BIRD Conference

Gene Expression/Regulation and Micro Arrays

A Tree Index to Support Clustering Based Exploratory Data Analysis 1
Christian Martin and Tim W. Nattkemper

XMAS: An Experiential Approach for Visualization, Analysis, and Exploration of Time Series Microarray Data 16
Ben Dalziel, Hui Yang, Rahul Singh, Matthew Gormley, and Susan Fisher

Clustering in a Fixed Manifold to Detect Groups of Genes with Similar Expression Patterns 32
N.M. Phuong and H.D. Tuan

Sequence Analysis and Alignment I

Families of FPGA-Based Accelerators for BLAST Algorithm with Multi-seeds Detection and Parallel Extension 43
Fei Xia, Yong Dou, and Jiaqing Xu

Fast Structured Motif Search in DNA Sequences 58
Mihail Halachev and Nematollaah Shiri

A Discriminative Method for Protein Remote Homology Detection Based on N-nary Profiles 74
Bin Liu, Lei Lin, Xiaolong Wang, Qiwen Dong, and Xuan Wang

Sequence Analysis and Alignment II

Searching for Supermaximal Repeats in Large DNA Sequences 87
Chen Na Lian, Mihail Halachev, and Nematollaah Shiri

Motif Location Prediction by Divide and Conquer 102
Mohammed Alshalalfa and Reda Alhajj

Protein and RNA Structure and Function I

Translational Control by RNA-RNA Interaction: Improved Computation of RNA-RNA Binding Thermodynamics 114
Ulrike Mückstein, Hakim Tafer, Stephan H. Bernhart, Maribel Hernandez-Rosales, Jörg Vogel, Peter F. Stadler, and Ivo L. Hofacker

A Symmetry-Free Subspace for *Ab initio* Protein Folding Simulations ... 128
Xiangchao Gan, Leonidas Kapsokalivas, Andreas A. Albrecht, and Kathleen Steinhöfel

Protein and RNA Structure and Function II

Comparative Analysis of Disulfide Bond Determination Using Computational-Predictive Methods and Mass Spectrometry-Based Algorithmic Approach ... 140
Timothy Lee and Rahul Singh

Exploration of Evolutionary Relations between Protein Structures ... 154
Natalja Kurbatova and Juris Viksna

Two Local Search Methods for Protein Folding Simulation in the HP and the MJ Lattice Models ... 167
L. Kapsokalivas, X. Gan, A. Albrecht, and K. Steinhöfel

A Robust Class of Stable Proteins in the 2D HPC Model ... 180
Alireza Hadj Khodabakhshi, Ján Maňuch, Arash Rafiey, and Arvind Gupta

Machine Learning and Data Analysis

A Novel Adaptive Multiple Imputation Algorithm ... 193
Veselka Boeva and Elena Tsiporkova

Topological Metrics in Blast Data Mining: Plasmid and Nitrogen-Fixing Proteins Case Studies ... 207
Pietro Lió, Matteo Brilli, and Renato Fani

Gene Expression Mining for Cohesive Pattern Discovery ... 221
Ramkishore Bhattacharyya and Balaram Bhattacharyya

Databases and Data Integration I

Identifying Subcellular Locations from Images of Unknown Resolution ... 235
Luís Pedro Coelho and Robert F. Murphy

E-BioFlow: Different Perspectives on Scientific Workflows ... 243
Ingo Wassink, Han Rauwerda, Paul van der Vet, Timo Breit, and Anton Nijholt

Knowledge Acquisition Focused Cooperative Development of Bio-ontologies – A Case Study with BIO2Me ... 258
Dominic Mainz, Ingo Paulsen, Indra Mainz, Katrin Weller, Jochen Kohl, and Arndt von Haeseler

Pathways, Networks, Systems Biology I

Nested q-Partial Graphs for Genetic Network Inference from "Small n, Large p" Microarray Data 273
Kevin Kontos and Gianluca Bontempi

A Computational Method for Reconstructing Gapless Metabolic Networks 288
Esa Pitkänen, Ari Rantanen, Juho Rousu, and Esko Ukkonen

Finding Frequent Subgraphs in Biological Networks Via Maximal Item Sets 303
Hans Zantema, Stefan Wagemans, and Dragan Bošnački

Multi-functional Protein Clustering in PPI Networks 318
Clara Pizzuti and Simona E. Rombo

Pathways, Networks, Systems Biology II

Protein-Protein Interaction Network Querying by a "Focus and Zoom" Approach 331
Valeria Fionda, Luigi Palopoli, Simona Panni, and Simona E. Rombo

Matching Spatial Regions with Combinations of Interacting Gene Expression Patterns 347
Jano van Hemert and Richard Baldock

Combining Molecular and Physiological Data of Complex Disorders 362
Emanuel Schwarz, F. Markus Leweke, Sabine Bahn, and Pietro Lió

Combining Protein-Protein Interaction (PPI) Network and Sequence Attributes for Predicting Hypertension Related Proteins 377
Richard J.B. Dobson, Patricia B. Munroe, Charles A. Mein, Mark J. Caulfield, and Mansoor A.S. Saqi

Databases and Data Integration II

Genome Structure and Characterisation of an Endogenous Retrovirus from the Zebrafish Genome Project Database 392
Roziah Kambol and M. Faris Abtholuddin

Bayesian Phylogeny on Grid 404
Richard C. van der Wath, Elizabeth van der Wath, Antonio Carapelli, Francesco Nardi, Francesco Frati, Luciano Milanesi, and Pietro Lió

Workshop ALBIO

Comparison of Exact String Matching Algorithms for Biological Sequences 417
Petri Kalsi, Hannu Peltola, and Jorma Tarhio

A New Approximation Algorithm for the Minimum Fragment Removal Problem 427
Hatem Tahri and Mourad Elloumi

Indexing Factors in DNA/RNA Sequences 436
Tomáš Flouri, Costas Iliopoulos, M. Sohel Rahman, Ladislav Vagner, and Michal Voráček

Implementation of a Swap Matching Algorithm Using a Graph Theoretic Model 446
Pavlos Antoniou, Costas S. Iliopoulos, Inuka Jayasekera, and M. Sohel Rahman

Suffix Tree Characterization of Maximal Motifs in Biological Sequences 456
Maria Federico and Nadia Pisanti

Efficient Seeding Techniques for Protein Similarity Search 466
Mikhail Roytberg, Anna Gambin, Laurent Noé, Sławomir Lasota, Eugenia Furletova, Ewa Szczurek, and Gregory Kucherov

An Algorithm for Multiple and Global Alignments 479
Mourad Elloumi and Ahmed Mokaddem

LASA: A Tool for Non-heuristic Alignment of Multiple Sequences 489
Ernst Althaus and Stefan Canzar

SVM-Based Local Search for Gene Selection and Classification of Microarray Data 499
Jose Crispin Hernandez Hernandez, Béatrice Duval, and Jin-Kao Hao

GENFOCS – A Comparative Tool on Gene Finding with Sensitivity and Specificity 509
M.A. Lijo Anto, G. Gopakumar, Achuthsankar S. Nair, and Indira Ghosh

Gene Machine© – A Hardware/Software Platform for Analyzing Genome Data 517
Roger Marshall

Workshop PETRIN

Complex Representation of DNA Sequences 528
Carlo Cattani

Wavelet Analysis of Impulses in Axon Physiology 538
Carlo Cattani and Massimo Scalia

Acquisition and Algorithms for Fluorescence Applications 546
Elena Mocanu and Paul Sterian

Control of the Lasers Dynamics by Pumping Modulation for Biophotonics Applications 556
Paul Sterian, Octavian Marin, and Valerică Ninulescu

Quantum Well Lasers for Medical Industry 563
Valerică Ninulescu, Vlăduţ-Bogdan Nicolae, and Andreea Sterian

Computational Model for the Study of the Fractal Parameters Characteristic for the Surface of Titanium Dental Implants 571
Stefan Pusca and Theodora Toma

VEMS-TM – A Management Infrastructure 581
Trandafir Moisa and Cristian Morarescu

PTF Model for Evaluating the Relationships between the VAC and Antimicrobial Resistance in Human Communities 587
Irina Codita and Stefan Pusca

Author Index 595

A Tree Index to Support Clustering Based Exploratory Data Analysis

Christian Martin and Tim W. Nattkemper

Technical Faculty
AG Applied Neuroinformatics
Bielefeld University
{christian.martin, tim.nattkemper}@uni-bielefeld.de

Abstract. In microarray data analysis, visualizations based on agglomerative clustering results are widely applied to help biomedical researchers in generating a mental model of their data. In order to support a selection of the to-be-applied algorithm and parameterizations, we propose a novel cluster index, the *tree index (TI)*, to evaluate hierarchical cluster results regarding their visual appearance and their accordance to available background information. Visually appealing cluster trees are characterized by splits that separate those homogeneous clusters from the rest of the data, which have low inner cluster variance and share a medical class label. To evaluate clustering trees regarding this property, the TI computes the likeliness of every single split in the cluster tree. Computing TIs for different algorithms and parameterizations allows to identify the most appealing cluster tree among many possible tree visualizations obtained. Application is shown on simulated data as well as on two public available cancer data sets.

1 Introduction

In modern biomedical research, the number of experiments and studies using microarray technology keeps continuously increasing [1,2]. Microarray data is usually characterized by a high dimensionality (many genes), few data points (few samples or experimental conditions), a low signal-to-noise ratio, outliers, and missing values making many standard statistical test methods applicable only to a limited extend.

In the following, we consider the general task of exploratory data analysis of a preprocessed microarray data set $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_d\}$ of d biological samples. When exploring this microarray data, the analysis very often includes unsupervised cluster algorithms. Unlabeled data is divided into natural groups, which may correspond to particular macroscopic phenotypes or functional categories. The cluster algorithms can be classified as hierarchical, partitioning and density-based methods [3,4,5].

Agglomerative clustering [6,7] is the basis for most visual data mining tasks in microarray applications, since in the cluster tree (alias dendrogram) the intrinsic hierarchical cluster structure of a data set is visually accessible at once. Most recently, *normalized cuts* [8], a *spectral clustering* approach has also been applied to microarray data [9,10]. One problem in clustering based exploratory data analysis is the variability of the cluster result dependent on the applied cluster algorithm and parameterizations (preprocessing of the data, (dis-)similarity measure). There is hardly any consensus about how to choose these [11,12]. This results in an enormous number of potential visual displays for one data set leading to the confusion of the biomedical researcher. It is common practice to test different algorithms and parameterizations and to select the cluster result which seems to be the most appropriate according to one's knowledge and anticipations. Thus, an analytical and objective evaluation of cluster results would help to identify the algorithm and parameterization that yield objectively reasonable cluster results. Cluster indices assess the quality of a clustering by evaluating the data inside the clusters and by quantifying the amount of global structure captured by the clustering. Cluster indices can be grouped into *internal* and *external* ones [11,13]. Internal indices evaluate the quality of a cluster by using only intrinsic information of the data. They use the same data which has been used by the cluster algorithm itself. The following internal measures have been developed: *Goodman-Kruskal Index* [14], *Calinski Harabasz Index* [15], *Dunn's index* [16], *C-Index* [17], *Davis-Bouldin index* [18], *Silhouette index* [19], *Homogeneity* and *Separation* [20], and *Index I* [21]. Most of these measures have already been successfully applied to microarray data [22], and are integrated in software packages for analysis of gene expression data [23,24].

More recently, external indices have gained a remarkable popularity to evaluate results of various cluster algorithms. External evaluation is based on the assumption that a *real* class label or category *(gold standard)* is known for each element. The cluster result which best reflects both the internal structure and the preset categories, obtains the highest score. The label can be a particular macroscopic phenotype, a functional category or any other category of interest. An important statistical measure is the *Rand Index* [25] or the *adjusted Rand index* [26], measuring the similarity between two partitions that are the clustering and the external label. A further improvement is the *weighted Rand index*, proposed and applied on microarray data [27]. Furthermore the following indices are proposed in the bioinformatics literature: The *cumulative hypergeometric distribution* is used to compute a p-value measuring the probability of observing at least a certain number of genes of the same annotation in a cluster [28,29]. The *biological homogeneity index (BHI)* is proposed, measuring the fraction of genes with the same annotation in one cluster, and the *biological stability index (BSI)* measuring the stability of cluster results in a leave-one-out approach [30]. Clusterings of genes are compared using the concept of mutual information [31]. ANOVA is applied to measure the accordance of the clustering to a linear combination of a binary vector specifying the membership to functional categories [12]. Finally, a *figure of merit (FOM)* is proposed to evaluate a clustering obtained by a leave-one-out approach [32]. The left out sample is used as external label for validation.

A drawback of all indices proposed so far is that they all work on results obtained by partitioning methods. The data must be clustered in k groups, whereas k must either be estimated beforehand or during the cluster evaluation process. Hierarchical cluster trees are usually evaluated by cutting the tree at some level yielding k clusters. Even though an evaluation of a hierarchical cluster tree applying traditional indices (for partitions) at any level of the tree is imaginable, the development of a stable and unbiased index for trees is not straight-forward. In this paper we propose a novel external cluster index for cluster trees, the *tree index*. It is optimized to identify the algorithm and parameterization (preprocessing of the data, (dis-)similarity measure), yielding the clustering that

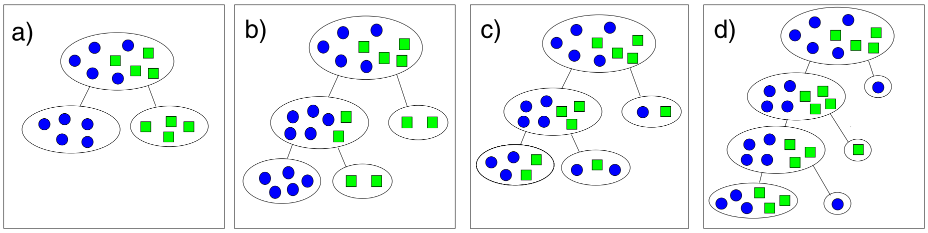


Fig. 1. The first splits of four different cluster trees are shown. In an optimal cluster tree the data is divided into homogeneous clusters at the very first split **(a)**. Usually such an optimal cluster tree cannot be generated for real data. An appealing cluster tree is rather characterized by many (here: two) splits inside the cluster tree each dividing a heterogeneous cluster into almost homogeneous subclusters **(b)**. The purer and larger the subclusters in a split, the cluster tree is well **(b)** or not well **(c)** suited for a visual datamining task. A degenerated cluster tree **(d)** separating only single elements from the rest of the data in each split if of a lower quality.

is best suited for visualization. In biomedical applications, microarray data is usually analyzed in combination with additional variables, like clinical data or tumor classifications. Thus we measure the usefulness of a tree visualization according to an external class label. For demonstration, the index is applied to cluster trees created by agglomerative clustering and normalized cuts on simulated data as well as on two public available cancer data sets.

2 Methods

We consider a preprocessed microarray data set with d samples (for instance derived from d tissue samples) of g genes, $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_i, \ldots, \mathbf{x}_d\}, \dim \mathbf{x}_i = g$. Based on some background information, one out of κ possible external labels or categories $c_i \in \{\mathcal{C}_1, \ldots, \mathcal{C}_\kappa\}$ is assigned to each sample $\mathbf{x}_i$ (for instance $\mathcal{C}_j$ = tumor classification of the tissue). In contrast to classification, we use the data labels to tune our visualization and not to predict a class for a new sample. Let us now assume that $\mathcal{X}$ has been clustered by some hierarchical agglomerative or divisive cluster algorithm yielding a cluster tree (Fig. 1). To characterize the features of a cluster tree that allow efficient visual data mining, we consider the tree as a result of a statistical process. In the ideal case, the data is divided into homogeneous clusters at the first split (Fig. 1a). Usually such an optimal cluster tree cannot be generated for real data. In a more realistic scenario an appealing cluster tree is characterized by many splits that divide a heterogeneous clusters into nearly homogeneous subclusters (Fig. 1b). The purer and larger the subclusters in a split, the more interesting they are, since each of them is defined by a clear pattern of variables that separate it from the rest of the data. Cluster trees with heterogeneous subclusters (Fig. 1c) or degenerated cluster trees (Fig. 1d) separating only single elements from the rest of the data in each split are of lower visual quality. When considering the splits of a cluster tree from a statistical point of view, the probabilities of the splits permit to distinguish between cluster trees of different qualities. Obviously,

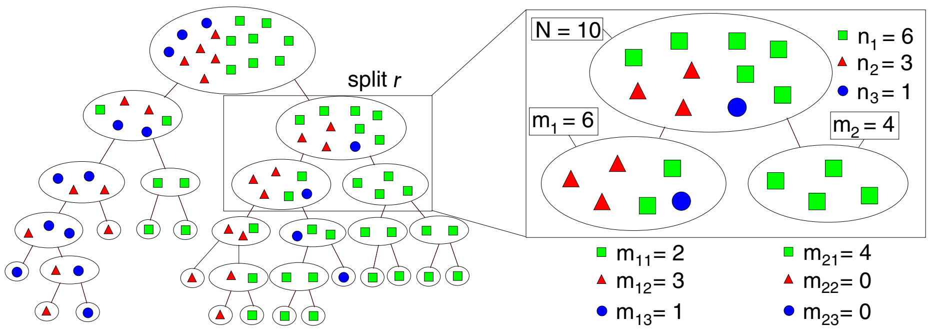


Fig. 2. In the r-th split, a cluster with $N = 10$ elements belonging to $\kappa = 3$ categories is split into $l = 2$ subclusters, each containing m_i elements with $m_{i\lambda}$ elements belonging to category $\mathcal{C}_\lambda$. In this split a completely homogeneous cluster is separated from the rest of the data. From a statistical point of view this event is rather unlikely resulting in a high splitting score for the r-th split.

a cluster tree of high quality is characterized by many unlikely splits, separating large homogeneous clusters from the rest of the data.

We now introduce the *tree index*, which is based on the evaluation of probabilities of every single split in a cluster tree. Clusters, also homogeneous ones, are always split until only singleton clusters are left since the label is not considered during the clustering process (Fig. 2). In a first step a *splitting score* is computed for every single split in the cluster tree based on the probability of the split. In a second step, all splitting scores are combined to compute the final tree index.

Step 1: Looking at the r-th split (the splits are numbered arbitrary), a cluster with N elements is split into l (usually $l = 2$) smaller subclusters (Fig. 2). The elements of the main cluster belong to κ different categories whereas $n_\lambda, \lambda \in \{1, \ldots, \kappa\}$ specifies the number of elements belonging to category $\mathcal{C}_\lambda$. The i-th subcluster contains m_i elements with $m_{i\lambda}$ elements belonging to category $\mathcal{C}_\lambda$. The primary objective is to compute the probability of such a particular split by taking the observed distributions in the clusters into account. It is assumed that $m_i, i \in \{1, \ldots, l\}$ elements are drawn from the N elements by sampling without replacement. Thereby each element is drawn with the same probability. For two categories ($\kappa = 2$) and two subclusters ($l = 2$) the probability of the observed distribution is given by the *hypergeometric distribution*.

$$p(m_{11}, m_{12}; N, n_1, m_1, m_2) = \frac{\binom{m_1}{m_{11}}\binom{m_2}{m_{12}}}{\binom{N}{n_1}} \tag{1}$$

For the general case (κ categories and l subclusters) the probability is given by a generalized form of the *polyhypergeometric distribution* or *multivariate hypergeometric distribution* [33]. Let $\mathbf{M} = \{m_{i\lambda}\},\ \mathbf{n} = \{n_\lambda\}$, and $\mathbf{m} = \{m_i\}$ with $1 \le i \le l$ and $1 \le \lambda \le \kappa$.

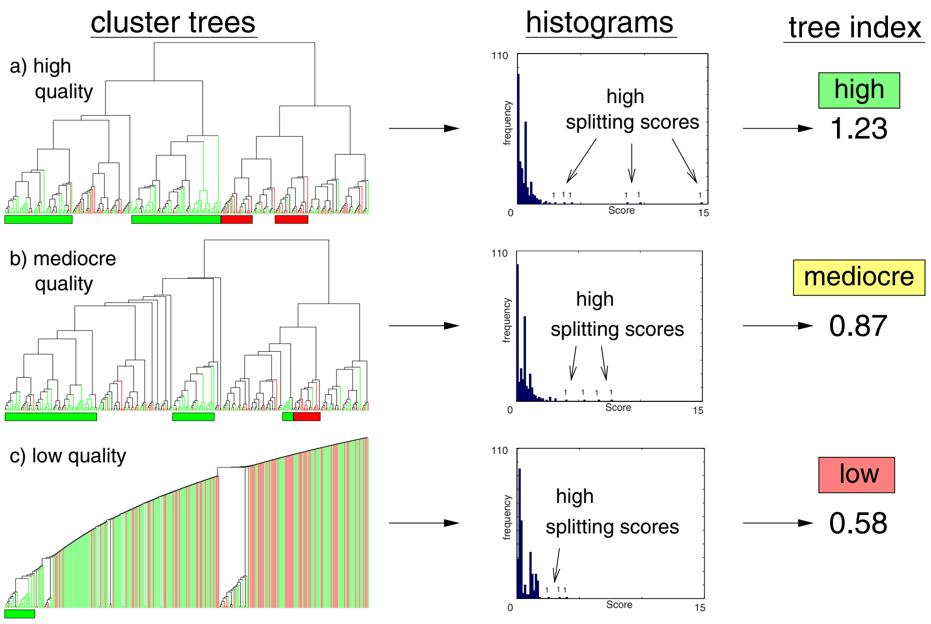


Fig. 3. Cluster trees and histograms with a high (obtained by spectral clustering, $\sigma = 10^{-2}$, Euclidean (dis-)similarity, all normalization), mediocre (obtained by complete linkage, $\sigma = 1$, eucl., all norm.), and low (obtained by single linkage, $\sigma = 1$, eucl., all norm.) tree index (TI) are shown. In all histograms, many splitting scores are close to zero. These result from less important splits dividing small clusters. The quantity and amplitude of a few high splitting scores characterize the quality of a cluster tree. A cluster tree with a high TI is characterized by a histogram with some splitting scores of high amplitude **(a)**. These splitting scores correspond to splits inside the cluster tree that divide clusters in large and nearly pure subclusters. A cluster tree with a mediocre TI is characterized by a histogram with some splitting scores of a middle amplitude **(b)**. These splitting scores correspond to less important splits inside the cluster tree that divide clusters in less larger and less purer subclusters than observed in the cluster tree with the high TI. A cluster tree with a low TI is characterized by only a very few splitting scores of low amplitude **(c)**. Such a degenerated cluster tree consists of many splits separating only one single element from the rest of the data.

$$p(\mathbf{M}; N, \mathbf{n}, \mathbf{m}) = \frac{\prod_{i=1}^{l} \frac{m_i!}{\prod_{\lambda=1}^{\kappa} m_{i\lambda}!}}{\frac{N!}{\prod_{\lambda=1}^{\kappa} n_\lambda!}} \tag{2}$$

$p(\mathbf{M}; N, \mathbf{n}, \mathbf{m})$ decreases with the size of the cluster that is split and with the homogeneity of the subclusters. The probability reaches its maximum if the distribution in a given cluster correlates to the distribution in the subcluster, indicating a random split. We define the splitting score S_r of the r-th split by its negative logarithmic probability.

$$\begin{aligned} S_r(\mathbf{M}; N, \mathbf{n}, \mathbf{m}) &= -\ln p(\mathbf{M}; N, \mathbf{n}, \mathbf{m}) \\ &= \ln N! - \sum_{\lambda=1}^{\kappa} \ln n_\lambda! - \sum_{i=1}^{l} \left(\ln m_i! - \sum_{\lambda=1}^{\kappa} \ln m_{i\lambda}! \right) \end{aligned} \tag{3}$$

A splitting score of a given cluster reaches its minimum if the distribution in the cluster correlates to the distribution in the subclusters. The splitting score increases with the size of the cluster that is split and with the homogeneity of the subclusters. Thus splits at higher levels in a cluster tree dividing larger clusters are generally capable to produce higher splitting scores. Splits at lower levels in a cluster tree divide clusters containing only few elements. This results in many splitting scores close to zero, since most of the splits are located in the lower part of a cluster tree. A split dividing a homogeneous cluster always has a splitting score of zero. Therefore the splits inside homogeneous clusters are of no importance for the further computation of the tree index.

Step 2: The set of all splitting scores enables to distinguish between cluster trees of different qualities. Independent of the internal structure of the cluster tree, the sum of all splitting scores is constant. Many splitting scores are zero (splits of homogeneous clusters) or close to zero (splits of small clusters). For illustration cluster trees of a high, mediocre and low quality are presented in Fig. 3.

A cluster tree of low quality is characterized by mostly low splitting scores and a very few high splitting scores (Fig. 3c). A cluster tree of high quality has considerably more high splitting scores (Fig. 3a).

To combine the complete set of splitting scores to a parameter-free index, we propose to use the standard deviation of splitting scores to capture the quality of a cluster tree, by defining the tree index (TI) by:

$$\mathrm{TI} = \sqrt{\frac{1}{R}\sum_{r=1}^{R}\left(S_r - \bar{S}\right)^2}, \text{ with } \bar{S} = \frac{1}{R}\sum_{r=1}^{R} S_r, \tag{4}$$

and R the number of splits in the cluster tree. Usually $\bar{S}$ is close to zero because many S_r are close to zero. Thus, the quantity and amplitude of high S_r basically determines the index. The higher the index, the more appealing is the corresponding cluster tree display.

3 Results

For illustration, the tree index is applied to cluster trees obtained from simulated data and two public available cancer data sets.

3.1 Simulated Data

Our artificial data set $\mathcal{C}$ consists of five classes, each containing $b = 8$ items that are scattered around their class centers with normally distributed noise ($\sigma^* = 0.1$):

$$\mathcal{C} = \bigcup_{i=1}^{5} \mathcal{C}_i, \text{ with } \mathcal{C}_i = \{(\mathbf{x}_j, c_j),\ \mathbf{x}_j \in \mathcal{N}(\mu_i, \sigma^*),\ c_j = i,\ j \in [1, b]\}, \tag{5}$$

whereas $(\mathbf{x}_j, c_j)$ comprises a two-dimensional data point $\mathbf{x}_j$ and the corresponding label c_j. The class centers are given by $\mu_1 = (2,2)^T$, $\mu_2 = (5,2)^T$, $\mu_3 = (3,10)^T$, $\mu_4 = (50,2)^T$, and $\mu_5 = (50,4)^T$, meaning that $\mathcal{C}_1$ and $\mathcal{C}_2$ as well as $\mathcal{C}_4$ and $\mathcal{C}_5$ are grouped close together, with a large gap between the two groups, whereas $\mathcal{C}_3$ is located in the further vicinity of $\mathcal{C}_1$ and $\mathcal{C}_2$. Now, additional normally distributed noise $\sigma \in [0.1, 100]$ is added to each point in the data set $\mathcal{C}$ to create a perturbed data set $\mathcal{D}_\sigma$:

$$\mathcal{D}_\sigma = \{(\mathbf{x}_j + \eta_j,\ c_j),\ \mathbf{x}_j \in \mathcal{C},\ \eta_j \in \mathcal{N}(0, \sigma)\} \tag{6}$$

Four such data sets $\mathcal{D}_{0.1},\ \mathcal{D}_{1.12},\ \mathcal{D}_{6.32}$, and $\mathcal{D}_{89.1}$ are shown in Fig. 4a). Their corresponding hierarchical cluster results are displayed below (Fig. 4b). Fig. 4c) displays the corresponding scores of the four experiments. It can be seen that the number of high splitting scores decreases as noise increases. The four experiments of Fig. 4a) to c) are integrated in Fig. 4d), where for each σ, the experiment is repeated 50 times, and the computed TIs are displayed in Box-and-Whisker plots. The fact that the TI decreases as noise increases makes the TI a reliable index to measure how well the label is reflecting the structure of the clustered data and how well a specific cluster tree is suited for visualization.

In order to demonstrate the applicability to larger data sets that are more realistic in real-world applications and to address the issue of scalability of the tree index, the experiment is repeated with $b = 60$ items for each class, resulting in a data set of 300 items. The Box-and-Whisker plots in Fig. 5 indicate that the TI produces qualitatively similar results compared to the data set with 40 items in Fig. 4. Uniquely the TI's amplitude is affected by the number of items in the data set.

3.2 Real-World Data

By applying the TI on real-world data sets, we simulate the scenario where a biomedical researcher is looking for the most appropriate algorithm and parameterization to visualize the cluster structure in the data.

The first data set is the breast cancer data set of *van de Vijver et al.* [34][1] which is an extension to the study of *van't Veer et al.* [35]. For each of the 295 subjects in the study, 24496 genes are analyzed and clinical data is available. In our study the clustering of subjects is performed on logarithms of ratios of a set of 231 marker genes identified by *van't Veer et al.* [35]. The logarithms are either scaled to $[-1, 1]$ (all normalization) or they are scaled separately to $[-1, 1]$ for each gene (single gene normalization). The data is separated into two classes of those tumors that develop metastasis and those which do not. We use this information as the external label $(\mathcal{C}_i)$ since the user seeks for groups of cases that have a similar genetic profile and are in the same tumor class.

The second data set is the multi-class cancer data set of *Ramaswamy et al.* [36] containing 288 subjects and 16063 genes. The data is separated into 22 different cancer types that are taken as external labels $(\mathcal{C}_0, \ldots, \mathcal{C}_{21})$. Thereby it is assumed that there is a correlation between the cancer type and the microarray data.

In order to create a large range of possible tree visualizations, two different preprocessings are applied (all and single gene normalization), and two different (dis-) similarity measures with five different scaling factors (see next paragraph) are used. The data set is clustered by the normalized cuts algorithm [8] applied in a hierarchical manner and by five variants of hierarchical agglomerative clustering (single linkage, complete

[1] Downloadable at `http://www.rii.com/publications/2002/nejm.html`

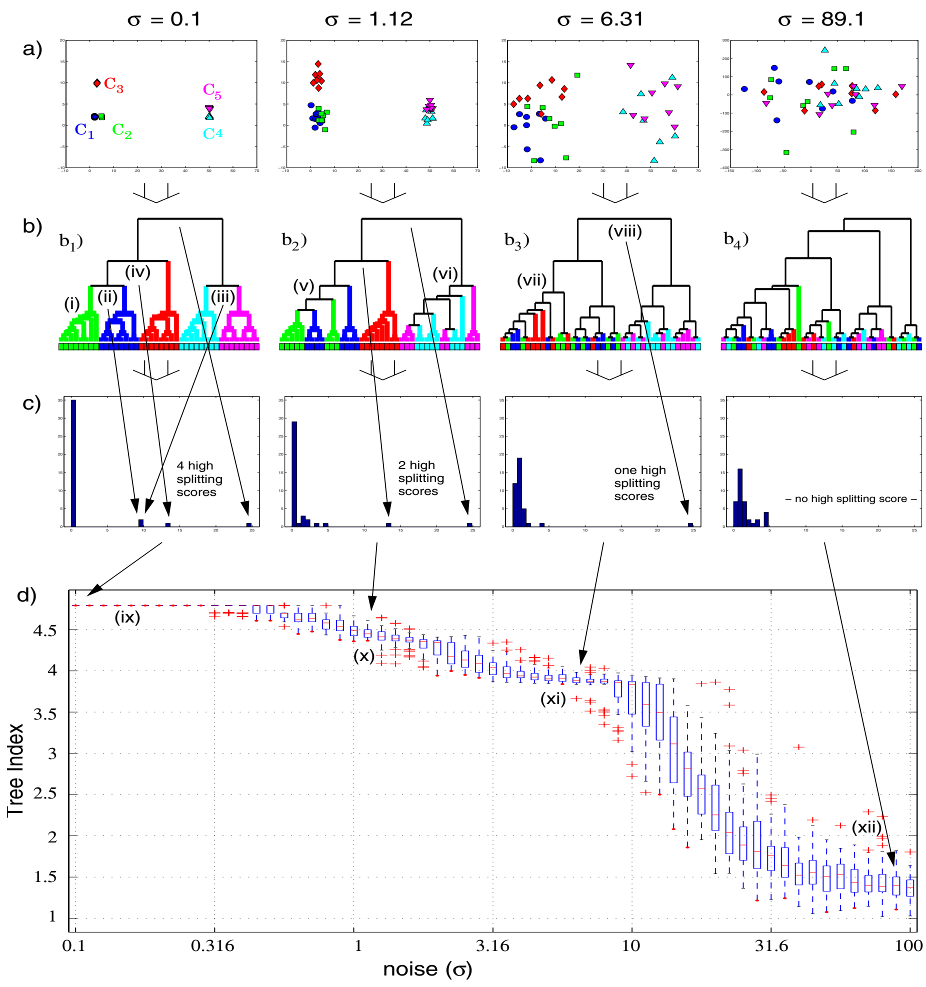


Fig. 4. Four perturbed data sets $\mathcal{D}_{0.1}$, $\mathcal{D}_{1.12}$, $\mathcal{D}_{6.32}$, and $\mathcal{D}_{89.1}$ are shown in **a)**. Their corresponding hierarchical cluster results are displayed below **(b)**. In **b**$_1$**)** the five classes are well separated. In a first step, the items of each class are grouped together (i). Then the classes $\mathcal{C}_1$ and $\mathcal{C}_2$ (ii) as well as $\mathcal{C}_4$ and $\mathcal{C}_5$ (iii) are linked to each other, followed by $\mathcal{C}_3$ that is linked to $\mathcal{C}_1$ and $\mathcal{C}_2$ (iv). As noise increases, $\mathcal{C}_1$ and $\mathcal{C}_2$ (v) as well as $\mathcal{C}_4$ and $\mathcal{C}_5$ (vi) cannot be separated any more by the cluster algorithm **(b**$_2$**)**. With a further increase of noise, $\mathcal{C}_3$ (vii) melts with $\mathcal{C}_1$ and $\mathcal{C}_2$ **(b**$_3$**)**, but $\mathcal{C}_1, \mathcal{C}_2$ and $\mathcal{C}_3$ are still separated from $\mathcal{C}_4$ and $\mathcal{C}_5$ (viii). Finally, with very high noise, an identification of the original classes is not possible any more **(b**$_4$**)**. **c)** displays the corresponding scores of the four experiments. It can be seen that the number of high splitting scores decreases as noise increases. The four experiments of **a)** to **c)** are integrated in **d)**, where for each σ, the experiment is repeated 50 times, and the computed TIs are displayed in Box-and-Whisker plots. Obviously, the TI decreases as noise increases. (ix) marks the position of the perfect separation of the clusters, (x) the position where $\mathcal{C}_1$ and $\mathcal{C}_2$ as well as $\mathcal{C}_4$ and $\mathcal{C}_5$ are combined in one cluster. (xi) marks the position where $\mathcal{C}_3$ cannot be separated from $\mathcal{C}_1$ and $\mathcal{C}_2$ any more and (xii) indicates a complete random clustering.

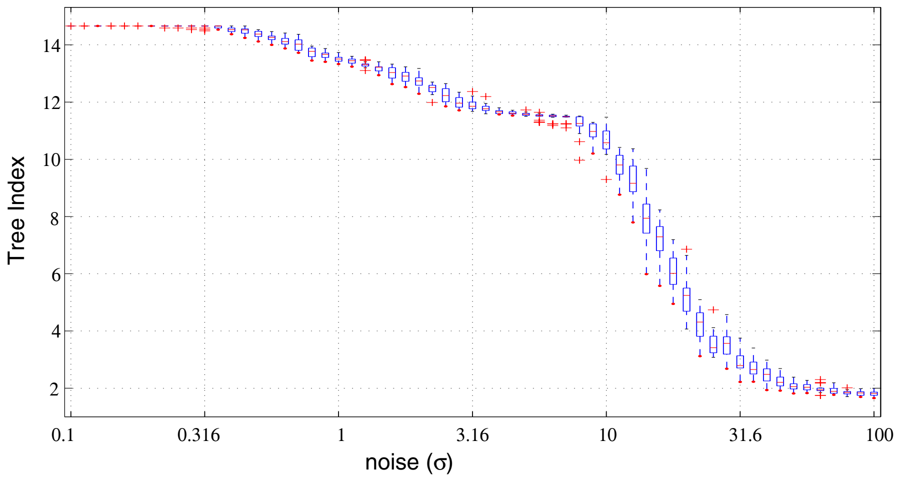


Fig. 5. The TI is applied to our simulated data set of 300 items. The Box-and-Whisker plots indicate that the TI produces qualitatively similar results compared to the data set with 40 items. Uniquely the TI's amplitude is affected by the number of items in the data set.

linkage, average linkage, Centroid, Ward). This results in a total of $2 \times 2 \times 5 \times 6 = 120$ cluster results. The cluster tree with the highest tree index is selected for final visualization.

Similarity and dissimilarity measures. Both the Euclidean distance and the Pearson correlation coefficient are used with a scaling factor specifying the sensitivity of the measures. The normalized cuts algorithm requires a similarity measure $w_{ij} \in [0,1]$ of two expression profiles $\mathbf{x}_i$ and $\mathbf{x}_j$ of dimension g whereas hierarchical agglomerative clustering requires a dissimilarity measure $d_{ij} \in [0,1]$. For our studies we apply $d_{ij} = 1 - w_{ij}$. The first similarity measure is defined as

$$w_{ij} = \exp\left\{-\frac{\mu(\mathbf{x}_i,\mathbf{x}_j)}{\sigma g}\right\}, \text{with } \mu(\mathbf{x}_i,\mathbf{x}_j) = \sqrt{\sum_{k=1}^{g}(x_{ik} - x_{jk})^2} \tag{7}$$

and scaling factor σ. The second similarity measure is based on the Pearson correlation coefficient [2], which corresponds to the intuitive understanding of correlation and is often used in the domain of microarray data analysis [7,10]. It is defined as

$$w_{ij} = \exp\left\{-\frac{1-\rho(\mathbf{x}_i,\mathbf{x}_j)}{\sigma}\right\}, \text{with } \rho(\mathbf{x}_i,\mathbf{x}_j) = \frac{1}{g}\sum_{k=1}^{g}\left(\frac{x_{ik} - \bar{\mathbf{x}}_i}{s_i}\right)\left(\frac{x_{jk} - \bar{\mathbf{x}}_j}{s_j}\right) \tag{8}$$

where $\bar{x}_i = \frac{1}{g}\sum_{k=1}^{g} x_{ik}$ and $s_i = \sqrt{\frac{1}{g}\sum_{k=1}^{g}(x_{ik} - \bar{x}_i)^2}$. In our study we use five different scaling factors $\sigma \in \{10^{-3}, 10^{-2}, 0.1, 1, 10\}$.

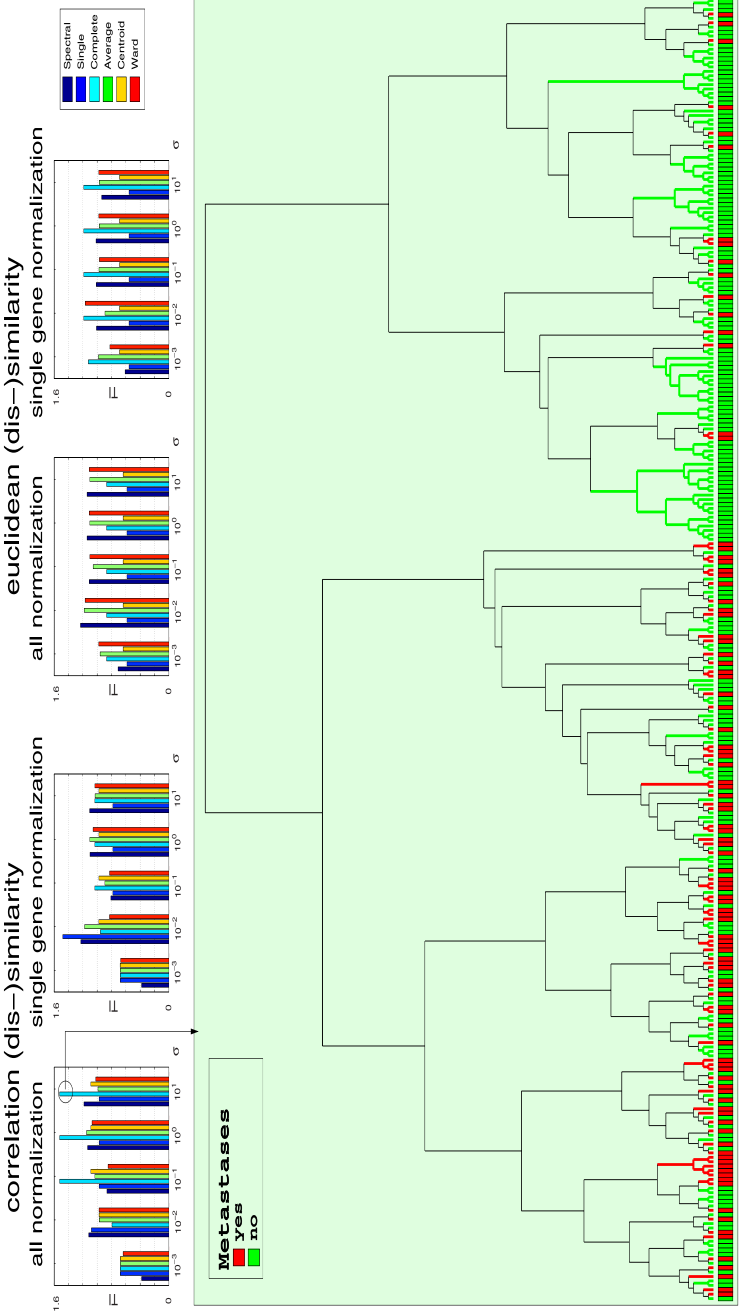


Fig. 6. The van de Vijver data set is clustered by different cluster algorithms and parameterizations. The highest tree index is obtained for complete linkage clustering, the correlation dissimilarity measure, all normalization, and a scaling factor of 10. In the cluster tree, the subjects are colored according to their category (metastasis or no metastasis). It can be seen that in the very first split the data has been separated in an nearly homogeneous cluster (many subjects without metastasis) and a heterogeneous cluster. Such a split obtains a high splitting score and increases the tree index considerably.

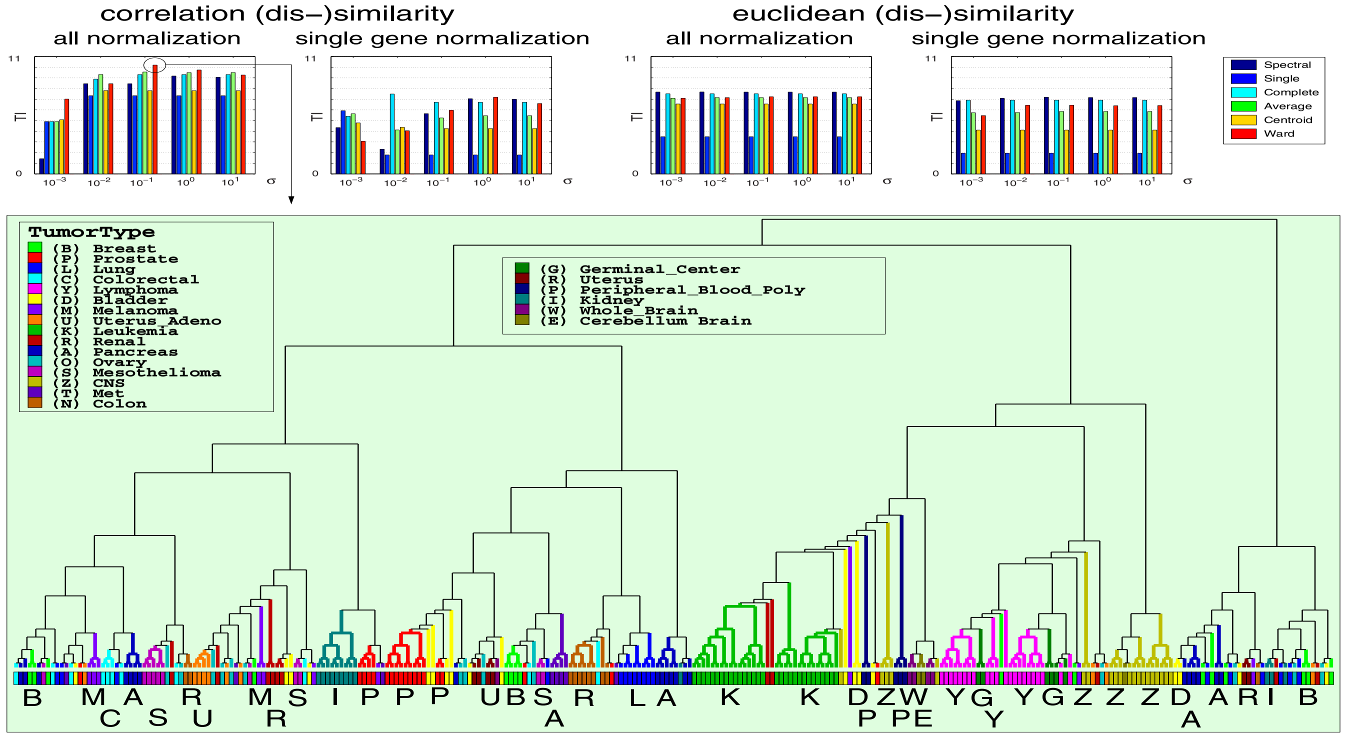


Fig. 7. The Ramaswamy data set is clustered by different cluster algorithms and settings. The highest tree index is obtained for Ward clustering, the correlation dissimilarity measure, all normalization, and a scaling factor of 0.1. In the cluster tree, the subjects are colored according to their category (tumor type). Additionally, homogeneous clusters containing two or more elements are labeled with letters. It can be seen that in various splits, homogeneous clusters are separated from the rest of the data. Such splits obtain high splitting scores and are responsible for a high tree index.

Results. Results of the van de Vijver breast cancer data set are displayed in Fig. 6. The highest tree index is obtained for complete linkage clustering, the correlation dissimilarity measure, all normalization, and a scaling factor of 10. In the cluster tree, the subjects are colored according to their category (metastasis or no metastasis). It can be seen that in the very first split the data has been separated in an nearly homogeneous cluster (many subjects without metastasis) and a heterogeneous cluster. Such a split obtains a high splitting score and increases the tree index considerably.

Results of the Ramaswamy data set are displayed in Fig. 7. The highest tree index is obtained for Ward clustering, the correlation dissimilarity measure, all normalization, and a scaling factor of 0.1. In the cluster tree, the subjects are colored according to their category (tumor type). It can be seen that in various splits, homogeneous clusters are separated from the rest of the data. Such splits obtain high splitting scores and are responsible for a high tree index.

4 Discussion

Hierarchical cluster algorithms are frequently used for clustering microarray data. Different cluster algorithms and parameterizations produce different cluster results. The algorithm and parameterization leading to the most appealing cluster visualization need to be detected according to a specific external label. An appealing cluster tree is characterized by splits dividing a heterogeneous cluster into nearly homogeneous subclusters regarding externally given additional variables which are interpreted as labels.

We propose a novel index, the tree index, which is based on the probability of each split. The tree index can identify the cluster algorithm and parameterization yielding the clustering best suited for visualization. In our study we varied the applied cluster algorithms, preprocessings and (dis-)similarity measures to create a large range of possible tree visualizations. Since the application of cluster algorithms and the computation of the tree index are not very time consuming and can be performed automatically for a large range of parameterizations, many more preprocessings and (dis-)similarity measures could be tested. Other important issues like gene selection or outlier deletion might also be considered to obtain cluster trees with even higher tree indices. The direct analysis of the structure of the cluster tree has the advantage that — in contrast to cluster indices that work on partitions — there is no need to estimate the number of clusters or to cut the cluster tree at some level.

In step 1, the splitting scores are computed using the probabilities of the splits, i.e. densities of the hypergeometric distribution. More robust splitting scores might be obtained using the $p-$value of the hypergeometric distribution. However, the computation of the $p-$value of the generalized hypergeometric distribution, as needed for $\kappa > 2$ categories, is not a trivial task.

In step 2, different scoring methodologies might be considered to compute the final tree index. When combining the R splitting scores to a vector of size R, the L_p-norm allows to define the final tree index in multiple ways. $p = 2$ leads to a result qualitatively similar to taking the standard deviation. $p = \infty$ is equivalent to judging the tree's quality exclusively by the maximal splitting score. More complex scoring methodologies might also take the tree level of the splitting scores into account.

The tree index might be compared with other cluster indices. However, such a comparison is not straight-forward since those indices have to be adapted in order to address the issue of cluster tree evaluation appropriately.

The tree index optimizes the tree structure of the cluster tree, not its display. The leaf ordering inside the cluster tree is still arbitrary. For final visualization we recommend the application of a leaf ordering algorithm [37] and an enhanced visualization technique with carefully selected graphical attributes (like color scale, line width, etc.).

In the presented examples single biological samples are clustered. The tree index might also be applied to trees clustering genes. A possible external label might be the primary function of a gene. Many databases exist for gene annotation and gene ontology [38,39]. However, a gene is usually involved in more than only one function or pathway and the gene annotations are still incomplete. Adaptations of the tree index are necessary to apply it with such multi-variate and incomplete external labels. Another application of the tree index is that it can be used to test the robustness of cluster trees. The influence of noise added to the microarray data or changing the scaling parameter of the (dis-) similarity measure have to be further examined.

5 Availability

An implementation of the tree index in Matlab can be downloaded at
`www.techfak.uni-bielefeld.de/ags/ani/projects/TreeIndex`

Acknowledgments

This research is supported by the International Graduate School for Bioinformatics and Genome Research at Bielefeld University, Germany as well as by the Technical Faculty of Bielefeld University, Germany. We thank Rosetta Inpharmatics and the Broad Institute who made available gene expression data and clinical data from cancer patients via internet, which was used in our study.

References

1. Quackenbush, J.: Computational analysis of microarray data. Nat. Rev. Genet. 2(6), 418–427 (2001)
2. Ochs, M.F., Godwin, A.K.: Microarray in cancer: Research and applications. Biotechn. 34, 4–15 (2003)
3. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern Classification. John Wiley and Sons, Inc., New York (2001)
4. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning Data Mining, Inference, and Prediction. Springer Series in Statistics. Springer, Heidelberg (2001) Fondi di Ricerca Salvatore Ruggieri - Numero 555 d'inventario
5. Jain, A.K., Murty, M.N., Flynn, P.J.: Data clustering: a review. ACM Comput. Surv. 31(3), 264–323 (1999)
6. Hartigan, J.A.: Clustering Algorithms. Wiley, Chichester (1975)
7. Eisen, M.B., Spellman, P.T., Brown, P.O., Botstein, D.: Cluster analysis and display of genome-wide expression patterns. PNAS 95, 14863–14868 (1998)

8. Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE PAMI 22(8), 888–905 (2000)
9. Kluger, Y., Basri, R., Chang, J., Gerstein, M.: Spectral biclustering of microarray data: Coclustering genes and conditions. Genome Res. 13(4), 703–716 (2003)
10. Xing, E., Karp, R.: CLIFF: Clustering of high–dimensional microarray data via iterative feature filtering using normalized cuts. Bioinformatics 17(suppl. 1), 306–315 (2001)
11. Handl, J., Knowles, J., Kell, D.B.: Computational cluster validation in post-genomic data analysis. Bioinformatics 21(15), 3201–3212 (2005)
12. Gat-Viks, I., Sharan, R., Shamir, R.: Scoring clustering solutions by their biological relevance. Bioinformatics 19(18), 2381–2389 (2003)
13. Halkidi, M., Batistakis, Y., Vazirgiannis, M.: On clustering validation techniques. Journal of Intelligent Information Systems 17(2-3), 107–145 (2001)
14. Goodman, L., Kruskal, W.: Measures of associations for cross-validations. J. Am. Stat. Assoc. 49, 732–764 (1954)
15. Calinski, R., Harabasz, J.: A dendrite method for cluster analysis. Comm. in Statistics 3, 1–27 (1974)
16. Dunn, J.: Well separated clusters and optimal fuzzy partitions. J. Cybernetics 4, 95–104 (1974)
17. Hubert, L., Schulz, J.: Quadratic assignment as a general data-analysis strategy. Br. J. Math. Stat. Psychol. 29, 190–241 (1976)
18. Davies, D., Bouldin, D.: A cluster separation measure. IEEE Trans. Pattern Recogn. Machine Intell. 1, 224–227 (1979)
19. Rousseeuw, P.: Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. J. Comput. Appl. Math. 20, 53–56 (1987)
20. Shamir, R., Sharan, R.: Algorithmic approaches to clustering gene expression data. In: Jiang, T., Smith, T., Xu, Y., Zhang, M.Q. (eds.) Current Topics in Computational Biology. MIT Press, Cambridge (2001)
21. Maulik, U., Bandyopadhyay, S.: Performance evaluation of some clustering algorithms and validity indices. IEEE transactions PAMI 24(12), 1650–1654 (2002)
22. Chen, G., Jaradat, S.A., et al.: Evaluation and comparison of clustering algorithms in analyzing ES cell gene expression data. Statistica Sinica 12, 241–262 (2002)
23. Bolshakova, N., Azuaje, F., Cunningham, P.: An integrated tool for microarray data clustering and cluster validity assessment. Bioinformatics 21(4), 451–455 (2005)
24. Bolshakova, N., Azuaje, F.: Estimating the number of clusters in DNA microarray data. Methods Inf. Med. 45(2), 153–157 (2006)
25. Rand, W.: Objective criteria for the evaluation of clustering methods. J. of the American Statistical Association 66, 846–850 (1971)
26. Hubert, A.: Comparing partitions. J. of Classification 2, 193–198 (1985)
27. Thalamuthu, A., Mukhopadhyay, I., Zheng, X., Tseng, G.C.: Evaluation and comparison of gene clustering methods in microarray analysis. Bioinformatics 22(19), 2405–2412 (2006)
28. Tavazoie, S., Hughes, J.D., Campbell, M.J., Cho, R.J., Church, G.M.: Systematic determination of genetic network architecture. Nat. Gen. 22, 281–285 (1999)
29. Toronen, P.: Selection of informative clusters from hierarchical cluster tree with gene classes. BMC Bioinformatics 5(1), (32) (2004)
30. Datta, S., Datta, S.: Methods for evaluating clustering algorithms for gene expression data using a reference set of functional classes. BMC Bioinf. 7(397) (2006)
31. Steuer, R., Selbig, P.H.J.: Validation and functional annotation of expression-based clusters based on gene ontology. BMC Bioinformatics 7(380) (2006)
32. Yeung, K., Haynor, D., Ruzzo, W.: Validating clustering for gene expression data. Bioinformatics 17(4), 309–318 (2001)

33. Johnson, N.L., Kotz, S., Balakrishnan, N.: Discrete multivariate distributions. Wiley, Chichester (1997)
34. van de Vijver, M.J., Yudong, D., van't Veer, L., Hongyue, D., et al.: A gene-expression signature as a predictor of survival in breast cancer. The New Eng. J. Med. 347(25), 1999–2009 (2002)
35. van't Veer, L.J., Dai, H., van de Vijver, M.J., He, Y.D., A.A.M.H., Mao, M., Peterse, H.L., van der Kooy, K., Marton, M.J., A.T.W.: Gene expression profiling predicts clinical outcome of breast cancer. Nature 415, 530–536 (2002)
36. Ramaswamy, S., Tamayo, P., Rifkin, R., Mukherjee, S., Yeang, C., Angelo, M., Ladd, C., Reich, M., Latulippe, E., Mesirov, J., Poggio, T., Gerald, W., Loda, M., Lander, E., Golub, T.: Multiclass cancer diagnosis using tumor gene expression signatures. PNAS 98(26), 15149–15154 (2001)
37. Ding, C.: Analysis of gene expression profiles: class discovery and leaf ordering. In: Proc. RECOMB 2002 (2002)
38. Mewes, H., Frishman, D., Guldener, U., Mannhaupt, G., Mayer, K., Mokrejs, M., Morgenstern, B., Munsterkotter, M., Rudd, S., Weil, B.: MIPS: a database for genomes and protein sequences. Nucleic Acid Res. 30, 31–34 (2002)
39. GO-Consortium: The Gene Ontology Consortium; Gene Ontology: tool for the unification of biology. Nat.Gene. 25, 25–29 (2000)

XMAS: An Experiential Approach for Visualization, Analysis, and Exploration of Time Series Microarray Data*

Ben Dalziel[1], Hui Yang[1,**], Rahul Singh[1,**], Matthew Gormley[2], and Susan Fisher[2,3,4,5,6]

[1] Department of Computer Science, San Francisco State University
huiyang@cs.sfsu.edu,rsingh@cs.sfsu.edu
[2] Departments of Cell and Tissue Biology
[3] Anatomy
[4] Obstetrics, Gynecology and Reproductive Sciences
[5] Pharmaceutical Chemistry
[6] Institute for Regeneration Medicine and the Human embryonic Stem Cell Program, University of California San Francisco. San Francisco, CA

Abstract. Time series microarray analysis provides an invaluable insight into the genetic progression of biological processes, such as pregnancy and disease. Many algorithms and systems exist to meet the challenge of extracting knowledge from the resultant data sets, but traditional methods limit user interaction, and depend heavily on statistical, black box techniques. In this paper we present a new design philosophy based on increased human computer synergy to overcome these limitations, and facilitate an improved analysis experience. We present an implementation of this philosophy, XMAS (eXperiential Microarray Analysis System) which supports a new kind of "sit forward" analysis through visual interaction and interoperable operators. Domain knowledge, (such as pathway information) is integrated directly into the system to aid users in their analysis. In contrast to the "sit back", algorithmic approach of traditional systems, XMAS emphasizes interaction and the power, and knowledge transfer potential of facilitating an analysis in which the user directly experiences the data. Evaluation demonstrates the significance and necessity of such a philosophy and approach, proving the efficacy of XMAS not only as tool for validation and sense making, but also as an unparalleled source of serendipitous results. Finally, one can download XMAS at http://cose-stor.sfsu.edu/~huiyang/xmas_website/xmas.html

1 Introduction

Microarray-based experimentation is a technique, which measures the expression levels for hundreds and thousands of genes within a tissue or cell simultaneously. It

[*] This research was partially funded by a grant from the National Science Foundation IIS-0644418 (CAREER) – Singh; SFSU CCLS Mini Grants – Yang.
[**] Corresponding authors.

therefore provides a data rich environment to obtain a systemic understanding of various biochemical processes and their interactions. Data from microarray experiments have been used, among others, to infer probable functions of known or newly discovered genes based on similarities in expression patterns with genes of known functionality, reveal new expression patterns of genes across gene families, and even uncover entirely new categories of genes [1], [2]. In more applied settings, microarray data has provided biologist with ways of identifying genes that are implicated in various diseases through the comparison of expression patterns in diseased and healthy tissues.

The area of microarray data analysis remains particularly active, leading to the development of numerous algorithms and software tools. The algorithmic underpinnings of these methods span a variety of pattern analysis, machine learning, and data mining methodologies including Bayesian belief networks (BBN), clustering, support vector machines (SVM), neural networks and Hidden Markov models. A survey of many of these techniques can be found in [1], [3], [4] and [5]. From a user perspective, a number of vendors have developed software systems for microarray data analysis such as Ingenuity [6], Onto-Express [7] and GenMAPP [8]. Furthermore, plug-ins have been developed for existing software systems such as the BioConductor [9] package for R [10], along with SAM [11] and PAM [12] for Excel.

Despite this un-arguable richness of analysis tools, it is acknowledged however, that analysis of microarray data is currently at a bottleneck [13]. Some of the most fundamental reasons behind this include:

- *Emphasis on the algorithmics to the exclusion of the user*: Holistically taken, most microarray analysis implementations are algorithm-oriented and do not provide sufficient support for exploration and/or hypotheses formulation. From an end user perspective, they function as a "black box" giving users very limited control over the analysis process outside what the underlying algorithmic mechanism is intended for. Among others, this limits the ability of users to integrate their domain expertise into the analysis process or explore alternatives which the algorithm design had not foreseen.
- *Interpretability*: Methods involving complex algorithms (such as BBN, SVM, and dimensionality reduction) may produce results that are difficult to interpret or understand. This can create a disconnect between the algorithmic process and the biochemical interpretability of the information.
- *Biased statistical analysis*: An important challenge outside the aforementioned user-centric issues lies in the fact that many existing techniques (e.g., SAM and PAM) employ statistical approaches to analyze microarray data. This can lead to bias, since in the majority of microarray studies the data is under-constrained (there are far fewer samples than genes or probes of interest). A representative example is the dataset used in this paper. It studies the placenta over the duration of pregnancy and is composed of just 36 samples containing expression levels for over 40,000 probes. As a result, it is difficult to construct reliable statistical samples or assume a reasonable data distribution model to carry out further analysis.

Given the aforementioned context, we propose re-thinking the design philosophy for developing microarray data analysis systems. Our central observation notes the fact that computers are inherently strong at large scale processing, data storage and data integration. However they lack the human skills of contextual reasoning, pattern detection, hypotheses formulation, exploratory behaviors, and sense making. Thus the primary design goal we seek to establish is the ability to exploit human-machine synergy by taking advantage of the aforementioned complementarities.

In the area of human-computer interactions, such an emphasis on exploration and hypothesis formulation in data rich environments has been the focus of study in [14] and [15], where the term "experiential environment" was used to denote systems and interfaces that take advantage of the human-machine synergy and allow users to use their senses and directly interact with the data.

In this paper, we describe the anatomy of a microarray data analysis system called XMAS (eXperiential Microarray Analysis System) that is developed by using and extending the ideas of experiential computing. The proposed system is (1) direct in that it does not use complex metaphors and commands; (2) supports unified query and presentation spaces; (3) maintains user context; (4) provides external contextual information through assimilating a variety of supplementary data such as pathway data from the Kyoto Encyclopedia of Genes and Genomes (KEGG) [16], and (5) supports algorithmic and user-directed analysis, exploration and hypotheses formulation. Our ultimate goal is to promote perceptual analysis by integrating the user directly in an interactive and reflexive visualization environment with powerful algorithmic capabilities. XMAS is not limited to the analysis of time series microarray data, and can be more widely applied to any time-series datasets. XMAS supports the following visualization and analyses:

- *Trajectory based gene clustering*: In time series microarray data, a trajectory is composed of a sequence of expression measurements collected at different time points for a certain probe or gene. It is essentially a time series of gene expression data w.r.t. a single probe or gene. This function clusters different genes according to the relative geometric similarity of their expression trajectories.
- *Data filtering*: This can be based on gene identifiers, pathways, and integrated or user defined annotations. These filters facilitate the specification of genes of interest, enabling the user to narrow down hypotheses. This functionality extends to support any integrated secondary data.
- *Interestingness evaluators*: XMAS implements a set of measurements such as Pearson's correlation and p-value to quantify the interestingness of the results, to aid the user during visual inspection and more generally the entire analysis process.
- *Visualizations*: Two primary visualizations provide interactive representations of data at different resolutions including (1) a discretized trajectory view; and (2) a precise gene expression view.
- *Interactions*: Users can directly manipulate, interact and explore the data using highly intuitive point-and-click interactions.

There exist systems which support some of the features described above. For example the commercial system OmniViz [17] offers various reflective and interactive visualizations in addition to the more traditional statistical measures and algorithmic capabilities. Systems which share this closer resemblance to XMAS lack the core experiential design philosophy, which in turn has a significant influence over the completed system in the following areas: interaction, visualization, data integration, and interoperability. This will become apparent through the remainder of the paper.

Through use of XMAS, users are expected to achieve three main goals: (1) to gain a deeper understanding of a time series microarray dataset; (2) to verify or compare phenomena reported in literature on comparable datasets; (3) to generate hypotheses through examining results from different analyses. The main contributions of this work include: (1) increased user involvement, comprehension and understanding through development of a new design philosophy for microarray data analysis; (2) improved biological results from analysis; and (3) a concrete web-based extensible implementation of this design philosophy. This paper goes on to describe XMAS; its fundamental components and associated combinatorial power in Section 2. In Section 3 experimental results and user evaluation are presented to demonstrate the efficacy of this approach.

2 System Description

XMAS is an experiential system for time series microarray data (TSMAD) analysis through realizing a collection of interactive visual data operators and assimilating different types of knowledge such as pathway information. As shown in Fig. 1, XMAS consists of the following main modules: (1) data preprocessing; (2) a collection of interoperable data operators, including a parameterized discretization operator, basic data integration operators, and trajectory-oriented data operators; (3) interestingness evaluators; and (4) visualization and Human Computer Interaction (HCI). Next, we first discuss the datasets utilized by XMAS, and then describe in detail its main modules.

2.1 Data Sets

XMAS focuses on the analysis of time series microarray data. Such data has been used to study the developmental nature of an organ (e.g., a cancerous tissue) by conducting Microarray experiments on samples drawn from this organ over time. The

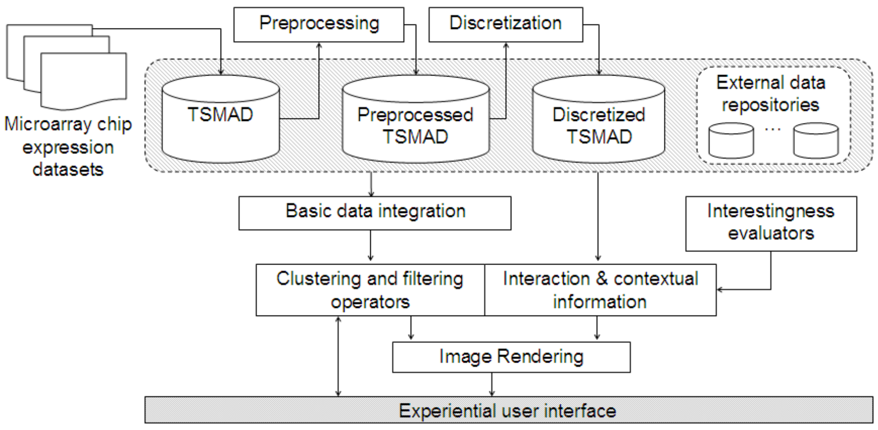

Preprocessing
Discretization
Microarray chip expression datasets
TSMAD
Preprocessed TSMAD
Discretized TSMAD
External data repositories
Basic data integration
Interestingness evaluators
Clustering and filtering operators
Interaction & contextual information
Image Rendering
Experiential user interface

Fig. 1. System overview of XMAS

genes of interest are generally specific to a study, which in turn determines the set of probes on a microarray chip that one is interested in looking into.

Let ***D*** be a TSMAD, $P=\{p_1, p_2, .., p_M\}$ be the set of M probes of interest, $T=<t_1, t_2, .., t_Q>$ be the ordered Q time points when Microarray analyses are conducted, and $S_i=\{ s_1, s_2, .., s_{Ni} \}$ be the set of N_i samples at *time* $t_i \in T$. Note that S_i and S_j ($i \neq j$) might be two different sets due to restrictions on acquisition of live tissues. Then ***D*** can be considered as a dataset of *M* time series, each of which corresponds to one probe and is referred to as a *complex probe trajectory*. For each probe $p_k \in P$, its trajectory has *Q* time points. Each time point is associated with a vector of N_i expression values, corresponding to the N_i samples at this point. To further enhance users' explorative power, and analysis experience, XMAS integrates a variety of existing domain knowledge such as a mapping database between the probe set *P* and the set of genes, and pathway data from KEGG. XMAS adopts MySQL, an open source RDMBS, to manage such data.

2.2 Data Preprocessing

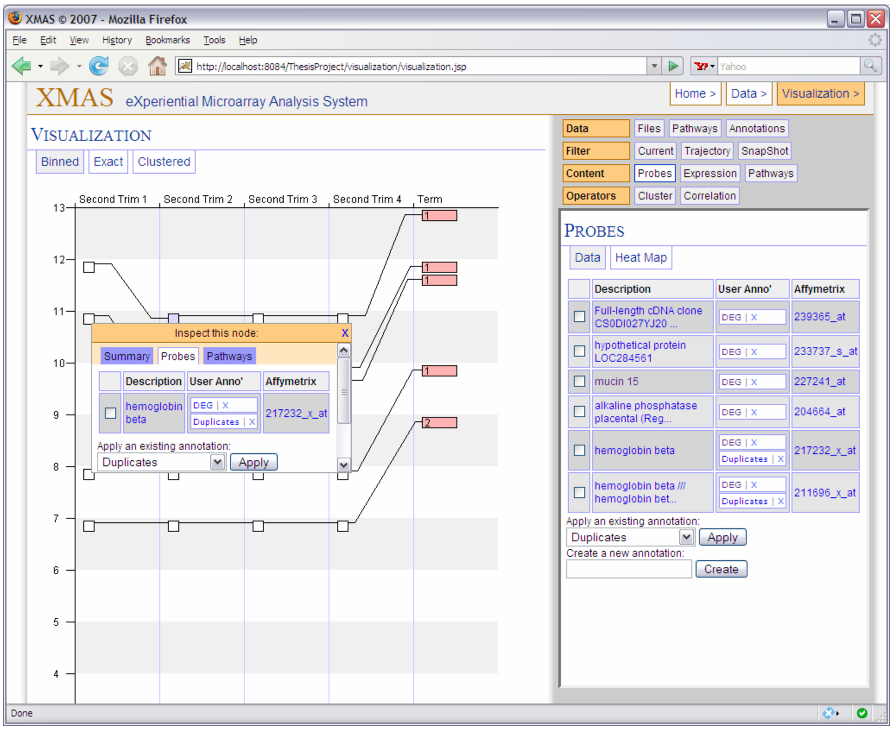
Given a TSMAD ***D***, this module first performs a base-2 logarithmic transformation over each expression value in ***D***. It then applies a simple data reduction technique to reduce each complex probe trajectory to a simple time series. Specifically, for a given complex trajectory, it replaces the vector of expression values at each time point by the median of this vector. One main reason the median is chosen is that it is more noise-tolerant. For the remainder of this paper, we refer to such simple time series as *simple probe trajectories* or *probe trajectories*. This process simplifies analysis at a global level, where the median expression is a reasonable representation of the constituent samples. Complete expression levels are preserved within XMAS and are accessible to aid in more concentrated analysis.

2.3 Interoperable Data Operators, Visualization and HCI

Interoperable data operators, intuitive visualization, and user-friendly HCI support form the core of XMAS. XMAS consists of data operators that can both function individually and collaborate with others when combined at users' command. Unlike most existing software systems for Microarray data analysis, XMAS injects visualization and HCI into data analysis. Therefore, users can not only visually observe the results at any moment, but also be able to interactively respond to XMAS to design their own explorative paths towards concept validation or hypothesis generation. It is due to this tight coupling of data operators, visualization and HCI, we will describe each data operator by also including the other two aspects.

Parameterized data discretization: One main interest in studying TSMADs is to characterize the temporal movement of genes in terms of expression level. Given that the collection of genes under study can be large, for instance, in the order of tens of thousands, examining a dataset on a trajectory-by-trajectory basis is time consuming and difficult. In addition, one also needs to reduce the impact from noise in the data. To address such issues, XMAS first applies equi-width discretization to each probe contained within the preprocessed TSMAD, where the width *w* (applied globally) is a user-specified parameter. The result of this intuitive probe association operator is a collection of *discretized probe trajectories*, where each expression level is represented by an integer, corresponding to its discretized value. The issue of information loss inherent to such discretization is countered through the preservation of the precise expression values which can be exposed through visualization or inspection.

Fig. 2 shows part of a screenshot of such discretized trajectories. In this figure, each discretized value (or bin) occupies one row space. Small squares or nodes in each bin can be clicked to reveal all the probes whose expression levels fall into this bin at a give time point. Moreover, all the nodes are arranged from left to right in columns, with the *i*th column corresponding to the *i*th time point. A node is colored in red if its expression level is higher than the previous node on a trajectory and blue if it is lower. The probes in the first node in a row share discretized expression value at the first time point. The probes in each of the rightmost nodes share identical discretized trajectories. And the probes in each of the middle nodes share a partial trajectory prior


XMAS © 2007 - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Yahoo
XMAS eXperiential Microarray Analysis System
Home > Data > Visualization >
VISUALIZATION
Binned Exact Clustered
Second Trim 1 Second Trim 2 Second Trim 3 Second Trim 4 Term
13 12 11 10 9 8 7 6 5 4
1 1 1 1 2
Inspect this node: X
Summary Probes Pathways
Apply Create
Data Files Pathways Annotations
Filter Current Trajectory SnapShot
Content Probes Expression Pathways
Operators Cluster Correlation
PROBES
Data Heat Map
Done

http://localhost:8084/ThesisProject/visualization/visualization.jsp

	Description	User Anno'	Affymetrix
☐	hemoglobin beta	DEG | X Duplicates | X	217232_x_at

Apply an existing annotation: Duplicates

	Description	User Anno'	Affymetrix
☐	Full-length cDNA clone CS0DI027YJ20 ...	DEG | X	239365_at
☐	hypothetical protein LOC284561	DEG | X	233737_s_at
☐	mucin 15	DEG | X	227241_at
☐	alkaline phosphatase placental (Reg...	DEG | X	204664_at
☐	hemoglobin beta	DEG | X Duplicates | X	217232_x_at
☐	hemoglobin beta /// hemoglobin bet...	DEG | X Duplicates | X	211696_x_at

Apply an existing annotation: Duplicates

Create a new annotation:

Fig. 2. The XMAS analysis environment is divided into three primary regions: (1) the visualization space displays discretized or precise trajectory views. Visualizations in this space can be manipulated in a similar way to various interactive web based mapping applications. This accommodates larger visualizations than would be practical in a static environment. Each node in the primary visualization is interactive, allowing the user to inspect content through in-place context windows (2). A complementary view, the visualization sidebar (3), provides similar data for the entire visualization. Operator specification tools in addition to operator summaries and correlation data are also accessible from this space.

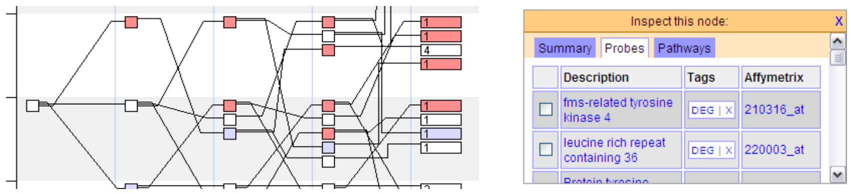
to that time point. All such nodes are expandable. Note that the system calls several operators described later to construct those nodes.

Basic data integration operators: The operators contained within this category realize integration of different datasets. They can be categorized as follows:

- *Gene-probe integrators:* These operators relate probes to genes or vice versa, for instance, identifying the list of probes associated with a given gene.
- *Probe-gene-pathway integrators:* This set of operators enriches a gene or probe with pathway information. For instance, one such operator determines whether a given gene participates in a pathway; whereas another operator lists all the genes or probes that are involved in a pathway.
- *Trajectory-trajectory integrators*: These operators relate the three forms of probe trajectories utilized by XMAS: complex, simple and discretized probe trajectories.

Trajectory-oriented data operators: This set of operators support users to explore the data by examining and uncovering the similarity among probe trajectories.

- *K-means clustering*: This operator puts probes of similar, non discretized trajectories into the same group. The user can choose to cluster based on either Euclidian or Pearson's Correlation distance metrics and can specify the value of K.
- *Expression level preserving trajectory-based clustering:* This operator identifies the genes whose discretized probe trajectories are identical and associates them in a single cluster. Two trajectories are identical if they have the same expression level at each time point. Fig. 2 shows examples of such clusters, each corresponding to one trajectory. One can inspect the probes and related contextual information in a cluster by clicking the corresponding node.
- *Trajectory shape based clustering:* This operator finds similar shaped trajectories across possibly different expression values. Probes of the same trajectory shape are essentially co-expressed at each time point. Therefore, each of such clusters identifies one co-expression pattern. We implement this operator in two steps. It first vertically translates all the discretized probe trajectories in a way such that the first node of each trajectory corresponds to the same expression level 0. For instance, for a given trajectory <2, 3, 1, 3, 4>, its translated trajectory is <0, 1, -1, 2>. The second step finds such clusters by calling the previous clustering operator. Fig. 3 shows part of a screenshot of such clusters. One can view the content of each cluster by expanding each of the rightmost nodes.



	Description	Tags	Affymetrix
☐	fms-related tyrosine kinase 4	DEG | X	210316_at
☐	leucine rich repeat containing 36	DEG | X	220003_at

Fig. 3. Trajectory shape based clustering translates trajectories to a common root. Each node is interactive, revealing contextual data about the content of the node as a mobile, in-place window in the visualization.

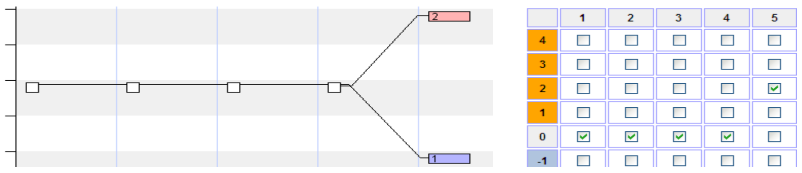


Fig. 4. Shape based trajectory specification, reveals 2 clusters of inverse trajectory shape

- *Discovery of inversely expressed probes/genes*: This operator identifies probes whose discretized trajectories are the inverse of each other. Fig. 4 shows the interactive query space and corresponding visualization showing five probes expressing with perfect discretized inverse correlation to twelve others.
- *Filtering operators:* Such operators utilize one or more basic data integration operators described earlier to identify trajectories that satisfy certain specified criteria. All such operators are integrated into one interactive user interface as shown in Fig. 2. XMAS currently supports the following filtering operators:
  - *Filtering by probes or genes*: This identifies probe trajectories associated with one or more specified genes.
  - *Filtering by pathway*: This identifies probes involved in a specified pathway
  - *Filtering by gene expression movement*: This identifies probes that are partially or entirely co-expressed. Fig. 4 illustrates the interface where users can specify a specific co-expression pattern of interest. This filtering operator can be applied to strictly trajectory-based clusters, or trajectory shape based clusters, as illustrated in Figs. 2 and 4 respectively. In Fig. 2, a user is interested in identifying all the probes or genes with a relative movement of 2 between the last two discretized expression levels. Fig. 4 illustrates the ability to include all the inversely expressed genes, this time for shape based clusters (i.e. with the same root). A similar operator is also included where one can identify the probes that have a similar expression level at one or more time points by specifying the range of expression levels at such time points.
  - *Exclude a probe from the resulted probe set*: This operator removes a probe from analysis. In Fig. 3, one can remove a probe by clicking the ‘x’ symbol.

Note that all the above data operators are interoperable with each other. This is essential, as XMAS does not prescribe data discovery paths for users. Instead, it empowers users to construct their own discovery paths by combining different operators in different order. XMAS achieves this by accommodating an integrated user interface shown in Fig. 2.

2.4 Interestingness Evaluators

Although visualization is powerful and intuitive for users to gain insight into a dataset, its effectiveness can be greatly reduced in a variety of situations. For instance, the amount of the data being visualized is too large to fit into a computer screen. In some cases, data might exhibit an inherently complex structure such that it is difficult for human beings to make sense of the visualized data. To overcome this limitation, XMAS includes a collection of evaluators to quantify the results.

- *Volatility of a trajectory*: Let $TR=<e_1, e_2, ..., e_Q>$ represent a discretized trajectory, where e_i is the expression value at the ith time point. The volatility of this trajectory is defined as $\Sigma_{i=1,Q-1}(|e_i-e_{i-1}|)$. One can use this measure to identify probes with extremely low or high volatility, where the former might not be of much interest and the latter might be a result of noise in the dataset.
- *Precision and recall*: These two measurements are used to quantify the strength of association between a pathway and the set of probes produced by a data operator. Let P be the pathway of interest and x be the number of participating probes of P. Let y be the number of probes returned by a certain operator, among which z probes are associated with P. Then Precision=z/y and Recall=z/x.
- *Pearson's correlation coefficient*: Let $X=<x_1, x_2, ..., x_Q>$ and $Y=<y_1, y_2, ..., y_Q>$ be two probe trajectories. One can use this evaluator to measure the direction and strength of the linear relationship between *X* and *Y*.
- *Identification of differentially expressed genes (DEGs)*: DEGs are selected by determining the moderated t statistic-adjusted P values (<0.05 using Bonferroni correction [18]). Fig. 2 highlights the DEGs within the current analysis, as leaf annotations in the primary visualization (1), and as “tags” in the list view (3).

P-value: We adopt P-values to measure the statistical and biological significance of observing a set of probes being associated with each other by a clustering operator described earlier. Given a background distribution, the lower the p-value, the more unlikely that observing a set of probes associated with each other is by chance. We next use the pathway annotation as an example to explain how P-values are computed. Let *N* be the number of probes under study, *D* be the number of probes in a given pathway, *n* out of these *N* probes are associated with each other by a data operator, and finally, *k* out of these *n* probes are also in the said pathway. The P-value of this association of n probes is then defined as: $P-value = \binom{D}{k}\binom{N-D}{n-k} \binom{N}{n}$

2.5 Interoperability, Interactivity and Extensibility of XMAS

Interoperability among data operators: Unlike most existing software tools for TSMADs, XMAS does not prescribe analytical tasks for users. Instead, it empowers users to construct their own data discovery paths tailored for their special needs by combining different operators in different orders. XMAS achieves this by realizing interoperable data operators and an integrative user interface shown in Fig. 2. Aided by visualization, users can use this interface to select a sequence of data operators that are most likely to maximize their understanding of a problem at hand. A use case is described in detail in section 3 to illustrate this feature and its advantages.

Interactivity: Interactions with operators in XMAS are direct, i.e., no complex metaphors are involved. In addition, XMAS maintains contextual information on both users’ behavior and data produced from such behavior. This ensures that there is no unnecessary context switching, thereby reducing the cognitive load from users.

Extensibility: Due to its modular architecture design (Fig. 2), XMAS can be readily extended in one or more of the following aspects: (1) integrate additional supplementary datasets such as gene ontology (GO)[19] functional categories and implement new data integration operators to enrich users' analytical experience; (2) integrate new data operators; and (3) realize additional interestingness evaluators.

3 Experimental Evaluation

XMAS' analytical power lies in the union of three areas: (1) visualization; (2) interactivity; and (3) interoperability. As discussed earlier, existing algorithms and software systems lack some or all of these desirable components. Considering the general trends in the state of the art: user interaction is limited to data entry, parameter specification and analysis via a simple (text based) command driven interface. Workflow is linear and disjoint, (often spread over numerous systems), and data presentation is generally textual (with notable exceptions such as pathway visualization in GenMAPP).

In this section we present evaluation of XMAS which demonstrates the importance and necessity of having the three areas coexist. First, we describe how XMAS can be used as an interactive visual tool to foster a greater breadth and depth of understanding within microarray data. Second, a common information goal serves as the entry point to a highly-non-traditional workflow drawing on many interoperable components of XMAS. Finally, comparative quality information is presented to support the generated hypotheses. Throughout, the inherent facilitation of hypothesis generation and serendipitous discoveries are highlighted. All evaluations were performed on the data set described below.

3.1 Data Description

To demonstrate the efficacy of XMAS, we used it to analyze a publicly available TSMAD [GEO Accession No: GSE5999] which captures expression data of human placentas during pregnancy. Using the description of a TSMAD provided in section 2.1, five time points (Q=5), comprising N_1=6, N_2=9, N_3=6, N_4=6, and N_5=9 samples capture genome wide (45,000 probes representing 39,000 gene transcripts) expression profiles of non-contiguous placentas between 14 and 40 weeks of pregnancy. The 5 distinct gestational time intervals (Q) range between 14-16, 18-19, 21, 23-24, and 37-40 weeks. The experiments which compose Q=1 through Q=4 capture the stage of pregnancy known as midgestation, and the samples from Q=5 are contained within the third trimester, also known as Term. For complete experimental protocol, description and analysis workflow, readers are referred to [20]. The findings on this dataset, reported in [20] will be cross-referenced where necessary. The dataset was first preprocessed as described in Section 2.2. It was then discretized as explained in Section 2.3. Throughout the following evaluation, a bin size of 1 (i.e., w=1) was used.

3.2 XMAS as a Visual Interactive Tool to Aid in Data Comprehension

Developing a detailed understanding of a TSMAD is an important step towards generating focused analysis and hypotheses. Traditionally, the development of a broad and formal understanding is based almost exclusively on the dissection of output from utilizing a variety of analysis systems and algorithms. In contrast, XMAS provides an integrated environment to facilitate this process. We next describe two scenarios (among many), where XMAS is being used to help expert users gain both a global and localized view of the data and many times serendipitous discoveries.

***Expression pattern knowledge discovery*:** Visualization of discretized trajectories and shaped based trajectory clustering (i.e. unique trajectories) provide a global view of the entire dataset (Fig. 2(1)). As the user began to specify the operators (Section 2.3), the reflective query space updated to indicate the quantity of probes, DEGs and unique trajectories that would match the defined operator (Fig. 2). This reciprocal interaction aided the user to gain insight into the distribution of probes, DEGs, and the variability of probe expression during the specification refinement process. For instance, with 2 mouse clicks—one for the discretization operator and the other for the shape-based trajectory clustering--XMAS reveals that there are 76 distinct expression patterns and 504 DEGs in the dataset. Using the filter as shown in Fig. 4, more detailed information of such patterns were identified within a few mouse clicks: 6 patterns showing a significant expression increase ($\geq$ 4-fold) at Term, 11 showing an expression decrease ($\geq$4-fold) at Term, and only 1 showing a 16-fold increase. One more click revealed that only one probe involved in the last case. Such information provides the user with an insight into both the global *and* localized behavior of their data. This is in sharp contrast to traditional analyses, where such information is gleaned through utilizing a number of tools. Additionally, due to effective integration of user knowledge, our evaluation has shown that XMAS can often uncover previously unknown, yet interesting patterns in the data, thereby leading to serendipitous discoveries.

***Pathway involvement analysis*:** The identification of known biological processes (or pathways) involved in a TSMAD is one main goal in microarray analysis. Following the identification of such pathways, domain users often find it necessary to further support such identification by investigating the relative involvement of each pathway in the context of the entire data set (i.e. not exclusive to DEGs, which are traditionally the sole focus of pathway analysis such as GenMAPP). This is generally a labor-intensive and manual process, which can take up to several hours and may become impractical for large pathways. We next use the *Apoptosis* pathway as an example to demonstrate how XMAS can significantly improve in this respect.

As illustrated in Fig. 5, we first used the pathway membership filter to identify the 631 probes involved in the *Apoptosis* pathway, among which 8 were annotated as DEGs. We then inspected the annotations accompanying each discretized trajectory in the visualization, to ascertain the quantity of probes sharing DEG expression profiles (at the discretized level). This, the user determined, was a good way of assessing the relative involvement of the entire pathway. Individual probes were subsequently re-included into analysis, enabling visual assessment on a probe-by-probe basis.

This simple concatenation of operators led to a focused analysis of pathway involvement, reducing what was previously a multi hour process to a few interactions (mouse clicks). Too often, traditional analysis concentrates exclusively on DEG lists, and here, simple trajectory association enabled the user to surround DEGs with

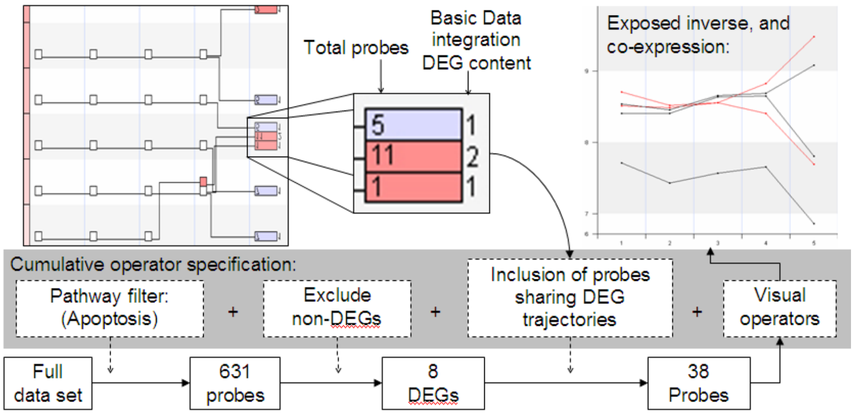

Total probes
Basic Data integration DEG content
5 1
11 2
1 1
Exposed inverse, and co-expression:
Cumulative operator specification:
Pathway filter: (Apoptosis)
+
Exclude non-DEGs
+
Inclusion of probes sharing DEG trajectories
+
Visual operators
Full data set
631 probes
8 DEGs
38 Probes

Fig. 5. User led analysis quickly identified DEG involvement in a given pathway. Probe context information presented within the visualization enables the user to pull in similar, yet non-DEG probes to focus an analysis on the relative involvement of the pathway as a whole.

contextually similar probes. Analysis of these probes facilitated a more confident declaration of significance, and led to the specification of a subset of probes which could form the basis of subsequent analysis.

3.3 Negative Expression Shift Approaching the End of Pregnancy

In this and the following sections, we described a complete workflow to illustrate the power of discovering serendipitous knowledge as a direct consequence of the integration of visualization, interactivity, and interoperability among data operators. Such integration enables a highly focused, yet simple analysis, which leads to the exposure of pathway involvement, hypothesized crosstalk, and co-expression patterns. These types of knowledge could not be reasonably developed by traditional means. The user workflow is described below and illustrated in Figs. 6 and 7.

Towards the end of pregnancy, the placenta begins to shut down in preparation for delivery. This process materializes at the genetic level as placental cells switch off, and is observed as a shift in expression between the second trimester intervals and term (time period 5) 0. The entry point to this analysis was to identify such probes.

Traditionally, such analysis involves the reduction of the data set into two representative samples, between which the expression characteristic can be evaluated. However, considerable details can be lost in this process. The analysis from 0, for example, assumed constant expression during midgestation, reducing 27 samples to just one. This is not the case, as one can observe directly within XMAS (Fig. 3). Furthermore, the lack of interaction in traditional analyses heavily restricts the users' ability to obtain a greater sense of completeness.

As shown if Fig. 6, we first performed a trajectory shape-based clustering to identify 39 probes that show a 4-fold or more increase at Term, of which 19 are DEGs. The visualization based contextual information further verified that the clustering

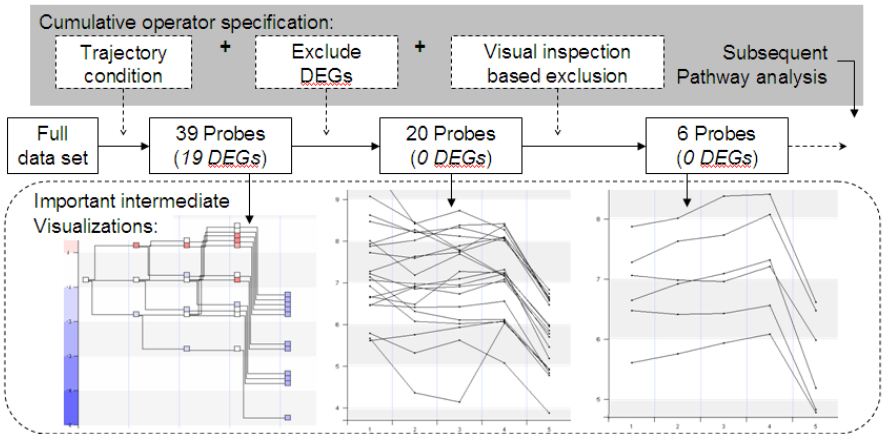

Cumulative operator specification:
Trajectory condition
+
Exclude DEGs
+
Visual inspection based exclusion
Subsequent Pathway analysis
Full data set
39 Probes (19 DEGs)
20 Probes (0 DEGs)
6 Probes (0 DEGs)
Important intermediate Visualizations:

Fig. 6. Workflow, illustrating the specification of operators used to focus analysis. Correlation scores demonstrate the power of exploratory analysis to expose common patterns of biological interest based on simple interactions. Key visualizations provide an insight into the environment in which the user is empowered to apply domain knowledge.

captured the target characteristic well. DEGs were subsequently excluded from analysis to concentrate on the remaining 20 candidates, as they share similar expression patterns with DEGs yet not categorized as DEGs. Through visual inspection of precise probe trajectories the user was able to exclude probes judged to be of lesser interest in the context of the current analysis. Interactively, we focused on the emergence of a specific trajectory shape, shared by 6 probes. Correlation analysis verified and strengthened this association. Through this process (Fig. 6), XMAS enables direct application of domain knowledge and intuition from the domain user. This is unmatched by other systems.

Main Observations: The quantity of discretized trajectories represented by the 39 probes (Fig. 6) indicates the details lost in traditional methods. XMAS facilitated a less strict, more intuitive specification of characteristics, which accommodated a greater sense of completeness than traditional analysis is capable of establishing. Furthermore, probe membership information, such as DEG content, was integrated into the analysis/query space in various ways. These provided valuable contextual information which aided the user in the decision making process. The 6 probes identified earlier were of great interest to domain experts, due to the reason that will be discussed in Section 3.4. Again, such probes would be unlikely to be associated without the direct application of user knowledge and intuition.

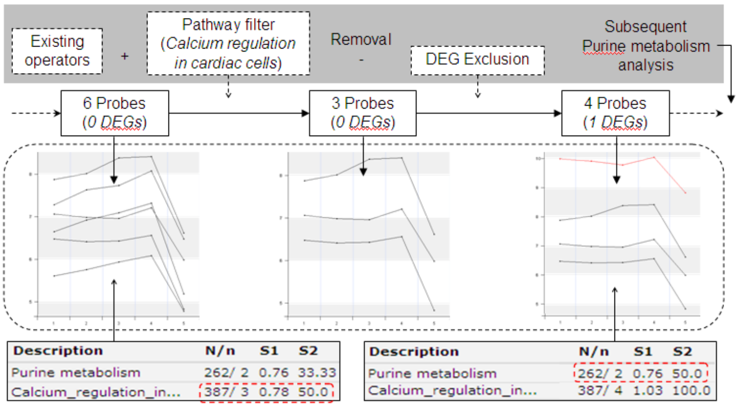
3.4 Interoperable Pathway Analysis

Biologists commonly want to identify the involvement of known biological processes in the observed time series. Systems such as GenMAPP, Ingenuity and GSEA provide mechanisms by which such pathways can be exposed, yet analysis within such systems is generally confined to DEGs. Statistical methods are employed to expose the most "significant" pathways represented, but issues relating to the completeness and quality of such subsets are here compounded.

Pathway analysis within XMAS can center on DEGs, as per traditional analysis, but is equally applicable to sets of probes sharing other characteristics – demonstrating the core value of interoperability. The power of XMAS to facilitate the exposition of probesets with significant commonality beyond, or in addition to differential expression, was explored in the previous scenario. Based on a serendipitous discovery, this scenario is extended, illustrating pathway analysis functionality within XMAS. This process is illustrated in Fig. 7.

It was indicated by the pathway membership view accompanying the visualization of the six probed from the previous use case (see Fig. 7), that the set has a significant three probe overlap with the pathway of *Calcium regulation in cardiac cells*. Interestingness measures provided quantitative support for the discovery, and the application of a corresponding pathway filter concentrated analysis on the three matching probes. DEG probes were reintroduced into the analysis space, revealing a single DEG sharing the developed characteristics. The appropriateness of the association of the additional DEG with the existing three probes was confirmed visually, and with the aid of the correlation matrices.

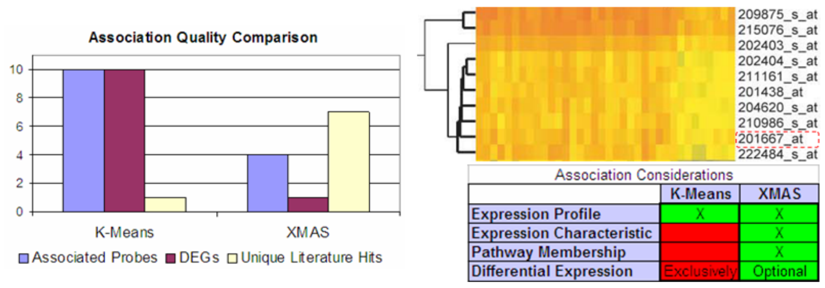
Serendipitous Discoveries: The exposure of 6 non-DEG probes, with a shared trajectory characteristic and expression profile led to the analysis of a pathway, which was unlikely to be judged significant by traditional analysis that focuses entirely and globally on the set of DEGs. The workflow that led to the association of non-DEGs with DEGs provided evidence to suggest that the localized observation was significant. Domain experts agree that the finding is striking, strengthening its candidacy for web lab experimentation. Further from the analysis of *Calcium Regulation*, the user noted a pathway overlap with *Purine metabolism*. This provides another extension point to analysis, which could manifest as a reverse analysis from local observation to global view of the relative involvement of *Purine metabolism*. *Smooth muscle contraction* is another such extension point.



Description	N/n	S1	S2
Purine metabolism	262/ 2	0.76	33.33
Calcium_regulation_in...	387/ 3	0.78	50.0

Description	N/n	S1	S2
Purine metabolism	262/ 2	0.76	50.0
Calcium_regulation_in...	387/ 4	1.03	100.0

Fig. 7. Workflow for the exploration of a serendipitous pathway discovery



Association Considerations

	K-Means	XMAS
Expression Profile	X	X
Expression Characteristic	🟥	X
Pathway Membership	🟥	X
Differential Expression	Exclusively	Optional

Fig. 8. Comparative assessment of association quality: The left figure compares the two sets of probes associated by k-means and XMAS respectively; the upper right figure identifies the common probe shared by these two sets; and the lower right table compares the factors under consideration by K-means and XMAS

Traditional analysis and analysis within XMAS are difficult to compare directly because of the differing emphasis on interaction and exploratory analysis, and global statistical/algorithmic analysis respectively. The outputs from both traditional and experiential approaches are comparable, however.

K-means analysis from 0, for example, associated the DEG from our set of four (201667_at) with 9 other DEGs, based on expression alone. This set serves as a direct comparison for the set of four which emerged from the previously described analysis. Despite having more probes, and more DEGs, the literature hits for our set far outweigh the expression (only) based association of k-means. See Fig. 8 for details.

4 Conclusions

This paper has presented XMAS, a web application developed with a new design philosophy to foster increased human-computer synergy. Various interoperable operators have been presented which combine with visualizations and HCI to compose an exploratory, interactive analysis system. Detailed use cases and comparisons made between XMAS and well established microarray analysis methods present evidence to prove the ability of this new approach to dramatically enhance the users experience during analysis. This materializes in the form of new, more complete hypothesis generation.

References

[1] Butte, A.: The use and analysis of microarray data. Nat. Rev. Drug. Discovery 1(12), 951–960 (2002)

[2] Microarrays: Chipping away at the mysteries of science and medicine, NCBI Just the Facts Series, `http://www.ncbi.nlm.nih.gov/About/primer/microarrays.html`

[3] Faramarz, V.: Pattern Recognition Techniques in Microarray Data Analysis: A Survey. Bioinformatics and Medical Informatics (980), 41–64 (2002)

[4] Aas, K.: Microarray Data Mining: A Survey. NR Note SAMBA/02/01 (2001)

[5] Mukherjee J.: ICB, http://chagall.med.cornell.edu/I2MT/MA-tools.pdf

[6] Ingenuity Systems, http://www.ingenuity.com

[7] Draghici, S., et al.: Onto-Tools, the toolkit of the modern biologist: Onto-Express, Onto-Compare, Onto-Design and Onto-Translate. NAR 31(13), 3775–3781 (2003)

[8] Salomonis, et al.: GenMAPP 2: new features and resources for pathway analysis. BMC Bioinformatics 8, 217 (2007)

[9] Gentleman, R.C.: Bioconductor: open software development for computational biology and bioinformatics. Genome Biology (2004)

[10] Project R, http://www.r-project.org/

[11] Tusher, V., et al.: Significance analysis of microarrays applied to the ionizing radiation response. PNAS 98, 5116–5121 (2001)

[12] Tibshirani, R., et al.: Diagnosis of multiple cancer types by shrunken centroids of gene expression. PNAS 99, 6567–6572 (2002)

[13] Liang, Y., Kelemen, A.: Associating phenotypes with molecular events: recent statistical advances and challenges underpinning microarray experiments. Functional & Integrative Genomics 6(1) (January 2006)

[14] Singh, R., Jain, R.: From Information-Centric to Experiential Environments. In: Goldin, D., Smolka, S., Wegner, P. (eds.) Interactive Computation: The New Paradigm, pp. 323–351. Springer, Heidelberg (2006)

[15] Jain, R.: Experiential computing. Commun. ACM 46(7), 48–55 (2003)

[16] Kanehisa, M., et al.: KEGG for linking genomes to life and the environment. Nucleic Acids Res. 36, D480–D484 (2008)

[17] OmniViz, http://www.biowisdom.com/content/omniviz

[18] Lönstedt, I., Speed, T.P.: Replicated microarray data. Stat. Sin. 12, 31–46 (2002)

[19] The Gene Ontology Consortium. Gene Ontology: tool for the unification of biology. Nature Genet. 25, 25–29 (2000)

[20] Winn, V., et al.: Gene Expression Profiling of the Human Maternal-Fetal Interface Reveals Dramatic Changes between Midgestation and Term. Endocrinology 148(3)

Clustering in a Fixed Manifold to Detect Groups of Genes with Similar Expression Patterns

N.M. Phuong and H.D. Tuan

School of Electrical Engineering and Telecommunications
University of New South Wales, Sydney, NSW, 2052, Australia
n.m.phuong@student.unsw.edu.au, h.d.tuan@unsw.edu.au

Abstract. Clustering genes into groups that exhibit similar expression patterns is one of the most fundamental issues in microarray data analysis. In this paper, we present a normalized Expectation-Maximization (EM) approach for the problem of gene-based clustering. The normalized EM clustering also follows the framework of generative clustering models but for the data in a fixed manifold. We illustrate the effectiveness of the normalized EM on two real microarray data sets by comparing its clustering results with the ones produced by other related clustering algorithms. It is shown that the normalized EM performs better than the related algorithms in term of clustering outcomes.

Keywords: clustering, microarray data, normalized EM, manifold.

1 Introduction

Microarray technology allows ones to simultaneously monitor expression measurements of thousands of genes, across various conditions or over time [1], [2], [3]. To explore a vast amount of data generated from microarray experiments, numerous data mining techniques have been proposed to extract insightful biological data-based knowledge. Clustering is one of the basic exploratory tools for microarray data analysis. A wide range of clustering methods have been proposed in gene expression community including hierarchical clustering [4], [5], [6]; self-organizing maps (SOM) [7]; k-means and its variants [8], [9], [10]; graph-based methods [11], [12]; and mixture model-based clustering [13], [14], [15].

One of the most important aspects in clustering is the metric utilized to gauge the similarity between data points and their cluster representatives. It is well-known that similar gene expression patterns often show co-linear stochastic relationship, especially on standardized data sets. This stochastic relationship is suitably evaluated by the cosine similarity among data vectors. Indeed, the similarity measure has been seen to be applied for clustering [16], [17]. In [16] Dhillon and Modha performed text clustering using spherical k-means. In [17] Banerjee et al proposed a method to estimate the concentration parameter of von Mises-Fisher distributions and applied their clustering algorithms for yeast cell cycle gene expression data. They were ,however, more concerned with the estimation of stochastic model parameters than the applicability of the clustering on a hypersphere for extracting useful knowledge from gene expression profiles.

In this paper, we propose a normalized EM algorithm for clustering gene expression data, in which data points are already projected onto a hypersphere. The proposed approach also follows the mixture model-based clustering framework but data points are assumed to be generated by a mixture of exponential distributions in a fixed manifold, which is the surface of a hypersphere. The normalized EM is able to work stable with high dimensional microarray data sets.

In short, our main contributions in this work are the following: (i) the normalized EM algorithm is introduced for the problem of clustering genes using microarray data; (ii) the viability of the proposed normalized EM is demonstrated by comparing its clustering performance with that of spherical k-means [16] and Gaussian parsimonious clusterings [18].

The remainder of this paper is as follows. Section 2 presents the statistical model of normalized EM and derivations of the algorithm. In section 3, the performance of the proposed approach is examined with the demonstration through two real microarray data sets. Finally, section 4 reviews what has been done in this work and briefly discusses directions of further research.

For the ease of presentation, some conventions of notation used in this paper are provided: n is the number of data points or genes to be clustered; p is the dimension of data points or the number of samples; $\mathcal{X} = \{x_i\}_{i=1}^{n}$ is the set of all data points; K is the number of clusters in a data set; $\{\mathcal{X}_h\}_{h=1}^{K}$ is a K-cluster partition of the data; $\langle . \rangle$ is the inner product of two vectors; $\|.\|$ is the Euclidean norm.

2 Statistical Model

A typical microarray data set is given as a matrix $G_{p\times n}$, where the entry (i, j) of the matrix represents the expression level of gene j in the i^{th} experiment. In other words, $G = [x_1, x_2, ..., x_n]$ where $x_j \in \mathbb{R}^p$ is the expression profile of the j^{th} gene in the microarray. Usually the number of genes n is much larger than the number of experiments p ($n \gg p$). Our primary aim is to detect groups of genes exhibiting similar expression patterns. Specifically, we have to classify the set of data points or genes into K groups $\mathcal{X}_1, \mathcal{X}_2, ..., \mathcal{X}_K$ such that genes within each cluster are highly correlated whereas expression patterns of genes between clusters are as much different as possible.

An important point to note here is that Euclidian distance is not capable of capturing the co-linear stochastic relationship between the original gene expression profiles. To overcome the limitation, data are projected onto the surface of a hypersphere. And as can be seen, after the data projection cosine similarity between any two data vectors still remains unchanged and can be trivially inferred from their Euclidian distance in the new manifold.

Spherical k-means is one of the heuristic clustering algorithms applied for spherical data, where at each iteration any data point is assigned to one of the clusters with probability one. This hard clustering procedure may not be robust against the complexity of microarray data, which are inherently dense and noisy.

We now introduce a new normalized EM soft clustering approach for both gene expression data. First, data points are normalized so that they lie on a hypersphere and then clustering of the data is performed on this hypersphere only. The statistical model for the normalized EM clustering is described in detail as follows:

First, each gene expression profile x_i is normalized so that they belong to a fixed manifold $S_\mu = \{x : \|x\|^2 = \mu, x \in \mathbb{R}^p\}$ for some $\mu > 0$. In other words, the data points are processed by

$$x_i \to \sqrt{\mu}\frac{x_i}{\|x_i\|},\ i = 1, 2, ..., n \tag{1}$$

Then these normalized x_i's are treated as drawn sampled outcomes of a mixture of K exponential distributions

$$p(x|\Theta) = \gamma_\mu \sum_{h=1}^{K} \pi_h e^{-\|x-\mu_h\|^2} \tag{2}$$

where $\Theta = (\pi_1, \mu_1, ..., \pi_k, \mu_k)$, in which the π_h, μ_h as mixing proportions and directional mean vectors respectively,

$$\sum_{h=1}^{K} \pi_h = 1, \pi_h \geq 0, \|\mu_h\|^2 = \mu,\ h = 1, 2, ..., K \tag{3}$$

and γ_μ is the normalizing constant

$$\gamma_\mu = 1/\int_{x \in S_\mu} e^{-\|x-\mu_h\|^2} dx. \tag{4}$$

Assuming that the data vectors are independent and identically distributed with distribution p. Then the data likelihood function is

$$\mathcal{L}(\Theta|\mathcal{X}) = p(\mathcal{X}|\Theta) = \prod_{i=1}^{n} p(x_i|\Theta) = \prod_{i=1}^{n} (\gamma_\mu \sum_{h=1}^{K} \pi_h e^{-\|x-\mu_h\|^2}). \tag{5}$$

The maximum likelihood problem is thus

$$\max_{\Theta} \{\mathcal{L}(\Theta|\mathcal{X}) : (3)\}. \tag{6}$$

However, maximizing the likelihood function (6) is very difficult and we relax it by maximizing the expectation of the marginal log-likelihood function [19].

Given current estimates $\Theta^{(\ell)}$ at the ℓ^{th} iteration ($\ell \geq 0$) of the EM iterative procedure, for each $h = 1, 2, ..., K$, the posterior probability $p(h|x_i, \Theta^{(\ell)})$ that x_i is generated by the h^{th} component of the mixture density is defined by

$$\begin{aligned} p(h|x_i, \Theta^{(\ell)}) &= \frac{p(h|\Theta^{(\ell)}) p(x_i|h, \Theta^{(\ell)})}{p(x_i|\Theta^{(\ell)})} \\ &= \frac{\pi_h^{(\ell)} e^{2\langle x_i, \mu_h^{(\ell)} \rangle}}{\sum_{h'=1}^{K} \pi_{h'}^{(\ell)} e^{2\langle x_i, \mu_{h'}^{(\ell)} \rangle}}. \end{aligned} \tag{7}$$

The expectation of the marginal log-likelihood function for the observed data over the given posterior distribution is

$$
\begin{aligned}
&E[\sum_{i=1}^{n} \log(\gamma_\mu \pi_h e^{-\|x_i-\mu_h\|^2})] \\
&= \sum_{i=1}^{n} E[\log(\gamma_\mu \pi_h e^{-\|x_i-\mu_h\|^2})] \\
&= \sum_{i=1}^{n}\sum_{h=1}^{K}[\log(\gamma_\mu \pi_h e^{-\|x_i-\mu_h\|^2})]p(h|x_i,\Theta^{(\ell)}) \\
&= \sum_{i=1}^{n}\sum_{h=1}^{K}(\log \pi_h - \|x_i-\mu_h\|^2)p(h|x_i,\Theta^{(\ell)}) + n\log\gamma_\mu \\
&= \sum_{i=1}^{n}\sum_{h=1}^{K}(\log \pi_h - 2\mu + 2\langle x_i,\mu_h\rangle)p(h|x_i,\Theta^{(\ell)}) + n\log\gamma_\mu \\
&= \sum_{h=1}^{K}\sum_{i=1}^{n}(\log \pi_h + 2\langle x_i,\mu_h\rangle)p(h|x_i,\Theta^{(\ell)}) - 2nK\mu + n\log\gamma_\mu . \qquad (8)
\end{aligned}
$$

The maximization (6) is relaxed by maximizing expectation of the marginal log-likelihood function

$$
\begin{aligned}
&\max_{\Theta}\,\{\sum_{h=1}^{K}\sum_{i=1}^{n}(\log \pi_h + 2\langle x_i,\mu_h\rangle)p(h|x_i,\Theta^{(\ell)}) - 2nK\mu + n\log\gamma_\mu : (3)\} \\
&= \max_{\Theta}\{\sum_{h=1}^{K}\sum_{i=1}^{n}(\log \pi_h)p(h|x_i,\Theta^{(\ell)}) + \\
&\quad +2\sum_{h=1}^{K}\sum_{i=1}^{n}\langle x_i,\mu_h\rangle p(h|x_i,\Theta^{(\ell)}) : (3)\} - 2nK\mu + n\log\gamma_\mu \\
&= \max_{\{\pi_h\}_{h=1}^{K}}\{\sum_{h=1}^{K}\sum_{i=1}^{n}(\log \pi_h)p(h|x_i,\Theta^{(\ell)}) : \sum_{h=1}^{K}\pi_h = 1, \pi_h \geq 0, h=1,2,..,K\} + \\
&\quad +2\sum_{h=1}^{K}\max_{\mu_h}\{\sum_{i=1}^{n}\langle x_i,\mu_h\rangle p(h|x_i,\Theta^{(\ell)}) : \|\mu_h\|^2 = \mu\} - 2nK\mu + n\log\gamma_\mu \qquad (9)
\end{aligned}
$$

To find $\max_{\{\pi_h\}_{h=1}^{K}}\{\sum_{h=1}^{K}\sum_{i=1}^{n}(\log \pi_h)p(h|x_i,\Theta^{(\ell)}) : \pi_h \geq 0, \sum_{h=1}^{K}\pi_h = 1\}$, we introduce Lagrange multiplier λ with the constraint $\sum_{h=1}^{K}\pi_h = 1$ and form the following Lagrangian

$$
L(\{\pi_h\}_{h=1}^{K}) = \sum_{h=1}^{K}\sum_{i=1}^{n}(\log \pi_h)p(h|x_i,\Theta^{(\ell)}) - \lambda(\sum_{h=1}^{K}\pi_h - 1). \qquad (10)
$$

Take the partial derivatives of (10) with respect to each π_h and λ, then set them to zero, we obtain

$$\sum_{i=1}^{n} \frac{1}{\pi_h} p(h|x_i, \Theta^{(\ell)}) - \lambda = 0,\ h = 1, 2, ..., K \tag{11}$$

$$\sum_{h=1}^{K} \pi_h = 1 \qquad . \tag{12}$$

From equations (11) and (12), we get

$$\lambda = n$$
$$\pi_h^{(\ell+1)} = \tfrac{1}{n} \sum_{i=1}^{n} p(h|x_i, \Theta^{(\ell)}),\ h = 1, 2, ..., K\ . \tag{13}$$

To find $\max\limits_{\mu_h} \sum\limits_{i=1}^{n} \langle x_i, \mu_h \rangle p(h|x_i, \Theta^{(\ell)})$ subject to $\|\mu_h\|^2 = \mu$, we introduce the Lagrange multiplier λ_h and the Lagrangian here is given by

$$L(\mu_h) = \sum_{i=1}^{n} \langle x_i, \mu_h \rangle p(h|x_i, \Theta^{(\ell)}) - \lambda_h(\|\mu_h\|^2 - \mu). \tag{14}$$

Similarly as above, take the partial derivatives of (14) with respect to $\{\mu_h, \lambda_h\}$ and set them to zero, we obtain

$$\sum_{i=1}^{n} x_i p(h|x_i, \Theta^{(\ell)}) - 2\lambda_h \mu_h = 0 \tag{15}$$

$$\|\mu_h\|^2 = \mu \tag{16}$$

Solving (15) and (16), we get

$$\lambda_h = \tfrac{1}{2\sqrt{\mu}} \|\sum_{i=1}^{n} x_i p(h|x_i, \Theta^{(\ell)})\|$$

$$\mu_h^{(\ell+1)} = \frac{\sqrt{\mu}[\sum\limits_{i=1}^{n} x_i p(h|x_i, \Theta^{(\ell)})]}{\|\sum\limits_{i=1}^{n} x_i p(h|x_i, \Theta^{(\ell)})\|} \tag{17}$$

The EM iterative procedure of the normalized EM is as follows:

$$\pi_h^{(\ell+1)} = \frac{1}{n}\sum_{i=1}^{n} p(h|x_i, \Theta^{(\ell)})$$
$$= \frac{1}{n}\frac{\sum_{i=1}^{n} \pi_h^{(\ell)} e^{2\langle x_i, \mu_h^{(\ell)}\rangle}}{\sum_{h'=1}^{K} \pi_{h'} e^{2\langle x_i, \mu_{h'}^{(\ell)}\rangle}} \tag{18}$$

$$\nu_h^{(\ell+1)} = \sum_{i=1}^{n} x_i p(h|x_i, \Theta^{(\ell)})$$
$$\mu_h^{(\ell+1)} = \frac{\sqrt{\mu}\nu_h^{(\ell+1)}}{\|\nu_h^{(\ell+1)}\|}. \tag{19}$$

The optimal parameter estimates Θ_{opt} are obtained when the difference between two observed data log-likelihoods corresponding to two successive iterations is less than a given tolerance threshold. Finally, each data point is assigned to the component with the maximum estimated posterior probability, i.e. a data point x_i is assigned to component h or cluster $\mathcal{X}_h$ if $h = \arg\max_{h'} p(h'|x_i, \Theta_{opt})$.

3 Results

The utility of the normalized EM clustering approach is demonstrated on two microarray data sets: (1) yeast cell cycle data with the five-phase criterion [15]; (2) yeast cell cycle data of regulated genes [20]. The clusterings of the normalized EM on these gene expression data sets are assessed with different values of μ and the obtained results are compared with those produced by spherical k-means for both data sets. The normalized EM is also compared with Gaussian parsimonious clustering models on the first data set. For the second data set, due to the lack of external criterion and the difference in similarity measures used in the normalized EM and Gaussian parsimonious clustering models, we just make comparison of the normalized EM with spherical k-means. Note that the analysis of the normalized EM is only provided for the values of μ in the range from 0 up to 350 as with the bigger values of μ, the iterative procedure of the normalized EM involves the difficulty of very large exponential computations.

Yeast Cell Cycle Data with the Five-Phase Criterion

This data set was created and used by Yeung et al [15]. It consists of 384 genes across 17 experiments and is supposed to include five clusters corresponding to five phases during the mitotic cell cycle: Early G1, Late G1, S, G2 and M. It should be noted that beside the original raw data set, the standardized data set derived from the original was also analyzed. Both the two data sets have been made publicly available by Yeung et al at http://faculty.washington.edu/kayee/cluster/. For the ease of comparing clusterings,

Table 1. Clustering results of the normalized EM on the raw data set (10 runs were performed for each value of μ)

μ	20	25	30	40	80	120	180	240	300	350
ARI	0.49	0.49	0.5	0.49	0.37	0.37	0.46	0.47	0.36	0.37
	0.49	0.49	0.37	0.49	0.49	0.37	0.37	0.37	0.48	0.37
	0.49	0.49	0.48	0.48	0.48	0.36	0.37	0.49	0.37	0.37
	0.49	0.49	0.38	0.48	0.48	0.48	0.48	0.36	0.37	0.47
	0.49	0.49	0.38	0.49	0.49	0.47	0.47	0.37	0.37	0.49
	0.49	0.5	0.49	0.49	0.48	0.48	0.49	0.37	0.48	0.48
	0.47	0.49	0.5	0.48	0.47	0.47	0.48	0.48	0.47	0.46
	0.50	0.49	0.5	0.49	0.47	0.37	0.37	0.37	0.49	0.37
	0.49	0.48	0.5	0.36	0.44	0.37	0.37	0.48	0.47	0.47
	0.49	0.49	0.48	0.48	0.37	0.49	0.37	0.37	0.48	0.48
Average ARI	0.49	0.49	0.46	0.47	0.45	0.42	0.42	0.41	0.43	0.43

we made use of adjusted rand index (ARI) [21], which is an information criterion to evaluate the degree of agreement between two partitions, one is the real clustering and the other is derived from given class labels. The higher the value of ARI, the better the predictive ability of a clustering algorithm.

Table 1 and Table 2 show the values of ARI produced by the normalized EM and spherical k-means respectively on the raw data set. Similar results were also obtained on the standardized data set for both the normalized EM and spherical k-means. The normalized EM worked well in the range from 20 to 350 and achieved the highest values of ARI when μ was in the range from 20 to 25. As can be seen, spherical k-means worked quite comparable to the normalized EM in term of clustering results. However, when the normalized EM worked best, e.g. μ was in the range from 20 to 25, it consistently produced higher average values of ARI compared to spherical k-means.

Table 2. Clustering results of spherical k-means on the raw data set (20 runs were performed)

ARI	0.37	0.48	0.37	0.48	0.46	0.46	0.37	0.37	0.45	0.38
	0.49	0.37	0.37	0.37	0.47	0.37	0.37	0.37	0.47	0.37
Average ARI	0.41									

In [15], Yeung et al utilized five typical Gaussian parsimonious models for clustering this yeast cell cycle data: EI(EII), VI(VII), VVV, diagonal and EEE and they showed that some of these models worked comparable to CAST, a leading heuristic clustering algorithm. The cluster quality of these Gaussian parsimonious clustering models on this data with the number of clusters $K = 5$ was again examined using Mclust package [22]. For diagonal models, EEI was taken to analyze. Since the normalized EM was run using random initializations, to be fair we also performed Gaussian parsimonious clusterings with random initializations as well. It should be noted that in [15], the authors made use of mixture model-based hierarchical clusterings to initialize the iterative procedure of the EM algorithm. Table 3 and Table 4 show the ARI values produced by Gaussian parsimonious clustering models on the raw data set and standardized data set

Table 3. Clustering results of Gaussian parsimonious clustering models on the raw data set (10 runs were performed for each clustering model)

	EII	VII	VVV	EEI	EEE
ARI	0.0114	0.0111	0.011	0.011	0.011
	0.0048	0.0048	0.0685	0.0048	0.0209
	0.0289	0.0256	0.0349	0.0316	0.0175
	0.0022	0.0077	0.0088	0.0083	0.0088
	0.0236	0.0219	0.0029	0.0121	0.0118
	0.0112	0.0054	0.0111	0.0111	0.0089
	0.0048	0.0689	0.0048	0.0506	0.0053
	0.0157	0.0349	0.0141	0.0252	0.0491
	0.0029	0.0083	0.008	0.0088	0.0027
	0.0001	0.0051	0.0026	0.0427	0.0104
Average ARI	0.0106	0.0194	0.0167	0.0206	0.0146

Table 4. Clustering results of Gaussian parsimonious clustering models on the standardized data set (10 runs were performed for each clustering model)

	EII	VII	VVV	EEI	EEE
ARI	0.5	0.45	0.21	0.47	0.45
	0.5	0.37	0.17	0.47	0.45
	0.37	0.48	0.32	0.49	0.45
	0.37	0.48	0.31	0.49	0.42
	0.46	0.48	0.29	0.49	0.48
	0.5	0.44	0.29	0.37	0.48
	0.37	0.5	0.23	0.49	0.46
	0.37	0.43	0.25	0.37	0.47
	0.5	0.47	0.29	0.37	0.43
	0.46	0.45	0.26	0.44	0.41
Average ARI	0.44	0.45	0.26	0.45	0.45

respectively. As can be seen, on the raw data set these models failed to discover the inherent cluster structure of the data as very low ARI values were achieved. On the standardized data set, except VVV model the other four models produced quite high ARI values. However, when the normalized EM worked best, e.g. μ was in the range from 20 to 25, it consistently produced higher average values of ARI compared to all the five Gaussian parsimonious clustering models, see Table 1 and Table 4 for verification. We mention again that on the standardized data set, the normalized EM produced similar results as shown in Table 1 for the raw data set.

Yeast cell cycle data of regulated genes

This data set consists of 800 genes across 77 experiments. These 800 genes were selected to meet an objective minimum criterion for cell cycle regulation [20]. For this data, as we do not know the number of underlying clusters that the data should have, the normalized EM and spherical k-means were run by trying various choices of the number of clusters K. In order to evaluate clusterings when class labels are unknown, internal indices have been used widely to measure clustering quality [17], [23], [11]. In [17], the homogeneity for spherical data of a clustering is defined to be

$$H_{avg} = \frac{1}{n} \sum_{h=1}^{K} \sum_{x_i \in \mathcal{X}_h} \frac{x_i^T \mu_h}{\|x_i\| \|\mu_h\|} \tag{20}$$

which is the sum of cosine similarities between all data points and their own cluster representatives. On the other hand, the inter-cluster separation is taken note as

$$S_{avg} = \frac{1}{\sum_{i \neq j} |\mathcal{X}_i||\mathcal{X}_j|} \sum_{i \neq j} |\mathcal{X}_i||\mathcal{X}_j| \frac{\mu_i^T \mu_j}{\|\mu_i\| \|\mu_h\|}. \tag{21}$$

The bigger the value of H_{avg} and the smaller the value of S_{avg}, the higher the predictive ability of a clustering algorithm. We made use of these figures of merit to compare clusterings produced by the normalized EM and spherical k-means. For each value of μ and each choice of K, we ran the normalized algorithm 10 times and took the average of H_{avg}, S_{avg} values of these clustering results. Similarly, we ran spherical k-means 10 times for each choice of K and the average values of H_{avg}, S_{avg} for these clusterings were computed. These summary statistics are shown in Figure 1 and Figure 2 to

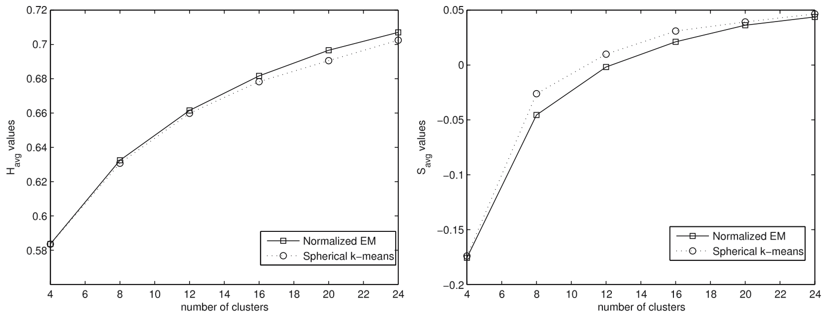


Fig. 1. Comparisons of cluster quality produced by the normalized EM ($\mu = 50$) and spherical k-means

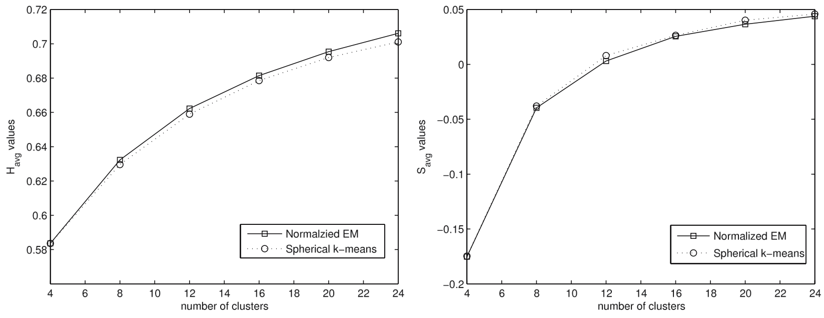


Fig. 2. Comparisons of cluster quality produced by the normalized EM ($\mu = 100$) and spherical k-means

compare the predictive ability of the two clustering algorithms. Note that for limited space, only the comparisons with $\mu = 50$ and $\mu = 100$ are shown. It can be seen that compared to spherical k-means, the normalized EM achieved higher values of H_{avg} and smaller values of S_{avg} even with different values of μ as well as various choices of K. This demonstrates that the normalized EM produced a bit higher cluster quality than spherical k-means. In fact, we took many other values of μ in the range from 40 to 350 and similar conclusions were taken out.

4 Conclusions and Future Works

We have introduced and described a statistical model for clustering data in a fixed manifold in order to identify groups of genes with similar expression patterns using microarray data. Additionally, the utility of the normalized EM has been confirmed by comparing its clustering results with the ones produced by spherical k-means and Gaussian parsimonious clusterings.

It is of interest and left for future work to show that the normalized EM is capable of leaving out noisy genes and only producing meaningful clusters, in which genes within each group are highly correlated. Besides, it should be noted that the normalized EM is also able to work very stable with the problem of clustering samples as well but these results are not included here.

References

1. Lockhart, D., Dong, H., Byrne, M., Follettie, M., Gallo, M., Chee, M., Mittmann, M., Wang, C., Kobayashi, M., Horton, H., Brown, P.: Expression monitoring by hybridization to high density oligonucleotide arrays. Nature Biotechnology 14, 1675–1680 (1996)
2. Schena, M., Shalon, D., Davis, R., Brown, P.: Quantitative monitoring of gene expression patterns with a DNA microarray. Science 210, 467–470 (1995)
3. Shalon, D., Smith, S., Brown, P.: A DNA microarray system for analyzing complex DNA samples using two-color fluorescent probe hybridization. Genome Research 6, 639–645 (1996)
4. Eisen, M.B., Spellman, P.T., Brown, P.O., Botstein, D.: Cluster analysis and display of genome-wide expression patterns. In: Proceedings of the National Academy of Sciences of the United States of America (1998)
5. Iyer, V., Eisen, M., Ross, D., Schuler, G., Moore, T., Lee, J., Trent, J., Staudt, L., Hudson, J., Boguski, M., Lashkari, D., Shalon, D., Botstein, D., Brown, P.: The transcriptional program in response of human fibroblasts to serum. Science 283, 83–87 (1999)
6. Wen, X., Fuhrman, S., Michaels, G.S., Carr, D.B., Smith, S., Barker, J.L., Somogyi, R.: Large-scale temporal gene expression mapping of central nervous system development. The national academy of sciences, 334–339 (January 1998)
7. Tamayo, P., Slonim, D., Mesirov, J., Zhu, Q., Kitareewan, S., Dmitrovsky, E., Lander, E., Golub, T.: Interpreting patterns of gene expression with self-organizing maps: methods and application to hematopoietic differentiation. In: Proceedings of the National Academy of Sciences of the United States of America, pp. 2097–2912 (1999)
8. Smet, F., Mathys, J., Marchal, K., Thijs, G., Moor, B., Moreau, Y.: Adaptive quality-based clustering of gene expresion profiles. Bioinformatics 18(5), 735–746 (2002)

9. Tavazoie, S., Hughes, J., Campbell, M., Cho, R., Church, G.: Systematic determination of genetic network architechture. Nature Genetics 22, 281–285 (1999)
10. Tseng, G.: Penalized and weighted k-means for clustering with scattered objects and prior information in high-throughput biological data. Bioinformatics 23(17), 2247–2255 (2007)
11. Sharan, R., Shamir, R.: Click: A clustering algorithm with applications to gene expression analysis. In: Proceedings of the 8th International Conference on Intelligent Systems for Molecular Biology (ISMB), pp. 307–316 (2000)
12. Xu, Y., Olman, V., Xu, D.: Clustering gene expression data using a graph-theoretic approach: an application of minimum spanning trees. Bioinformatics 17(4), 309–318 (2001)
13. Ghosh, D., Chinnaiyan, A.M.: Mixture modelling of gene expression from microarray experiments. Bioinformatics 18(2), 275–286 (2002)
14. McLachlan, G., Bean, R., Peel, D.: A mixture model-based approach to the clustering of microarray expression data. Bioinformatics 18(3), 413–422 (2002)
15. Yeung, K.Y., Fraley, C., Murua, A., Raftery, A.E., Ruzzo, W.L.: Model-based clustering and data transformations for gene expression data. Bioinformatics 17, 977–987 (2001)
16. Dhillon, I., Modha, D.: Concept decompositions for large sparse text data using clustering. Machine Learning 42(1), 143–175 (2001)
17. Banerjee, A., Dhillon, I., Ghosh, J., Sra, S.: Clustering on the unit hypersphere using von Mises-Fisher distributions. Journal of Machine Learning Research 6, 1345–1382 (2005)
18. Celeux, G., Govaert, G.: Gaussian parsimonious clustering models. Pattern Recognition 28(5), 781–793 (1995)
19. Dempster, A., Laird, N., Rubin, D.: Maximum likelihood for incomplete data via the EM algorithm. Journal of Royal Stastistical Society 29, 1–38 (1977)
20. Spellman, P.T., Sherlock, G., Zhang, M.Q., Iyer, V.R., Anders, K., Eisen, M., Brown, P., Bostein, D., Futcher, B.: Comprehensive identification of cell cycle-regulated genes of the yeast saccharomyces cerevisiae by microarray hybridization. Molecular biology of the cell 9, 3273–3297 (1998)
21. Hubert, L., Arabie, P.: Comparing partitions. Journal of Classification 2, 193–218 (1985)
22. Fraley, C., Raftery, A.: Mclust: software for model based cluster analysis. Journal of Classification 16, 297–306 (1999)
23. Bezdek, J.C., Pal, N.R.: Some new indexes of cluster validity. IEEE Trans. on Systems, Man and Cybertics 28, 301–315 (1998)

Families of FPGA-Based Accelerators for BLAST Algorithm with Multi-seeds Detection and Parallel Extension

Fei Xia, Yong Dou, and Jiaqing Xu

Department of Computer Science, National University of Defence Technology, Changsha, P.R. China, 410073
{xcyphoenix,yong_dou,xjqanswer}@hotmail.com

Abstract. As one of the most widely used bio-sequence searching tools, BLAST adopts index-based approach to detect the matches between two substrings by looking up a large table and processing one match per query. In this paper, we propose a systolic array approach to detect string matches without using looking up tables. The pipelining systolic array is implemented as a multi-seeds detection and parallel extension pipeline engine to accelerate the first two stages of NCBI BLAST family algorithms. Different from the index-based approach, our implementation consumes little memory resources and eliminates redundant string extensions by merging multiple adjoin seeds into a valid seed. Our FPGA implementation achieves superior performance results in both of processing element number and clock frequency over related works in the area of FPGA BLAST accelerators. The experimental results also show the speedup can reach about 17, 48, 14, 71 and 10 compared to the NCBI BLASTp, TBLASTn, BLASTx, TBLASTx and BLASTn programs for 3072-residue queries on Intel P4 CPU, respectively. Furthermore, the idea of multi-seeds detection also can be adopted in other seed-based heuristic searching applications.

1 Introduction

The comparison of DNA or protein sequences has become a fundamental task of modern molecular biology. BLAST (Basic Local Alignment Search Tool)[1] as one of the most important tools has been designed to run on commodity PC clusters at present, such as [2],[3],[4],[5],[6] to search for sequence similarity in genomic databases. With the exponential growth of the bio-sequence databases, such as the NCBI (National Center for Biotechnology Information) GenBank[7], which has doubled in size every 12∼16 months for the last decade and now stands at over 56 billion characters, the computational requirements for sequence comparisons have far exceeded the computing capability.

General-purpose microprocessors typically provide very limited bit-level parallelism. However, sequence comparison algorithms exhibit a much higher degree of bit-level data parallelism, typically hundreds of bit-level operations can be performed in parallel. Therefore, many researchers keen on implementing BLAST algorithms in hardware to avoid the low efficiency in general-purpose microprocessors. Recently, FPGA chips have emerged as one promising application accelerator, using a combination of FPGAs and general-purpose CPUs to accelerate BLAST algorithm attracts much more attention. A number of parallel architectures have been proposed, such as Mercury BLASTn[8],[9],[10], Tree-BLAST[11], Mercury BLASTp[12], RC-BLAST[13], FPGA/FLASH Accelerator[14], Multi-engine BLASTn Accelerator[15],[16] and many commercialized system, BEE2[17], CLC Cube[18], Mitrion[19] and DeCypher[20] et al. have been built.

Most of the current implementations adopt the index-based searching approach, which builds all kinds of tables to record the position of each *word* in query sequence, then drives the *words* (or named *w-mers*) in database flowing through the accelerator one by one and looks up the table to find the *seeds*. However, this method typically suffers two drawbacks. Firstly, only one word can be searched per cycle (meaning at most one seed can be detected per cycle), with the limitation on memory port number, no matter whether the table is stored in internal or external memory. Second, the storage and access overhead of lookup table become the resource bottleneck.

Specifically, Mercury BLASTn[9] and Mitrion[19] implement a pre-filter using hashing, then check *words* in database against a hash table constructed from the query one by one. Hash table is stored in an external SRAM attached to FPGA, since the internal block RAMs are too limited in size to hold the tables for large query sequence. The accessing delay to external SRAM incurs long pipeline cycle time. RC-BLAST[13] and BEE2[17] implement the word-finding stage by using query index. Each word from subject sequence is then used as an index to lookup the table in order. Because of the limitation of on-chip memory size, the design in RC-BLAST assumes that no word in query sequence is repeated more than three times. Obviously, the assumption is unreasonable. Compared with other designs, FPGA/FLASH adopted a novel approach, the database is also formatted as an index structure. Each word is associated with its position in the sequence and its neighboring environment. This information allows short un-gapped alignments to be immediately computed, avoiding millions of random accesses to the database. Unfortunately, the size of the database index has to be very large. As an example, storing a 40 amino acid substring environment leads to a 150 GB index for the Human genome. This is 50 times more than the raw data[14]. The storage cost will be intolerable with the steep growth of database. To improve searching efficiency, Multi-engines BLASTn[16] fitted 64 identical computing machines in single chip to compare the query with 64 subject sequences in database concurrently and Mercury BLASTp[12] implemented a two-seed generator for accelerating the first stage of BLASTp. Unfortunately, these approaches are still based on the query index essentially.

Besides the index-based searching approach, there exists another searching strategy, which uses systolic array without lookup tables. D.Hoang et al. [21],[22] implemented the Needleman-Wunsch and dynamic programming algorithms using systolic array implementation on SPLASH 2. Using JBits S.Guccione et al.[23] implements the Smith-Waterman algorithm. The most recent implementations were the Hyper Customized Processors in Nanyang Tech University[24] and FPGA-Based Accelerators by Tom Van Court et al[25]. It is a natural approach to use systolic array to mapping dynamic programming algorithms on FPGAs. But it is rare to use systolic array for mapping the BLAST algorithm, only Tree-BLAST[11] can be found.

At present, most of the seed-based solutions test the words from database stream in a serial mode, one match per cycle. The searching efficiency can be improved if hardware can detect multiple "*seeds*" concurrently and extend them in parallel. In this paper we present a *Multi-seeds Detection and Parallel Extension Engine* to accelerate BLAST family algorithms. Our design is based on systolic array rather than the static lookup table. It lessens the storage requirement to on-chip memory because all positions of match points can be calculated dynamically at seeds detection pipelines. The multi-seeds detection has three advantages: Firstly, it improves the searching capability in word-matching stage, which can execute up to 3072 matches/cycle and report all the match points contemporarily with the help of 3072 PEs. Secondly, all the reported seeds at a time are located in identical diagonal, which is convenient for filtering some invalid seeds. Finally, the mechanism of multiple seeds detection supplies enough seeds to reduce the empty time in the extension stage. As a result the extension efficiency can be improved. Our implementation also uses merging seeds strategy to reduce unnecessary extension. We fit our design on Altera FPGA chips EP2S130C5 to accelerate the first two stages of BLAST family algorithms. The experimental results show about at most 71 times faster than the desktop computer with a 2.60GHz Pentium4 and 1.5GB Memory running the NCBI BLAST family programs for 3072-residue queries.

2 BLAST Algorithm Overview

As one of the most widely used software tools searching for local similarities between a short query sequence and a large bio-sequence database, BLAST family is composed of five subprograms: BLASTn, BLASTp, BLASTx, TBLASTn and TBLASTx. They provide functionalities for comparing all possible combinations of query and database sequence types by translating the sequences. Nevertheless, the algorithms for each type of search operate are almost identically. The kernel of the algorithm can be summarized as a 3 step procedure:

Find Hits. It creates a list of all short sequence (*word* or *w-mer*) by using sliding window. Then detects substrings of fixed length w in DB stream that perfectly match a substring of the query (typically, $w = 11$ for DNA, 3 for protein) and records the positions of those exact match (hereafter called a "*seed*").

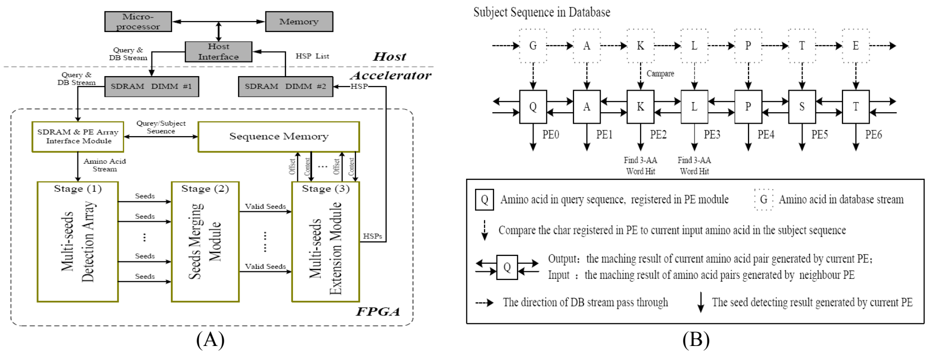
Ungapped Extension. Each *seed* is extended to either side to identify a longer pair of sequences between the query and the subject sequence from the database. Extension is continued until the score of the alignment drops below a threshold. These longer matches are called *high-scoring segment pairs* (HSPs).

Gapped Extension. HSP list is passed to the last stage, which uses the Smith-Waterman algorithm usually to extend it into a gapped alignment. The search result is a list of local alignments giving a measure of similarity between genomic sequences with the decreasing order of alignment's score.

Previous study[9] showed that most of the execution time is spent in the step 1 and 2, over 99%, especially in the first one, over 80%. Therefore, how to detect and locate word matching quickly is critical to accelerate BLAST algorithm.

3 The Structure of Multi-seeds Detection and Parallel Extension Engine

Our BLAST searching system consists of an algorithm accelerator engine and a host processor. The accelerator scans database for an input query sequence and produces a HSP list. Then the host analyses the HSP list in order to assign statistical significance to those matches. The accelerator engine comprises one



Micro-processor
Memory
Host Interface
Query & DB Stream
HSP List
Host
Accelerator
SDRAM DIMM #1
SDRAM DIMM #2
HSP
SDRAM & PE Array Interface Module
Qurey/Subject Seuence
Sequence Memory
Amino Acid Stream
Offset
Context
Stage (1)
Multi-seeds Detection Array
Seeds
Stage (2)
Seeds Merging Module
Valid Seeds
Stage (3)
Multi-seeds Extension Module
HSPs
FPGA
(A)
Subject Sequence in Database
G A K L P T E
Campare
Q A K L P S T
PE0 PE1 PE2 PE3 PE4 PE5 PE6
Find 3-AA Word Hit
Amino acid in query sequence, registered in PE module
Amino acid in database stream
Compare the char registered in PE to current input amino acid in the subject sequence
Output: the maching result of current amino acid pair generated by current PE;
Input : the maching result of amino acid pairs generated by neighbour PE
The direction of DB stream pass through
The seed detecting result generated by current PE
(B)

Fig. 1. (A) The Structure of Multi-seeds Detection and Parallel Extension Engine, (B) The Structure of Multi-seeds Detection Array

FPGA chip (Altera StratixII EP2S130C5), two 1GB SDRAM modules (Micron MT16LSDT12864A) and an USB2.0 interface which is connected to the host. The structure is shown in Figure 1(A). The design fitted in the FPGA includes SDRAM&PE Array Interface Module, Sequence Memory Group, Multi-seeds Detection Array, Seeds Merging Module and Multi-seeds Ungapped Extension Module. SDRAM&PE Array Interface is responsible for system initialization and providing the subject data stream. Sequence Memory holds the query and current subject sequence, and produces the subsequence including seeds for ungapped extension. The last three modules compose the algorithm core. In the following subsections, we take BLASTp algorithm as example to illustrate our implementation in detail.

3.1 Multi-seeds Detecting

The function of this stage is similar to word matching in NCBI BLASTp. It finds out the common appearance of subsequence with 3 amino acids (*3-AA word*) in both query sequence and subject sequence in database. The main difference is that with the help of systolic array, multiple seeds can be detected at each clock cycle, instead of one match per cycle in usual index-based method. Suppose $q_{i-1}q_iq_{i+1}$ and $s_{j-1}s_js_{j+1}$, $(i, j \geq 1)$ are substrings in query and subject sequence respectively. If $(q_{i-1} = s_{j-1}) \wedge (q_i = s_j) \wedge (q_{i+1} = s_{j+1})$ that means a 3-AA word matching occurred and a seed had been detected. The structure of systolic seed detecting array as shown in Figure 1(B).

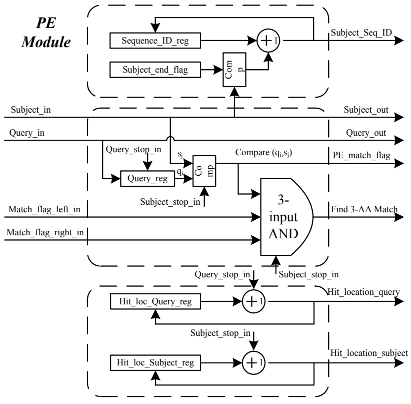
The array consists of a series of *Processing Elements* (PEs), which holds the query(a char per PE) while the database stream flows through the array. *PE[i]*(the *ith* PE) compares $q_i$with s_j, then send the match flag to previous and next PEs, per cycle. At the same time, *PE[i]* receives the match flags, compares results of amino acid pairs (q_{i-1}, s_{j-1}) and (q_{i+1}, s_{j+1}) , generated by neighbour PEs and judges if a seed has been detected. Therefore, the array is capable of processing word matching at up to L Matches/cycle (L is the PE array size) and can report multi-seeds per cycle if they are detected. The array reports two seeds (word *AKL* on PE2 and *KLP* on PE3) at the same time, as shown in Figure 1(B). The multi-seeds detect algorithm is illustrated in Figure 2(A).

Seed detecting and locating are two key functions of PE module implemented. Statement S3 in Algorithm 1 implements the seed-detecting. The location of word hit consists of the offsets in query and in subject, the subject sequence ID in database, which calculated dynamically by S2(Initial phase), S1 and S2 in processing phase respectively.

```
Algorithm 1: Seed-detecting Algorithm for Each PE  (PE[i], 1⩽i⩽L)
Initial phase:  // drive query sequence into PE array and record the location of each AA char
S1: Query_reg ← 0;   PE_match_flag ← 0;   Find_3-AA_Match ← 0;
    Hit_location_Query_reg ← 0; Hit_location_Subject_reg ← 0;  // registers clear
S2: if (Query_stop_in = 0)    // PE array no pause
        Query_reg ← Query_in;   Query_out ← Query_in;
        Location_Query_reg ← Location_Query_reg + 1;  // compute location in query
    else all signals hold on;

Processing phase:      // drive database stream into PE array and detect all the seeds
if (Subject_stop_in = 0)   // the database stream no pause
S1: Subject_out ← Subject_in;
    Location_Subject_reg ← Location_Subject_reg + 1;  // compute location in subject
S2: if (Subject_in = Sequence_end_flag)
        Sequence_ID_Subject ← Sequence_ID_Subject +1; // sequence number in DB
    else if (Sj = Qi)
         Current_PE_match_flag ← 1;  // amino acid pair matching is detected
         Match_flag_left_out ← 1;    // send match flag to previous PE
         Match_flag_right_out ← 1;   // send match flag to next PE
S3: if ((Current_PE_match_flag = 1)&(Match_flag_left_in = 1)&(Match_flag_right_in = 1))
         Find_3-AA_Match ← 1;          // seed is detected and set hitting flag
         Hit_Location_Query ← Hit_Location_Query_reg;
         Hit_Location_Subject ← Hit_Location_Subject_reg;// output match-point location
else all signals hold on;
```

(A)



(B)

Fig. 2. (A) The Seeds Detecting Algorithm for Each PE, (B) PE Module Structure

3.2 Successive Seeds Merging

The systolic array implements the multi-seeds detect procedure very quickly. The array may report a lot of seeds contemporarily when there is enough similarity between the query and subject. It is hard for ungapped extension stage to catch up with the speed of multi-seeds detect with the growth of the array size. Finally, it will cause the unbalance in processing capability between the two stages. To address the problem, we add a seeds merging stage to merge the adjacent successive word-hits (because those seeds belong to identical HSP) into a valid seed and pass it to extension stage as shown in Figure 3(A). The benefit of merging seeds is that the number of valid seed can be reduced significantly. As result, the efficiency of ungapped extension stage is improved since the duplication extension of single HSP had been eliminated.

3.3 Multi-seeds Extension

This stage extends the seeds to either side to identify a longer pair of protein sequence with the score exceeds the threshold. To improve the extension efficiency, we adopt the Multi-channel Parallel Extension Strategy, which will be introduced particularly in section 4.4.

4 FPGA Implementation and Optimization

4.1 Multi-seeds Detection Array

As for the basic cell in multi-seeds detection array, *PE Module* performs the character comparison in pipeline mode and calculates the hit position. The kernel in PE module is a 3-input AND Gate(the middle rectangle area in Figure 2(B)), which implements seed detection. The two input signals named *Match_flag_left_in* and *Match_flag_right_in* generated by adjacent PEs and the current pair match-flag are sent to input ports of the 3-input AND to generate a hit signal when all inputs are TRUE. Three accumulators calculate the offsets of the seed by counting the amino acid characters passed through. Since the *Find 3-AA Match* flag depends on the comparing result of amino acid pairs calculated by adjacent PEs, the calculating the hit flag is the critical path. Timing analysis shows the path delay is less than 3ns, thus it is not the bottleneck in FPGA implementation.

The systolic array consists of a series of Processing Elements. The PE array size is limited by logic (LUT) resource in FPGA. Generally, the larger array size is, the higher searching efficiency can be reached since more words are scanned and more seeds may be detected at the same time. However with the increase in seed-detection capability, multi-seeds recording becomes a critical issue because the number and location of seeds generated by PE array at each time is random. When there is enough similarity between the query and subject, a lot of seeds are reported contemporarily. The overhead recording the seeds orderly will lead to a long pause and low efficiency since the array must be held up until all the seeds have been recorded. To address this problem, we adopt two schemes: decomposing the PE array and merging successive seeds.

4.2 Decomposing the Detection Array

The idea of this strategy is decomposing the Multi-seeds Detection Array into *PE Groups* to record the seeds in parallel. To record the seeds detected by the array, the *Seeds Merging Module* should also be partitioned into some *SM subModules* (corresponding to *PE Groups*), each of which records and merges the seeds detected by local *PE Group* then sends it to a local *Hit FIFO*. The seed in *Hit FIFOs* is delivered to *Hit information FIFO* by multilevel *Fifo Merger Modules*. The partition and hierarchical merging process is illustrated in Figure 3(A).

Suppose the detection array detects H seeds each time and the seeds are located in G groups evenly. Each one has H/G seeds because the position of 3-AA word hitting is random and satisfies the uniform distribution. Recording these seeds only costs H/G cycles by using hierarchical merging strategy since all of the *SM subModules* can collect and combine those seeds in parallel. However, it takes H cycles to finish the process for *Seeds Merging Module* without merging seeds strategy. Therefore, the processing overhead for recording match points only occupies $1/G$ cycles of un-optimization.

The other advantage of decomposing the detection array is eliminating the bottleneck in implementing the huge multiplexer (MUX) between the *Multi-seeds Detection Array* and *Seeds Merging Module* as shown in Figure 3(B). We transform the huge multiplexer into several smaller ones (subMUX) by partitioning the large array and *Seeds Merging Module* into small groups. Thus the multiplexer units no longer become the bottleneck in FPGA implementation. The synthesis results show that the 64 PEs compose a group is an optimal choice. The clock frequency of the detection array with 512 PEs increases from 55MHz to 156MHz since the large *MUX* (512-line to one) is divided into eight small *subMUX* (64-line to one) and it does not change visibly with the array size growth. The main cost in implementation is adding multilevel *Hit FIFO* and

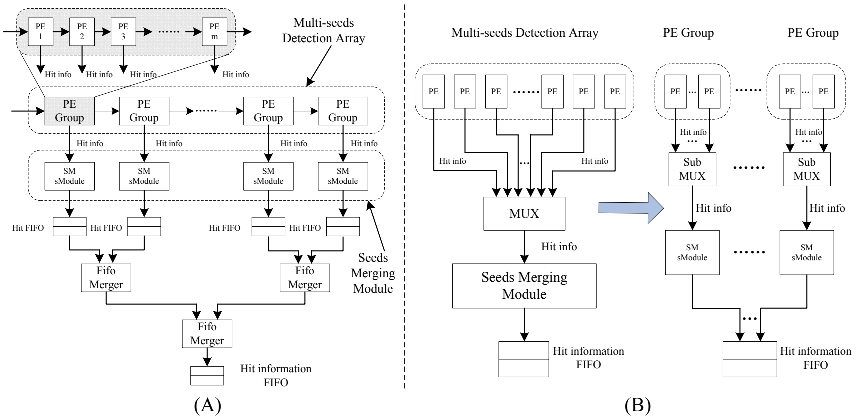


Fig. 3. (A) The Array Partition and Hierarchical Multi-seeds Merging Process, (B) The Port Connection between PE Array and Seeds Merging Module

Merging modules(the level number is $\log_2 G$). However, the storage resource is not bottleneck in our design and the LUT overhead caused by *FIFO Mergers* can be ignored compared to large MUX.

4.3 The Algorithm of Merging Successive Seeds

As far as each *PE Group* is concerned, recording multi-seeds still have to be processed in order. If the two subsequences in query and current subject are highly similar, many seeds will be reported by adjacent PEs contemporarily. It will take a long time to record them one by one. Additionally, it will cause redundant extension if every seed in successive position is sent to the *Hit FIFO* because the seeds belong to the identical HSP.

To filter the redundant seeds and reduce unnecessary extension overhead, we adopt a merging seeds strategy in *SM sModule.* The successive seeds merging algorithm is illustrated in Figure 4. Each *SM sModule* registers the seed flags as statement S1 in processing phase and checks whether word matches are detected. The *function* in S4 finds the first position "1" ("1" means a word hit), which corresponds to the location of first seed detected. The loop in S6 merges the successive word hits into a valid seed then reports it (S8). Suppose the current

```
Algorithm 2：Successive Seeds Merging
Initial phase:
  S1：Hit_location ← 0;  Subject_stop ← 0;  Word_hit_reg[1..m] ← 0;  i ← 0;
Processing phase:
  S1：Word_hit_reg[1..m] ← Word_hit[1..m];  Hsp_flag ← 0;  i ← 0;
  S2：While (Word_hit_reg[1..m] != 0)
        Do S3 ~ S8;
          S3：Subject_stop ← 1;          // Stop current subject sequence passing through
          S4：FUNCTION Find the first location of '1' (n);    // The value returned is n (1≤n≤m).
          S5：Word_hit_reg[n] ← 0;  n ← n+1;  i ← 1;
          S6：While (Word_hit_reg[n] = 1)
                Do { Word_hit_reg[n] ← 0;      // record the match point and clear the hit flag
                     n ← n+1;  i ← i+1; }
          S7：If (i > T)                 // judge if finds a segment matched exactly with enough length
                Hsp_flag ← 1;
          S8：Hit_location ← { Hsp_flag, Hit_info[n] };
  S9：Subject_stop ← 0;  Returns S1;
```

Fig. 4. Successive Seeds Merging Algorithm for Each SM subModule

status of *PE Group* as shown in Figure 1(B). Both PE2 and 3 find a seed at the same time. The *SM sModule* will deliver the two seeds to *Hit FIFO* and the extension operation will be executed twice without the phase of merging seeds. In fact only one extension is needed since the seed *AKL* and *KLP* can be merged into a bigger seed *AKLP*.

Statement S7 judges whether it finds a segment matched exactly with enough length from the count of successive hit flags (variable i). If i is greater than the value set by user (suppose $T = 8$, that means the substring with more than 10 amino acid pairs matched exactly is detected), then *Hsp_flag* is set active. The extension module will no longer extend the seed but output it directly because the extension have finished. Our method reduces unnecessary extension and endows the PE array with a measure of "*macroscopic*" searching ability.

4.4 Multi-channel Parallel Extension Strategy

In this stage seeds are read out from *Hit information FIFO* and extended (adopting Blosum62 Matrix) to either side to identify a HSP. So the *seed's context characters* are needed for extension. FPGA/FLASH[14] constructs the index for each word. Thus, it can get the subsequences directly. Mercury BLASTn[8] prefetches the seed's context because there is only one seed can be detected at a time. However, because of the powerful capability of multi-seeds detecting and the serial extension procedure (only one amino acid pair can be read out per cycle from *Sequence Memory*), the seed-extension capability can't catch up with the throughput of multi-seeds detection units.

To solve the problem, we adopt the *Multi-channel Parallel Extension* method by setting several *Ungapped Extension Modules* as shown in Figure 5. Because each *Extension Module* accesses query and subject sequence memory contemporarily to get the seed's context characters, several *Qry/Sub Memory* copies are fitted to supply enough access ports for multi-channel extension. Thus, multiple seeds from different *Hit info FIFO* can be extended in parallel.

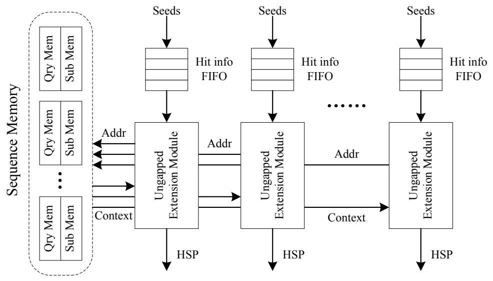


Fig. 5. The Structure of Multi-channel Parallel Extension

5 Experiments and Performance Comparison

The NCBI BLAST software with default parameters (Ver:2.2.16) runs on a desktop computer with a 2.60GHz P4 CPU and 1.5GB Memory. Theoretically, the *Multi-seeds Detection Array* can detect all seeds. We searched a sequence selected in Swiss-Prot with 2048 residues against a small part of Swiss-Prot with 65536 total letters. As a result, 12629 seeds have been detected and 793 seeds have been extended successfully. Using the merging seeds strategy, the number of seeds reported in our design is greatly less than that of the software, but the HSP list accords with the software version. We did a series of tests to evaluate our implementation in the aspects of synthesis performance, storage requirement, actual searching capability and speedup over related works.

5.1 Test 1: Comparing Synthesis Performance to Systolic Array Approaches

We fit our design on FPGA EP2S130C5 with 3072 PEs as shown in Table 1. Without seed-detecting, Tree-BLAST[11] finds HSP directly by adding up the scores of individual alignments between two amino acids. It allocates a BRAM for every four PEs to index the scoring matrix, therefore the BRAM count limits the query size up to 600 on XC2VP70 and 1024 on XC4VLX160. Different from Tree-BLAST, the array size is not limited by BRAM but LUTs in our implementation. It only consumes 38% on-chip memory resource of FPGA XC4VLX160, compared with nearly 88% of related work. We also implement 1024 PEs on XC2VP70-5, the same platform with Tree-BLAST. The result shows our design is superior to Tree-BLAST in both PE number and clock frequency.

Table 1. Performance results and comparison

	Ours			Tree-BLAST[11]	
FPGA	EP2S130C5	XC2VP70-5	XC4VLX160	XC2VP70-5	XC4VLX160
PEs Fitted	3072	1024	3072	600	1024
ALUT/Slice (%)	92098/(87%)	20007/(60%)	48272 /(71%)	−−	78%
Memory (%)	741376 bits/(11%)	36 BRAM/(11%)	110 BRAM/(38%)	−−	88%
Clock (MHz)	113	140	189	110	178
Single PE	42 ALUTs or 31 Slices			−−	−−

5.2 Test 2: Comparing Storage Requirement to Index-Based Approaches

As stated before, the systolic array storage requirements less than index-based approaches. The main storage expense in our design is *Sequence Memory* and multistage *Hit FIFO* (When the array size is 3072, the memory overhead is $692K$bits, which is only 11% of the memory capacity in EP2S130C5). On the contrary, the index-based approach is limited to the capacity of on-chip block RAMs. RC-BLAST[13] fitted a query index with the size of $64K \times 64$bits in Xilinx 4085XLA, which can only record three offsets for each word. Due to the same reason, Mercury[9] and Mitrion[19] have to store the hash table to external SRAM. The delay of memory access becomes the performance bottleneck. Compared to index-based RC-BLAST and systolic-based Tree-BLAST, our approach reduced the storage requirement by about 90% and 50% respectively. Furthermore, in our implementation, little memory requirement reduces the complexity of memory access and lessens the difficulty in FPGA layout and routing.

5.3 Test 3: Comparing to Index-Based Hardware Accelerators

(1) Word-scanning Capability. Most of the current implementations can execute only one word-match per cycle, such as [8],[9],[10],[13],[17],[19]. The word-scanning capability in Mercury BLASTn[8] is 96M matches/s. Mercury

BLASTp[12] designed a two-seed generator, the processing capability reaches up to 219M matches/s for 2048-residue queries. The capability in Multi-engine BLASTn Accelerator[16] achieves 6400M matches/s by using 64 identical parallel engines. Comparatively, our searching engine can execute 294912M word matches per second, over 40 times, by using the multi-seeds parallel detecting approach.

(2) Actual Searching Capability. We use the measurement unit, the number of *Kilo Amino Acids* (*Kaa*) compared to the number of *Mega nucleotides* (*Mnt*) performed every second, *KaaMnt*/s, to measure the actual computing power, because of the variation in the hardware structure, the amount of FPGA resource and clock frequency among all kinds of accelerators. The computational power of FPGA/FLASH and Timelogic Decypher Engine reported in[14] is 451 and 182*KaaMnt*/s respectively. In our implementation, it took 424ms to search a 3072-residue query against drosoph.nt downloaded from NCBI BLAST Database[26] on our engine. We calculate our computational capability:

$$\frac{3Kaa \times 122Mnt}{424ms} = 863KaaMnt/\sec$$

Hence, as for the actual searching capability, our design is 1.91 and 4.74 times as fast as the FPGA/FLASH and Timelogic Decypher Engine respectively.

5.4 Test 4: Comparing Execution-Time to Software Version

We fit the design on our testbed to accelerate the first two stages of NCBI BLAST family programs. The experimental results are listed in Table 2 ([*] The execution time of hardware accelerator is tested by simulation tools (ModelSim SE PLUS 6.2h) for the array size exceeding 4K-PE because of the limitation of FPGA logical resource).

(1) Comparing to BLASTp. We did a series of experiments to search queries selected in Swiss-Prot with different length(128∼8K, which equals array size) among the database Swiss-Prot, including 274,295 sequences, 100,686,439 total letters, downloaded from EBI[27]. Timings were averaged over at least 10

Table 2. Execution time (ms) and speed-up for different queries (SWt: software execution time, HWt: hardware execution time, Sp: Speedup)

Array Size	BLASTp			TBLASTn			BLASTx			TBLASTx			BLASTn		
(Query length)	SWt	HWt	Sp	SWt	HWt	Sp	SWt	HWt	Sp	SWt	HWt	Sp	SWt	HWt	Sp
128	1901	1047	1.82	3203	150	21.3	1594	1034	1.54	4031	158	25.5	6225	1115	5.58
256	3087	1057	2.92	3641	163	22.3	2641	1045	2.53	5906	195	30.3	7378	1128	6.54
512	5603	1090	5.14	5156	199	25.9	4378	1073	4.08	9978	266	37.4	8904	1147	7.76
1K	9327	1157	8.06	9891	254	38.9	7828	1137	6.88	16266	344	47.2	9953	1180	8.43
2K	17814	1227	14.5	15438	358	43.0	13187	1221	10.8	27703	459	60.3	11343	1218	9.31
3K	25132	1487	16.9	20328	424	47.9	18875	1336	14.1	37500	527	71.1	12360	1260	9.81
4K[*]	32469	1620	20.0	25656	480	53.4	26031	1478	17.6	45392	575	78.9	13531	1316	10.28
8K[*]	61162	2207	27.7	47797	570	83.8	49828	1987	25.1	77766	720	108	15344	1393	11.0

queries for each length, and each query's running time was averaged over three identical runs of BLASTp. The execution time in the first two stages of our multi-seeds detection engine and BLASTp for different queries are listed in the column 2~4 of Table 2.

The software execution time, with the growth of query size, is increasing very fast. It only took 1901 ms to search the database with 128-residue queries, while the time added up to 61162 ms to finish the mission with 8K-residue queries. The reason is the cost in both index constructing and searching object increasing greatly with the query size growth. However, the time on our accelerator increases very slowly. This is due mainly to searching cycles of our accelerator equals to the time of database stream flow through the array (that is $L + S$, where L is the array size and S is the database size) plus the pausing time. In the above factors, S is a const and the variation in L can be ignored compared with S. In addition, the pausing cost is related to the number of seeds detected directly. When searching domain (DB) is certain, the valid seeds number and the extension overhead will not increase sharply with the query size growth since the optimized strategies introduced in section 4 are used in our implementation. Thus the larger the array size is, the better the speedup achieves. It is about 17 times faster than the desktop computer for 3072-residue queries. Simulation result shows it can reach 27.7 with the array size of 8K.

(2) Comparing to TBLASTn. Queries were selected in Swiss-Prot with the length from 128 to 8K residues. The run time of TBLASTn is tested for searching the database drosoph.nt downloaded from NCBI BLAST Database[26], which includes 1170 sequences, 122,655,632 letters and the accelerator searches against the *Coding Sequence* (CDS) picked out from drosoph.nt. The time for database translation is not calculated, because this operation need to be done only once and the result can be reused for many other applications.

TBLASTn is used for searching protein sequence against DNA database. It translates all the DNA sequences into the 6 possible potential proteins before searching. Therefore, for the same query, it is slower than BLASTp. However, the execution time of our accelerator does not increase steeply with the query size growth for the same reason as the Test4(1), so the higher speedup can be achieved. Our implementation has a speedup of approximately 48 for 3072-residue queries using the array with 3072 PEs and the value can reach 84 for 8K-PE array.

(3) Comparing to BLASTx. BLASTx is used for searching DNA sequence against protein database. The queries are translated into six-frame protein sequence before searching. We selected a series of queries from drosoph.nt with the length from 128 to 8K residues to search against the protein database, Swiss-Prot, downloaded from EBI[27]. The execution time for different queries is also listed in Table 2.

For the same reason as BLASTp and TBLASTn, the software execution time of BLASTx is increasing very fast with the growth of query size. It only took 1594 ms to search the database with 128-residue queries, while the time added up to 49828 ms to finish the mission with 8K-residue queries. On the other hand, the execution time of hardware BLASTx algorithm does not increase steeply with the query size growth. Thus, the speedup raises with the growing query size. It is about 14 times faster than the desktop computer for 3072-residue queries. Simulation result shows it can reach 25 with the array size of 8K.

(4) Comparing to TBLASTx. Queries were selected in drosoph.nt with the length from 128 to 8K residues. The run time of NCBI TBLASTx is tested for searching the database drosoph.nt downloaded from NCBI BLAST Database. The accelerator searches against the *Coding Sequence* (CDS) picked out from drosoph.nt and the time for database translation is not calculated.

TBLASTx is used for searching six-frame translation of DNA sequence against six-frame translation of DNA database. Both the queries and the database are translated into the six possible potential proteins before searching. Therefore, it is much slower than other programs in BLAST family. However, the execution time of our accelerator does not increase steeply with the query size growth for the same reason as state before, so the much higher speedup can be achieved. Our implementation has a speedup of about 71 for 3072-residue queries using the array with 3072 PEs and the simulation result shows it can reach 108 with the array size of 8K.

(5) Comparing to BLASTn. The four subprograms in BLAST family: BLASTp, BLASTx, TBLASTn and TBLASTx provide functionalities for comparing all possible combinations of query and database types, but they search Protein vs. Protein sequence actually. Therefore, all of them can execute on our Multi-seeds Detection and Parallel Extension Engine with *w=3*, named PSSE (Protein Sequence Search Engine).

However, BLASTn program is used for searching DNA sequence against DNA database and the word length equals 11-nucleotide, typically. To accelerate the BLASTn program, we also designed a Multi-seeds Detection and Parallel Extension Engine with *w=11*, named DSSE (DNA Sequence Search Engine).

The main structure of DSSE is consistent with PSSE. The main difference lies in PE structure. The *PE[i]*(the *ith* PE) in DSSE array compares $q_i$with s_j, then send the match flag to adjacent *Ten* PEs (from *PE[i-5]* to *PE[i-1]* and from *PE[i+1]* to *PE[i+5]*), per cycle. At the same time, *PE[i]* receives the match flags, compares results of residue pairs generated by adjacent *Ten* PEs and generate a hit signal when all the ten inputs and the match flag of current residue pairs are TRUE. That means the exactly matched pairs $(q_{i-5} \cdots q_i \cdots q_{i+5} || s_{j-5} \cdots s_j \cdots s_{j+5})$ in the query and subject are found and a *11-mer seed* is detected. Then send it to extension stage.

We did a series of experiments to test our DSSE's searching capability. Queries with different length are selected in drosoph.nt and the experiment environment is the same as the other programs in BLAST Family. As shown in the last three columns of Table 2, the software execution time of BLASTn is increasing obviously. It increases from 6225 ms to 15344 ms with the query size growing from 128 to 8K-residue. However, the execution time of hardware BLASTn DSSE does not increase visibly for the same reason as the PSSE. Our implementation has a speedup of approximately 10 for 3072-residue queries using the array with 3072 PEs.

6 Conclusion

In this paper we present a systolic array, which supports Multi-seeds Detection and Multi-channel Ungapped Extension in parallel, to accelerate the first two stages of NCBI BLAST family algorithms. Our implementation reduces unnecessary extension by using merging seeds strategy and decreases the memory requirement on-chip as a result of eliminating the lookup tables. The experimental results show about 17, 48, 14, 71 and 10 times faster than BLASTp, TBLASTn, BLASTx, TBLASTx and BLASTn programs running on a desktop computer with 2.60GHz P4 CPU for 3072-residue queries, respectively. Furthermore, our Multi-seeds Detecting Array also can be used to accelerate the seed detection stage in other seed-based heuristic searching applications.

Acknowledgments. This work is supported by the National High Technology Research and Development Program of China (2007AA01Z106) and National Natural Science Foundation of China (60633050 and 60621003).

References

1. Altschul, S.F., Gish, W., et al.: Basic local alignment search tool. Molecular Biology, 403–410 (1990)
2. Farmerie, W.G., Hammer, J., et al.: Having a BLAST: Analyzing Gene Sequence Data with BlastQuest. In: Proc. 14th International Workshop on Database and Expert Systems Applications, pp. 37–41 (2003)
3. Kim, T.-K., Oh, S.-K., Lee, K.-H., Roh, D.-H., Cho, W.-S.: HGBS: a hardware-oriented grid BLAST system. In: 5th International Symposium on Cluster Computing and the Grid, pp. 520–526 (2005)
4. Lin, H., Ma, X., Chandramohan, P., Geist, A., Samatova, N.: Efficient Data Access for Parallel BLAST. In: Proc. 19th International Parallel and Distributed Processing Symposium (2005)
5. Oehmen, C., Nieplocha, J.: ScalaBLAST: A Scalable Implementation of BLAST for High-Performance Data-Intensive Bioinformatics Analysis. IEEE Trans. on Parallel and Distributed Systems (2006)
6. Yutao, Q., Feng, L.: CyberparaBLAST: the Parallelized BLAST Web Server. In: Proc. 2nd International Conference on Cyberworlds, pp. 474–477 (2003)
7. NCBI, GenBank Growth Statistics (2006), `http://www.ncbi.nlm.nih.gov/Genbank/genbankstats.html`
8. Buhler, J.D., Lancaster, J.M., et al.: Mercury BLASTN: Faster DNA Sequence Comparison Using a Streaming Hardware Architecture. In: Proc. 3rd Annual Reconfigurable Systems Summer Institute (2007)
9. Krishnanurthy, P., Buhler, J., et al.: Biosequence Similarity search on the Mercury system. In: Proc. 15th IEEE International Conference on Application-Specific Systems, Architectures and Processors, pp. 2004–365 (2004)

10. Lancaster, J., Buhler, J., et al.: Acceleration of Ungapped Extension in Mercury BLAST. In: Proc. 7th Workshop on Media and Streaming Processors, pp. 50–57 (2005)
11. Herbordt, M.C., Model, J., et al.: Single Pass, BLAST-Like, Approximate String Matching on FPGAs. In: Proc. 14th Annual IEEE Symposium on Field-Programmable Custom Computing Machines, pp. 217–226 (2006)
12. Jacob, A., Lancaster, J., et al.: FPGA-accelerated seed generation in Mercury BLASTp. In: Proc. 15th Annual IEEE Symposium on Field-Programmable Custom Computing Machines, pp. 95–106 (2007)
13. Muriki, K., Underwood, K.D., et al.: RC-BLAST: Towards a Portable, Cost-Effective Open Source Hardware Implementation. In: Proc. 19th IEEE International Parallel and Distributed Processing Symposium (2005)
14. Lavenier, D., Xinchun, L., Georges, G.: Seed-based Genomic Sequence Comparison using a FPGA/FLASH Accelerator. In: IEEE International Conference on Field Programmable Technology, pp. 41–48 (2006)
15. Sotiriades, E., Dollas, A.: Design Space Exploration for the BAST Algorithm Implementation. In: Proc. 15th Annual IEEE Symposium on Field-Programmable Custom Computing Machines (2007)
16. Sotiriades, E., Kozanitis, C., Dollas, A.: FPGA based Architecture for DNA Sequence Comparison and Database Search. In: Proc. 20th IEEE International Parallel and Distributed Processing Symposium (2006)
17. Chang, C.: BLAST Implementation on BEE2. Electrical Engineering and Computer Science University of California at Berkeley (2005), http://bee2.eecs.berkeley.edu
18. CLC Desktop Hardware-Acceleration. White paper on CLC Bioinformatics Cube (2006), http://www.clccube.com
19. Mitrion.Inc.: NCBI BLAST Accelerator (2007), http://www.mitrionics.com
20. Timelogic.Inc.: Timelogic DeCypher BLAST Engine (2006), http://www.timelogic.com/decypher_blast.html
21. Hoang, D., et al.: FPGA Implementation of Systolic Sequence Alignment. In: Proc. 2nd International Workshop on Field-Programmable Logic and Applications. LNCS, pp. 183–191 (1992)
22. Hoang, D., et al.: Searching Genetic Databases on Splash2. In: Proc. IEEE Workshop on FPGAs for Custom Computing Machines, pp. 185–191 (1993)
23. Guccione, S., Keller, E.: Gene Matching Using JBits. In: Glesner, M., Zipf, P., Renovell, M. (eds.) FPL 2002. LNCS, vol. 2438, pp. 1168–1171. Springer, Heidelberg (2002)
24. Oliver, T., et al.: Hyper Customized Processors for Bio-Sequence Database Scanning on FPGAs. In: Proc. ACM/SIGDA 13th international symposium on Field programmable gate arrays, pp. 229–237 (2005)
25. Court, T.V., Herbordt, M.C.: Families of FPGA-Based Accelerators for Approximate String Matching. Microprocessors and Microsystems 31, 135–145 (2007)
26. NCBI BLAST Database, National Center for Biotechnology Information (2006), http://www.ncbi.nih.gov/BLAST
27. EBI, European Bioinformatics Institute (2007), http://www.ebi.ac.uk/uniprot/database/download.html

Fast Structured Motif Search in DNA Sequences

Mihail Halachev and Nematollaah Shiri

Dept. of Computer Science & Software Engineering,
Concordia University, Montreal, Quebec, Canada
{m_halach, shiri}@cse.concordia.ca

Abstract. We study the problem of structured motif search in DNA sequences. This is a fundamental task in bioinformatics which contributes to better understanding of genome characteristics and properties. We propose an efficient algorithm for Exact Match, Overlapping Structured motif search (EMOS), which uses a suffix tree index we proposed earlier and runs on a typical desktop computer. We have conducted numerous experiments to evaluate EMOS and compared its performance with the best known solution, SMOTIF1 [1]. While in some cases the search time of EMOS is comparable to SMOTIF1, it is on average 5 to 6 times faster.

Keywords: DNA sequences, structured motif, suffix tree, performance.

1 Introduction

Advances in bioinformatics have facilitated experiments in biology laboratories and genome sequences acquired are growing at exponential rate. Understanding the properties and characteristics of these sequences requires various types of searches to be performed. A fundamental task in bioinformatics is searching in new sequences for previously known information, expressed as *structured motifs*. Examples of potential applications include searching for composite regulatory binding sites in DNA sequences and finding *long terminal repeat* (*LTR*) *retrotransposons*, which have significant presence in typical mammalian genome and are believed to have major impact on genome structures and functions [2,3].

A structured motif consists of several simple motifs, interleaved by variable-length bounded gaps. Each simple motif can be represented either as a string of symbols from a specific alphabet (*pattern* representation), or as a matrix which gives the probability of observing a specific nucleotide at each position in the simple motif (*profile* representation). A gap is represented as [x, y], which denotes the minimum and the maximum gap sizes allowed between two adjacent simple motifs. This structured motif model provides a suitable way for simultaneously searching for several (related to each other) DNA sequences, while accounting for some possible evolutionary mutations.

Consider the following sample structured motif $SM = M_1[2,5]M_2[6,7]M_3$, taken from [4]. In Table 1, rows 2 to 5 represent each simple motif as a profile, while row 6 gives their corresponding pattern representations using the IUPAC alphabet. As in [1,5], in this work we adopt the IUPAC alphabet for pattern representation of the

Table 1. A Sample Structured Motif

Bases	M_1									[2,5]	M_2		[6,7]	M_3		
A	2	12	17	1	11	1	35	0	24		1	0		3	1	35
C	0	10	8	5	2	0	0	19	0		0	25		5	35	1
G	2	5	5	2	10	34	1	0	0		26	11		0	0	0
T	32	9	6	28	13	1	0	17	12		9	0		28	0	0
IUPAC	D	N	N	N	N	D	R	Y	W	[2,5]	D	S	[6,7]	H	M	M

Table 2. The IUPAC Alphabet

Bases	A	C	G	T	U	A,G	C,T	G,T	A,C	G,C	A,T	C,G,T	A,G,T	A,C,T	A,C,G	A,C,G,T
Symbol	A	C	G	T	U	R	Y	K	M	S	W	B	D	H	V	N

structured motif, and use the DNA alphabet {A,C,G,T} for sequence data to be searched. The mapping convention between the DNA bases and the corresponding IUPAC symbols is shown in Table 2. To avoid confusion, we refer to characters in a pattern motif as *symbols* and to sequence characters as DNA bases, or *bases* for short.

The goal of structured motif search, given a sequence *S*, is basically to find the starting positions *p* in *S* at which a match between the query *SM* and a substring of *S* occurs. For example, consider our sample *SM* and *S*[*p*, *p*+25], a sample substring of *S* starting at position *p*, shown in the first row in Figure 1. The next four rows in the figure show 4 matches between *SM* and *S*[*p*], for the gap sizes {3,6}, {3,7}, {5,6}, and {5,7}, respectively.

S[*p*, *p*+25]=	T	A	C	G	T	A	A	T	T	G	G	A	A	C	A	C	G	C	A	T	A	C	A	A	A	A
SM =	D	N	N	N	N	D	R	Y	W	-	-	-	D	S	-	-	-	-	-	-	H	M	M			
SM =	D	N	N	N	N	D	R	Y	W	-	-	-	D	S	-	-	-	-	-	-	-	H	M	M		
SM =	D	N	N	N	N	D	R	Y	W	-	-	-	-	-	D	S	-	-	-	-	-	-	H	M	M	
SM =	D	N	N	N	N	D	R	Y	W	-	-	-	-	-	D	S	-	-	-	-	-	-	-	H	M	M

Fig. 1. Matching a *SM* with a substring of *S*

In this paper, we propose a search algorithm for finding structured motifs represented as patterns over the IUPAC alphabet. The algorithm proceeds in three steps. First, based on a heuristic using the information content of each simple motif in *SM*, the algorithm selects a simple motif estimated to have the fewest number of matches in *S*. In the second step, using a suffix tree index created for the sequence *S*, we retrieve all starting positions in *S* at which a match with the selected simple motif occurs. We refer to these positions as *anchors*. In the final step, the simple motif and the sequence *S* are aligned at all anchor positions, to search for matches for the whole structured motif *SM*. A match of *SM* in *S* occurs if, for a unique and allowed combination of the gap values, a match for all simple motifs of *SM* is found in *S*.

The rest of the paper is organized as follows. In Section 2 we provide a background and review related work. Section 3 recalls the suffix tree index we use in this work. In Section 4, we propose our algorithm for structured motif search, EMOS (for Exact Match, Overlapping Structured motif search). Experiments and results are presented in Section 5. We conclude and outline future work in Section 6.

2 Background and Related Work

Formally, a structured motif *SM* consisting of two or more *simple motifs* M_k separated by gaps of possibly variable lengths, is represented as $SM = M_1[i_1, j_1]M_2[i_2, j_2]M_3\ldots M_{n-1}[i_{n-1}, j_{n-1}]M_n$, where $i_k \leq j_k$. The gap values i_k and j_k indicate respectively the minimum and the maximum gap sizes allowed between the last symbol in M_k and the first symbol in M_{k+1}.

In the related literature, we can find several variants of the structured motif search, depending on the definition of match and the constraints imposed by the gaps. We next review these variants and define the problem addressed in this paper.

An *exact match* of *SM* in *S* is defined as an "exact" match between all the symbols in *SM* and the corresponding bases in a substring of *S*, obeying the gap constraints. Note that when using IUPAC representation for a motif and using DNA representation for the sequence data, the notion of "exact" match between a motif symbol and a sequence base is defined by the mapping scheme shown in Table 2. Examples of exact matches of *SM* in *S* are shown in Figure 1. Even though this "exact" matching allows some flexibility and is approximate in nature, some applications may require even more relaxed matching. One such example is allowing for errors (i.e., insertions, deletions, and substitutions of symbols/bases) when comparing simple motif symbols and their corresponding DNA bases. This variation of the structured motif search is referred to as *approximate match* [1]. For example, in our sample *SM* and *S* (Fig. 1), suppose the number of errors allowed per simple motif is at most 1. Then approximate match search returns a match for gap sizes {2, 6}, in addition to the results returned by exact match. Another type of approximate search allows up to *q'* (out of all *n*) simple motifs to be missing. This version of the problem is referred to as *q-occurrence search* [1,5] and its goal is to find all occurrences of *SM* in *S*, where at least $q = n - q'$ simple motifs are matched. For our sample *SM* and *S*, this type of search for $q = 2$ will return a match for gap sizes {4, 6}, in addition to the results returned by exact match. In this work, we are interested in *exact match* structured motif search, for which all IUPAC symbols in *SM* must match (according to the mapping scheme in Table 2) the corresponding DNA bases in *S*, and no missing simple motifs are allowed.

Considering the gap constraints, the *fixed* motif search problem is the simplest case in which $i_k = j_k$, for all *k* in [1, *n*-1], and every i_k is known in advance and is positive. When at least one i_k is different from j_k, the problem is referred to as *structured* motif search. Yet another version of the problem allows negative values for i_k, with the restriction that the absolute value of i_k is smaller than the size of M_k. This amounts to allowing partial overlap between M_{k+1} and some of the rightmost symbols of M_k, and hence adds more flexibility to the search. This variant is referred to as *overlapping structured* motif search and is addressed in this work. If the gap ranges are not known in advance, the problem is called *extended structured* motif search [1].

To summarize, the problem we investigate in this paper is the exact match overlapping structured motif search where structured motifs are represented as patterns over the IUPAC alphabet, and no missing simple motifs are allowed.

Searching for structured motifs represented as patterns is an active research area [6,7,8,5,1]. Anrep [6,7] allows the user to specify the simple motifs of the structured motif via declarative, free-format, and strongly typed language, called A. These simple motifs are referred to as *network expressions*, and are essentially regular expressions, excluding the Kleene star operator. Using this notation, the user may specify the required parameters for approximate match and q-occurrence search. The gaps between simple motifs are called *spacers*. Anrep executes in two steps. First, using an ε-automaton, it searches for simple motifs that satisfy the approximate threshold constraints. In the second step, it looks for structured motif occurrences using a backtracking match algorithm, optimized according to some statistical criterion. As one of the first efforts on structured motif search, Anrep provides a unified pattern representation of bio-sequence data, however the search performance was improved by other solutions, discussed next.

Navarro *et al.* [8] refer to structured motifs as *Classes of Characters and Bounded Gaps* (CBG) expressions. The proposed solutions (forward and backward search) are based on a non-deterministic ε-automaton with bit-parallelism. CBG provides efficient search for relatively short structured motifs whose maximum span (i.e., the sum of the sizes of all simple motifs plus the sum of the maximum sizes of the gaps) is less than the number of bits in a computer word. For longer patterns, the size of the automaton grows accordingly and the advantage of bit-parallelism deteriorates when performing bit operations on several computer words instead of one. As a result, the application of CBG is limited to searching for patterns with small number of symbols and gaps.

SMaRTFinder [5] adopts a two-step approach. It first finds all the occurrences of each simple motif, using a suffix tree (ST) index for the sequence data. The index can be constructed either on the fly (the *lazy* approach), which builds only parts of the ST that are relevant to the particular motif, or construct the entire ST for the sequence in advance (the *eager* approach). In the second step, SMaRTFinder solves a constraint satisfaction problem, by building a constraint graph for all possible pairs of simple motifs occurrences (represented as nodes) which locally satisfy the gap constraints. Subsequently this graph is pruned only to *feasible* nodes (i.e., nodes that represent occurrences of simple motifs that certainly belong to a match for the structured motif), and the set of all structured motif matches is obtained by a depth-first traversal of the pruned graph. The experimental results in [5] indicate a significant search time advantage of SMaRTFinder over Anrep, when searching for a randomly generated set of 1,000 structured motifs in a 5 MB DNA sequence. Further, the SMaRTFinder exhibits linear search time performance with respect to the number of matches found, while the performance of Anrep depends strongly on the success of the statistical optimization of the backtracking match algorithm. An important note made by Policriti *et al.* [5] is that the performance of SMaRTFinder for such number of queries is independent of the approach (eager or lazy) taken for constructing the ST index.

In a recent work [1], Zhang and Zaki proposed the SMOTIF technique for structured pattern and profile motif search. It consists of several algorithms, which support exact match, approximate match, and *q*-occurrence search operations. There are two alternative implementations for structured pattern search: SMOTIF1 and SMOTIF2. Below, we review their approaches to the exact match search problem.

As a first step, SMOTIF1 scans once the sequence to be searched, and then converts it into an equivalent inverted format [9,10], where each character in the sequence is associated with its *post-list* – a sorted list of the positions at which the base occurs in the sequence. Also, the structured motif is converted to its SMOTIF representation, by adding a gap [0,0] between adjacent symbols within each simple motif, and then consolidating the gaps, if possible. For example, the motif GCN[0,1]TB is converted into G[0,0]C[1,2]T[0,0]B. The second step in SMOTIF1 starts from the last two motif symbols (T and B in this example) and computes the post-list of T[0,0]B, using *positional joins* over T and B's post-lists (each of which viewed as a union of the post-lists of their corresponding matching bases, computed in the first step). The result essentially is the list of all starting positions of T[0,0]B in the sequence. Next, the algorithm recursively expands the positional join process, by considering the first unprocessed symbol to the left (in our example, C), and performs a positional join over its post-list and the post-list of T[0,0]B. Upon completion of the recursive positional join process, the post-list of the entire structured motif is obtained, i.e., the list of all starting positions of the structured motif in the sequence. If required by the application, the set of matching positions for each symbol in the motif can be recovered.

In contrast to SMOTIF1, which performs positional joins on the post-lists of individual motif symbols, SMOTIF2 performs the positional join process on the post-lists of the whole simple motifs. That is, the post-list (i.e., the starting positions) of each simple motif is obtained by *lazy* construction of the same ST index as in SMaRTFinder. While this first step is the same for SMOTIF2 and SMaRTFinder, the main difference between them is in the second step, in which they process the information obtained. The positional join technique of SMOTIF2 proves to be more efficient than the constraint satisfaction technique used in SMaRTFinder.

The experimental results in [1] show that SMOTIF1 and SMOTIF2 are respectively up to 18 and 4 times faster than SMaRTFinder searching for three real-life motifs in chromosome 1 of *A. Thaliana*. Also, the performance of the three search techniques is compared more comprehensively for searching chromosome 20 of *Homo sapiens* for a set of 100 random structured motifs. Again, SMOTIF1 and SMOTIF2 are considerably faster than SMaRTFinder, 6 and 4 times respectively. Further, SMOTIF1 performs significantly better than SMOTIF2 when no missing simple motifs are allowed. One of the shortcomings of SMOTIF2 (as well as of SMaRTFinder) is that in the first step, it searches for the exact match occurrences of all simple motifs of a *SM*. In case of long sequences and/or several simple motifs that have low information content, this may lead to enormous intermediate output which does not fit in main memory. As a result, SMOTIF2 and SMaRTFinder run out of memory in such cases and cannot conclude the *SM* search.

Our EMOS algorithm has three important characteristics that distinguish it from the above solutions. First, it handles efficiently some typical structured motif search challenges, e.g., the maximum span of a *SM* could be several thousand bases long; some of the simple motifs could be very short and/or with low information content, etc. Second, while EMOS can search in very long sequences (e.g., chromosome 2 of *Homo sapiens* containing around 238 million bases), it does not require extensive memory space and runs on typical desktop computers (with 2 GB RAM as in our computer system). Third, the fastest known solution for exact match overlapping structured motif search is SMOTIF1 [1]. The results of our experiments indicate that for majority of the cases, EMOS is 5 to 6 times faster than SMOTIF1.

3 The STTD64 Index

Similar to SMOTIF2 and SMaRTFinder, a part of our solution to the structured motif search problem uses a suffix tree based index. As illustrated in [11], such an index is suitable for biological sequences and provides efficient and versatile support for numerous bioinformatics search applications. In a previous work [12], we proposed the suffix tree indexing structure STTD64 (Suffix Tree, Top Down, 64 bits). Using this index, we also developed efficient and scalable algorithms for string searching (exact match and k-mismatch problems), and for finding supermaximal repeats in sequences. Interested readers are referred to [13] to experiment with and evaluate these applications accessible through a web-based interface. Our focus in this work is to develop an efficient motif search algorithm, for which we use the STTD64 index. Here, we assume the index is already constructed and stored on disk. In Section 5, we justify this assumption. Below we review the key points in the STTD64 representation to better understand the EMOS algorithm and a source of its efficiency.

To illustrate the STTD64 index, consider sequence *S* = AGAGAGCTT$. Figure 2 shows a general, high-level graphical representation of a ST for *S*. In the figure, the numbers in squares illustrate the order in which the nodes in the tree are evaluated and recorded in STTD64. Each edge is labeled with the corresponding bases from *S*. The number below each leaf node *s* indicates the starting location in *S* at which we can find the suffix indicated by the labels of the path from the root to *s*.

Figure 3 shows the actual STTD64 index for *S*. Each ST node is represented as a record of size 64 bits (shown vertically), divided into four fields, as follows. For each node *v*, we store, in the first field of 32 bits (the first row in Fig. 3), its left pointer value. This value, denoted *lp*(*v*), is the sum of the leftmost starting location in *S* at which we can find the substring encoded from the ST root to node *v* plus the *depth* of node *v*. The depth of a node *v* is defined as the number of characters from the root to the parent of *v*. For example, for node 10, the substring that is encoded from the root to this node is "GAG." The leftmost occurrence of "GAG" in *S* starts at *S* [1]. The parent of node 10 is node 2, and hence the depth of node 10 is 1. Thus, *lp*(10) = 1+1 = 2. For each ST node, the second field of size 1 bit (second row in Fig. 3), stores its leaf value. A leaf value 0 indicates that the current node is a branch node, while leaf value 1 indicates a leaf node. For clarity, the leaf nodes are shown in gray in the figure. For each ST node, the third field of size 1 bit (third row in Fig. 3), stores its rightmost value. A rightmost value 1 indicates that the current node is the rightmost child of its parent. For example, node 5 is the rightmost child of the root, and node 7 is the rightmost child of node 1. In the last field of size 30 bits (the fourth row) we store different information depending on whether the current node is a branch or a leaf node. In case it is a branch, we store a pointer to the location in the STTD64 index at which the first child of this branch node is stored, thus providing means for downwards traversal of the ST. These pointers are illustrated by the arrows above Figure 3. In case of a leaf, in the fourth field we store the depth of the leaf node.

It should be noted that the starting locations of the suffixes in *S* indicated in Figure 2 by the number below each leaf node are just to facilitate our description in the text; they are not explicitly stored in the STTD64 representation. For search applications which use STTD64, the availability of depth values allows for fast computation of the starting locations. For example, the starting location of the suffix encoded by the path from the root to node 13 is computed by subtracting the depth value of node 13 from its *lp* value, i.e., the starting location of the suffix "GAGCTT$" is 6 – 3 = *S*[3] (see Figures 2 and 3). Storing the information required for this computation in a single node eliminates unnecessary ST traversals, leading to a significant decrease of the number of disk I/O operations, and eventually to search time improvements for applications based on STTD64.

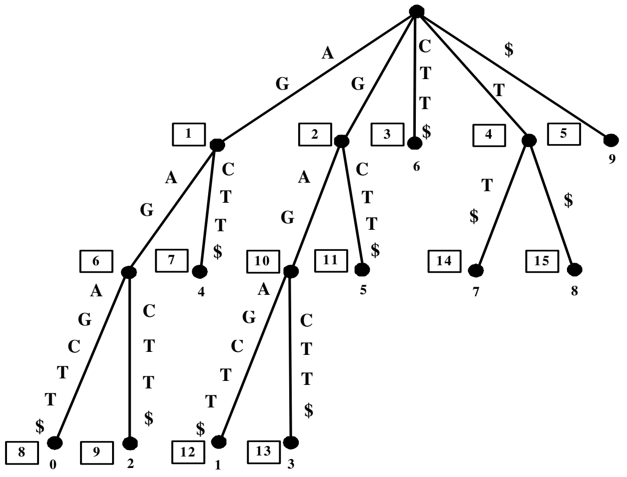
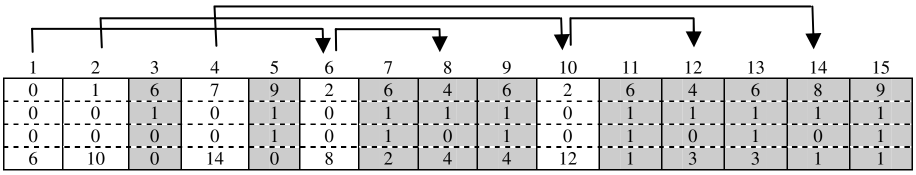


Fig. 2. A ST for *S* = AGAGAGCTT$



1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	1	6	7	9	2	6	4	6	2	6	4	6	8	9
0	0	1	0	1	0	1	1	1	0	1	1	1	1	1
0	0	0	0	1	0	1	0	1	0	1	0	1	0	1
6	10	0	14	0	8	2	4	4	12	1	3	3	1	1

Fig. 3. STTD64 Representation of ST for *S* = AGAGAGCTT$

Although inspired by the suffix tree based *wotd* index [14] adopted in SMaRT-Finder and SMOTIF 2, STTD64 differs from it. First, STTD64 requires 16 bytes per sequence character (in the worst case), compared to the 12 bytes required by the *wotd*, which is known to be the most space efficient ST representation [14]. The reason is that in STTD64 we store the additional information, i.e., the depth of the ST leaf nodes. Regardless of the larger storage requirements, STTD64 construction time is comparable with TDD [15], which is known to be the most time efficient ST construction technique. Second, STTD64 has a theoretical limit of 4 GB on the size of the sequence being indexed (assuming there are no repeats longer than 2^{30} bases), and in [12] we have shown its capability for indexing the entire human genome (approximately 3 GB) on a regular desktop computer with 2 GB RAM. On the other hand, the theoretical limit for sequences handled by *wotd* is around 700 million bases [14]. However, since both construction algorithms (*wotd-eager* and *wotd-lazy*) proposed in

[14] are memory based, the practical limit depends on the available RAM size. The results in [14] show that *wotd-eager* construction algorithm needs around 10.5 bytes per input character, which for a typical desktop with 2 GB RAM leads to a practical limit on the input sequence size of around 200 million bases. The exact memory requirement is not given for *wotd-lazy*, but it is more than *wotd-eager*.

Based on the above comparisons, we find that STTD64 is more suitable and efficient for indexing biological sequences, and we use it as our index in conjunction with the EMOS algorithm, presented next.

4 The EMOS Algorithm

Our proposed EMOS (Exact Match, Overlapped Structured motif search) algorithm takes as input a structured motif *SM* represented as a pattern over the IUPAC alphabet, a DNA sequence *S* to be searched, and its STTD64 index. The output of the algorithm consists of all positions in *S* at which an exact match between the *SM* and a substring of *S* starts, together with the length of the match. The EMOS algorithm is presented in Figure 4. Our algorithm takes a three-step approach in solving the structured motif search problem. In a nutshell, in the first step, based on preprocessing the structured motif, a suitable simple motif (M_a) is selected. In the second step, using the STTD64 index, all exact occurrences of M_a in the sequence are found. In the third step, the structured motif is aligned with the sequence by using the exact occurrences of M_a as anchors, and the symbols of the remaining motifs are compared with the corresponding sequence bases.

To illustrate the EMOS algorithm, consider the following example: Find all exact match occurrences of *SM* = WN[-1,2]KW[2,4]Y in our sample sequence *S* = AGAGAGCTT$. For clarity of the exposition and to avoid unnecessary technical details, in our discussion we will use the graphical ST representation of the index (Fig. 2), noting that EMOS uses the equivalent STTD64 representation (Fig. 3).

In Step 1 of our algorithm, we preprocess *SM* to select a suitable simple motif, as the anchor motif M_a. Intuitively, "most suitable" is a simple motif with smallest number of exact matches in *S*, which in the subsequent phase are to be used as anchors. Since we do not know in advance the number of exact matches of each simple motif in *S*, we employ the following fast heuristic for selecting M_a. By reading *SM* once, we compute the *selectivity power* (*SP*) of each simple motif, by considering the information content of its symbols, according to Table 2. For example, the selectivity powers of the three simple motifs of the *SM* are computed as follows: $SP(M_1) = SP(\text{WN}) = {}^2/_4 * {}^4/_4 = 0.50$; $SP(M_2) = SP(\text{KW}) = {}^2/_4 * {}^2/_4 = 0.25$; $SP(M_3) = SP(\text{Y}) = {}^2/_4 = 0.50$. Assuming uniform distribution of the DNA bases in *S*, the expected number of exact matches is estimated to be around 50% of the size of *S* for M_1, 25% for M_2, and 50% for M_3. Thus, we select M_2 to be our M_a. One drawback of this selection heuristic is that in practice the distribution of bases in nucleotide sequences is not strictly uniform. However, our experimental results indicate that although not optimal, the employed heuristic provides an overall *SM* search speedup of 2 to 3 times, compared to an alternative in which we pick M_a randomly, as discussed in more detail in the experimental section. Note that since the *SP* for each simple motif is strictly greater than 0, this step

```
Algorithm EMOS(Sequence S,Index STTD64,Structured Motif SM)
 0. Read Sequence S from disk to memory
Step 1 //Preprocess the Structured Motif SM
 1. Select the anchor simple motif, $M_a$;
Step 2 //Find all occurrences of $M_a$ in S
 2. ARS := STTD64 root; //ARS is Answer Root Set
 3. for each symbol in $M_a$ (starting from $M_a$[0]){
 4.   dna_set := convert $M_a$[i] to its corresponding DNA base(s);
 5.   for each STTD64 node in ARS, node_old
 6.     for each node_old outgoing edge
 7.       if (its label match one base in dna_set)
 8.       then insert in ARS edge destination node,node_new;
 9.     delete node_old from ARS;
10. } //at this point ARS contains all answer roots
11. $M_a$(all):=∅;
12. for each answer root (AR) node in ARS{
13.   for each leaf in subtree rooted at AR
14.     compute S location, add it to $M_a$(all);
15. }//$M_a$(all) contains start locations of all exact matches of $M_a$ in S
Step 3 //For each L in $M_a$(all), align SM with S, check the other simple motifs
16. L = first location in $M_a$(all);
17. while (not all locations in $M_a$(all) are examined){
18.   sm_len = $|M_a|$;
      //Explore SM to right and left of $M_a$, matching remaining motifs
19.   extendRight(a, L);
20. }//End EMOS

extendRight(v, loc){
  //$M_v$ - verified simple motif; loc - a starting location of $M_v$ in S;
  right := v+1;
  while (right <= n){
    for each x, $i_v$ <= x <= $j_v$ (starting from $i_v$){
      align $M_{right}$ and S such that $M_{right}$[0] is at S[loc+$|M_v|$+x];
      compare $M_{right}$ symbols with the corresponding S bases
      if (mismatch) x++; //consider next gap value
      else //exact match for $M_{right}$ found
        sm_len = sm_len + $|M_{right}|$ + x;
        loc = loc+$|M_v|$+x; //loc points to the leftmost $M_{right}$ symbol
        if (right < n) //not reached the rightmost simple motif yet
          extendRight(right, loc);
        else //a match for rightmost simple motif found
          extendLeft(a, L);
    }//end for
  }//end while
}//end extendRight
extendLeft(v, loc){
  left := v-1;
  while (left >= 1){
    for each x, $i_{left}$ <= x <= $j_{left}$ (starting from $i_{left}$){
      align $M_{left}$ and S such that $M_{left}$[0] is at S[loc-$|M_{left}|$ - x];
      compare $M_{left}$ symbols with the corresponding S bases
      if (mismatch) x++; //consider next gap value
      else //exact match for $M_{left}$ found
        sm_len = sm_len + $|M_{left}|$ + x;
        loc = loc-$|M_{left}|$ - x; //loc points to the leftmost $M_{left}$ symbol
        if (left > 1) //not reached the leftmost simple motif yet
          extendLeft(left, loc);
        else //a match for leftmost simple motif found
          return (SM occurs at S[loc], length = sm_len bases);
    }//end for
  }//end while
}//end extendLeft
```

Fig. 4. EMOS Algorithm

cannot lead to a premature and incorrect conclusion that *SM* does not occur in *S*, i.e., the adopted heuristic does not contravene the 100% recall and precision of our EMOS algorithm.

In Step 2, we find all exact match occurrences of the selected M_a in *S* in two phases. In the first phase (lines 2-10), starting from the ST *root* and the first symbol of M_a, the ST is traversed downwards, matching the corresponding bases for each of the

M_a symbols. At the end of the traversal, the set *ARS* contains the *answer roots* for *all* DNA queries that can be derived from the IUPAC represented M_a and for which at least one occurrence in *S* is found. An answer root for a query is a ST node, which is the root of the subtree that contains all the query answers. In the second phase (lines 11-15), each answer subtree is traversed and from each leaf node, we obtain a starting location of M_a in *S*. Consider our selected M_a = KW, where K = G or T and W = A or T (see Table 2). Initially *ARS* = {*root*} (Fig. 4, line 2). The first iteration of the FOR loop (lines 3-10), converts the first M_a symbol K to the *dna_set* = {G, T}. All outgoing edges of the *root* (Fig. 2) are examined for a match between their label and the bases in the *dna_set*, and as a result at the end of this iteration *ARS* = {node 2, node 4}. For the second M_a symbol, *dna_set* = {A, T} and after considering the outgoing edges of each of the *ARS* nodes, the set *ARS* is updated to {node 10, node 14}. Since all M_a symbols are processed, the second phase starts. The answer subtree rooted in node 10 has two leaf nodes – nodes 12 and 13, representing starting locations *S*[1] and *S*[3], which are added to the M_a(*all*) set. Similarly, answer root node 14 has one leaf node in its subtree (node 14 itself), representing starting location *S*[7], also added to the M_a(*all*). So, at the end of the second phase (also the end of Step 2), the starting locations of all exact matches of M_a in *S* are in the set M_a(*all*) = {1, 3, 7}.

In Step 3, starting with the first verified simple motif (M_a) as an anchor, the algorithm iteratively explores the right adjacent simple motifs (Fig. 4, function *extendRight*). If for a particular combination of gap values, exact matches for all simple motifs to the right of M_a are found, then the algorithm explores iteratively the M_a's left adjacent simple motifs in a similar manner (i.e., function *extendLeft*). If for a particular combination of gap values, exact matches for all simple motifs to the left of M_a are found, then an exact match for the whole *SM* is found, and its starting location in *S* and its length are returned to the user. In the context of our ongoing example, consider the first location of the M_a(*all*) set, *L* = 1. In Step 3, the algorithm sets *sm_len* = |M_a| = 2, and calls the function *extendRight* with *a* = 2 and *L* = 1 (line 19). For *L* = 1, EMOS finds two occurrences of *SM* in *S*, one of size 7 bases, and one of

```
extendRight(2,1):
  right := 2+1 = 3,
  x = 2 => compare M3[0] = Y with S[1+2+2] = S[5] = G - no match
  x = 3 => compare M3[0] = Y with S[1+2+3] = S[6] = C - match,
    all M3 symbols matched, sm_len=2+1+3=6, M3 is rightmost simple motif
    extendLeft(2,1):
      left := 2-1 = 1,
      x = -1 => compare M1[0] = W with S[1-2-(-1)] = S[0] = A - match
                compare M1[1] = N with S[1] = G - match
                all M1 symbols matched, sm_len = 6+2+(-1)=7,
                M1 is leftmost simple motif
                print: SM occurs at S[0], length = 7 bases
      x = 0  => compare M1[0] = R with S[1-2-0] = S[-1]:terminate extendLeft;
  x = 4 => compare M3[0] = Y with S[1+2+4] = S[7] = T - match,
    all M3 symbols matched, sm_len=2+1+4=7, M3 is rightmost simple motif
    extendLeft(2,1):
      left := 2-1 = 1,
      x = -1 => compare M1[0] = W with S[1-2-(-1)] = S[0] = A - match
                compare M1[1] = N with S[1] = G - match
                all M1 symbols matched, sm_len = 7+2+(-1)=8,
                M1 is leftmost simple motif
                print: SM occurs at S[0], length = 8 bases
      x = 0  => compare M1[0] = R with S[1-2-0] = S[-1]:terminate extendLeft;
end extendRight(2,1);
```

Fig. 5. Partial illustration of Step 3 (EMOS)

size 8 bases, both starting at *S*[0], as shown in Figure 5. Overall, there are 6 exact match occurrences of *SM* in *S*: four starting at *S*[0] with gaps sizes {-1,3}, {-1,4}, {1,2}, and {1,3}, respectively; and two starting at *S*[2] with gaps sizes {-1,2} and {-1,3}.

5 Experiments and Results

We conducted numerous experiments to evaluate the performance of our EMOS algorithm and compare it to SMOTIF1, the most efficient alternative solution. In our experiments we used chromosomes Y, 20, 10, and 2 of *Homo sapiens* (build number 36, version 2 [16]) as sequences to be searched. From each of the sequences we removed the still unknown nucleotides (represented by N), which leads to sequence sizes of 26, 60, 132, and 238 million bases (Mb), respectively.

We used a standard 32-bit desktop computer with Intel Pentium 4 @ 3GHz, 2 GB RAM, 300 GB HDD, 2 MB L2 cache, running Linux kernel 2.6. All I/O operations are unbuffered at the OS level. All times reported are real times.

In our experiments we used the source code of SMOTIF1 available at [17]. Its current implementation is limited to processing only a single structured motif query at a time. While our implementation of EMOS accepts a set of structured motifs as input, in our experiments we pose only a single query at a time to both programs. This makes the comparison fair to SMOTIF1, since otherwise EMOS is much faster. EMOS is implemented in C and is available through the WEB, as part of our FASST (Fast and Scalable Search Tool for biological sequence data) project [13].

5.1 Comparison with SMOTIF1

In our first set of experiments we compare the performance of the two algorithms using the same collection of random structured motifs used in [1], which was provided to us by the authors. The set contains 100 random structured motifs over the IUPAC alphabet. Each structured motif consists of 3 to 8 simple motifs of length between 5 and 10 symbols. The number of simple motifs and their lengths are selected uniformly at random within these ranges. The gaps between simple motifs are chosen as a random subinterval of [-5, 100]. Recall that negative values for the gap size allow for partially overlapping simple motifs. The measured search times are accumulated for all 100 queries and reported in Table 3. Our results indicate that EMOS performs 5 to 6 times faster than SMOTIF1, due to reasons discussed in detail in Section 5.2.

In our second set of experiments we compare the performance of the two algorithms for real-life structured motifs. We use the same 4 real-life motifs as in [1], which are obtained by a multiple alignment of 36 *A. thaliana* LTR retrotransposons.

Table 3. Cumulative Search Times for 100 random structured motifs

Sequence	Sequence Size	SMOTIF1 (sec)	EMOS (sec)	Speedup
chr_Y	26 Mb	422	88	**4.8**
chr_20	60 Mb	1,072	184	**5.8**
chr_10	132 Mb	2,205	371	**5.9**
chr_2	238 Mb	3,859	645	**6.0**

The motifs are shown in Figure 6 (where *ZZ* stands for Zhang & Zaki) and the measured search times are presented in Figure 7. Again, for structured motifs with practical selectivity, EMOS significantly outperforms SMOTIF1, providing up to 8 times faster search (i.e., *ZZ4* in chr_2). For *ZZ3*, due to the low selectivity power of all its simple motifs (there are almost 4 million occurrences of *ZZ3* in chr_2), the advantage of selecting a suitable simple motif M_a which will reduce the work done in Step 3 cannot be achieved, and EMOS exhibits performance similar to SMOTIF1.

ZZ1 = HNGTNYDNHDNBTNNDNA[0,3]YNHTNYRHGGNBTNAR[0,2]ARDBNBH
ZZ2 = TNVRNKAYNKNVVNDV[9,11]HNRR[6,8]YDNNVNNV[9,13]HB[4,5]TNNNNRBNYDBDNNRR
ZZ3 = DNNNNDRYW[2,5]DS[6,7]HMM[1,2]TNDB
ZZ4 = DBNNNND[48,102]KRRYMYNNNMRNHYNDVNYAYVH[7,10]VNNNYNNND[34,63]WD[2,8]KNNH[3,5]
. VNDDRNNNNNNHVNNNNNNNHHH

Fig. 6. Real-life structured motifs [1]

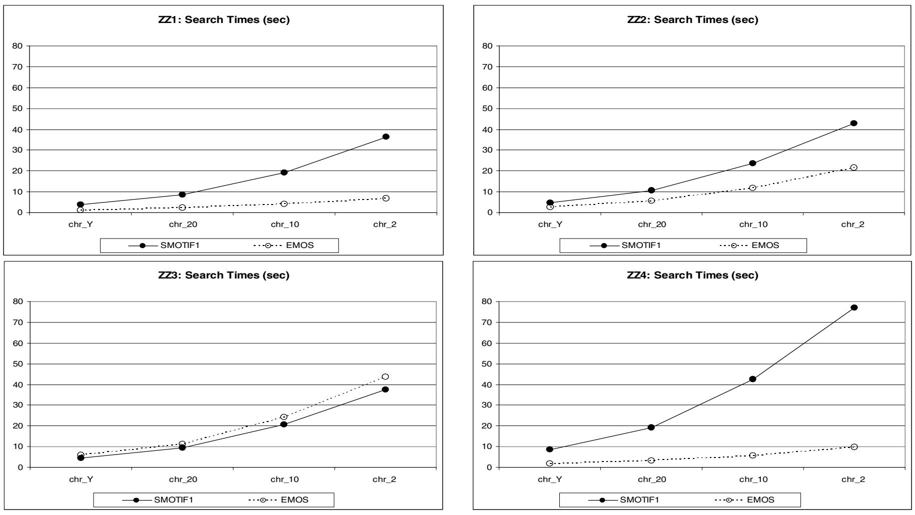


Fig. 7. Search Times for real-life structured motifs

5.2 Sources of EMOS Speedup

There are two major sources for the improvement achieved by EMOS. First, our approach of finding exact matches for a single simple motif and then aligning the structured motif with the sequence and performing character comparisons is more efficient compared to the approach taken by SMOTIF1, which is based on extracting the full post-lists for all motif symbols and then performing positional joins on them. Second, the heuristic used for selecting a suitable simple motif, although imperfect, significantly reduces the amount of work done by our algorithm, thus further improving its search time performance.

To evaluate the first source of improvement, in our third set of experiments, we modified EMOS so that a random simple motif is selected as M_a (i.e., EMOS_random).

In this way its performance is independent of how suitable the selected M_a is. The measured search times are presented in Table 4. As can be seen from the results, our approach is 2.2 to 2.6 times more efficient than the alternative. This is explained by the fact that in the first step SMOTIF1 considers all possible locations in the sequence, i.e., the full post-lists of all *SM* symbols. In the second step, it explores these post-lists by performing positional joins. On the other hand, by first performing exact match search for a single simple motif, EMOS greatly reduces the number of sequence locations (usually less than 10% of the sequence size) that has to be further examined in Step 3 by aligning the *SM* with *S* at these anchor locations.

Table 4. Cumulative Search Times for 100 random structured motifs (random M_a)

Sequence	SMOTIF1 (sec)	EMOS_random (sec)	Speedup
chr_Y	422	174	**2.4**
chr_20	1,072	410	**2.6**
chr_10	2,205	935	**2.4**
chr_2	3,859	1,728	**2.2**

The second source of the improvement provided by EMOS is based on selecting a *suitable* simple motif as M_a. The goal is to choose M_a in such a way, that the number of anchor positions that has to be examined in Step 3 is the smallest. We choose the M_a by computing the selectivity powers of all simple motifs as discussed in Section 4. Table 5 shows the search time improvement obtained by selecting and using a suitable M_a compared to using a random simple motif as M_a. Our experimental results indicate that by selecting a suitable M_a, the number of sequence locations that have to be examined in Step 3 of EMOS is further reduced by a factor of 10 (i.e., now anchor locations are less than 1% of the sequence size). As a result, an additional search time improvement of 2 to 2.7 times is obtained.

Table 5. 100 random structured motifs: Random versus Suitable M_a

Sequence	EMOS_random (sec)	EMOS (sec)	Speedup
chr_Y	174	88	**2.0**
chr_20	410	184	**2.2**
chr_10	935	371	**2.5**
chr_2	1,728	645	**2.7**

A related interesting question is to evaluate the performance of our heuristic used for selecting M_a. As already illustrated, the adopted technique for calculating the selectivity power is fast and provides significant search time improvement (Table 5), but is it optimal as an estimator of the actual number of exact matches of M_a in *S*? In our last set of experiments, using the 100 random structured motifs, we study the accuracy of the estimator. We compute the estimation percentage error $E = (num\_act - num\_est) / num\_est$, where *num_est* is the number of matches estimated by our heuristic and *num_act* is the actual number of matches found by searching the sequence. The sample distribution of the estimation error is approximately normal. Our heuristic slightly overestimates the number of actual exact matches (mean error = -5%). However, the standard deviation of the error is relatively high (50%), reflecting that in some cases the estimator performs poorly. This is due to our assumption of uniform distribution of the DNA bases in real-life sequences. One way to relax this assumption and to provide more accurate estimation would be to read the sequence content as part of the preprocessing step and obtain the actual base distribution, which is a topic we currently study.

Now, we revisit our assumption of STTD64 index being already created. This is a fair assumption, since the content of biological sequences is relatively stable and there are numerous other applications that benefit from the index. However, if the index has not been created in advance, it can be constructed for around 900 seconds for chr_2, for example [12]. The search times for chr_2 in Table 3 indicate that the cost of creating the STTD64 index will be amortized after around 30 structured motif queries.

Last, we remark that on average 70% of the EMOS search time is spent reading the data sequence from disk to memory. Of the remaining 30%, selecting a suitable simple motif (Step 1) takes less than 1%, using the STTD64 index to obtain all exact match occurrences of the anchor motif (Step 2) requires around 10%, and extending the structured motif at the anchor positions (Step 3) accounts for the last 20%. The search time distribution explains why EMOS is much faster when searching in a particular sequence for a set of motif queries, as opposed to a single one at a time. It also points out that in order to further improve the search time of EMOS, a more efficient implementation of Step 3 will be beneficial. We plan to investigate this issue, by considering alternative approaches, such as the positional join approach of SMOTIF1.

6 Conclusions and Future Work

In this work we study the problem of exact match overlapping structured motif search in DNA sequences. Our main contributions are two. First, the existing solutions to this problem, as a first step, either search for matches of all simple motifs (SMOTIF2 and SMaRTFinder) or do not search for any simple motif at all (SMOTIF1). The drawback of the first approach is that for long sequences and/or several simple motifs that have low information content, the preliminary output may become larger than the main memory and thus render the algorithms incapable of concluding the search. In contrast, our proposed solution performs an index-based search for a single simple motif, by using only the related parts of the disk-resident STTD64 index, thus not requiring extensive RAM space. Another major advantage of our approach is that it significantly reduces the number of sequence locations that has to be examined further, thus avoiding a shortcoming of SMOTIF1, which although based on efficient positional joins, starts with much larger input, i.e., the full post-lists of all structured motif symbols.

Second, in order to further reduce the number of sequence locations to be examined in the last step of our approach, we propose a heuristic for selecting a suitable simple motif for which the index-based search is carried out. A suitable simple motif is such that the number of its occurrences in S is the smallest. We select this anchor simple motif based on its information content, computed using the selectivity power of its symbols. This computation is fast and reduces the sequence locations that have to be examined in the last step of our approach to around 1% of the sequence size.

We have implemented these ideas in the EMOS algorithm (Exact Match, Overlapping Structured motif search) and conducted numerous experiments to evaluate and compare its performance to SMOTIF1, the most efficient known solution. Our experimental results show that for a set of 100 randomly generated structured motifs EMOS provides 5 to 6 times faster search than SMOTIF1 on average. Intuitively, for the cases when all simple motifs of the structured motif are with low information content, EMOS cannot take advantage of the two sources of improvement we introduce. In these cases, it exhibits performance comparable to SMOTIF1.

There are two main areas in which the performance of EMOS can be further improved – implementing its third step in a more efficient way and improving the accuracy of our estimator for selecting a suitable simple motif as an anchor, which we are currently investigating. Also, we plan on extending the algorithm to perform approximate motif search and to accept a profile representation of the structured motif.

Acknowledgements

This work was partially supported by Natural Sciences and Engineering Research Council (NSERC) of Canada, Genome Quebec, and by Concordia University.

References

1. Zhang, Y., Zaki, M.J.: SMOTIF: efficient structured pattern and profile motif search. Algorithms for Molecular Biology, 1–22 (November 2006)
2. McCarthy, E., McDonald, J.: LTR_STRUC: A Novel Search and Identification Program for LTR Retrotransposons. Bioinformatics 19(3), 362–367 (2003)
3. Feschotte, C., Jiang, N., Wessler, S.: Plant transposable elements: where genetics meets genomics. Nature Review Genetics 3(5), 329–341 (2002)
4. Jurka, J., Kapitonov, V., Pavlicek, A., Klonowski, P., Kohany, O., Walichiewicz, J.: Repbase Update, a database of eukaryotic repetitive elements. Cytogenet Genome Res 110(1-4), 462–467 (2005)
5. Policriti, A., Vitacolonna, N., Morgante, M., Zuccolo, A.: Structured Motif Search. In: Int'l Conf. on Research in Computational Molecular Biology, pp. 133–139 (2004)
6. Mehldau, G., Myers, G.: A system for Pattern Matching Applications on Biosequences. Computer Applications in the Biosciences 9(3), 299–314 (1993)
7. Myers, E.: Approximate Matching of Network Expressions with Spacers. J. Comput. Biol. 3(1), 33–51 (1996)
8. Navarro, G., Raffinot, M.: Fast and Simple Character Classes and Bounded Gaps Pattern Matching, with Application to protein Searching. J. Comput. Biol. 10(6), 903–923 (2003)
9. Zaki, M.J.: SPADE: An Efficient Algorithm for Mining Frequent Sequences. Machine Learning Journal 42(1/2), 1–31 (2001)
10. Zaki, M.J.: Sequence Mining in Categorical Domains: Incorporating Constraints. In: ACM Int'l Conf on Information and Knowledge Management, pp. 422–429 (2000)
11. Gusfield, D.: Algorithms on strings, trees and sequences: computer science and computational biology. Cambridge University Press, Cambridge (1997)
12. Halachev, M., Shiri, N., Thamildurai, A.: Efficient and scalable indexing techniques for biological sequence data. In: Hochreiter, S., Wagner, R. (eds.) BIRD 2007. LNCS (LNBI), vol. 4414, pp. 464–479. Springer, Heidelberg (2007)

13. FASST web-interface, http://sepehr.cs.concordia.ca/
14. Giegerich, R., Kurtz, S., Stoye, J.: Efficient implementation of lazy suffix trees. Software – Practice and Experience 33(11), 1035–1049 (2003)
15. Tian, Y., Tata, S., Hankins, R.A., Patel, J.: Practical methods for constructing suffix trees. VLDB Journal 14(3), 281–299 (2005)
16. Human Genome Data, ftp://ftp.ncbi.nih.gov/genomes/H_sapiens/Assembled_chromosomes
17. SMOTIF1 source code, http://www.cs.rpi.edu/~zaki/software/sMotif/

A Discriminative Method for Protein Remote Homology Detection Based on N-nary Profiles

Bin Liu[1], Lei Lin[2], Xiaolong Wang[1,2], Qiwen Dong[2], and Xuan Wang[1]

[1] Harbin Institute of Technology Shenzhen Graduate School, Shenzhen, China
[2] School of Computer Science and Technology, Harbin Institute of Technology, Harbin, China
{bliu, linl, wangxl, qwdong, wangxuan}@insun.hit.edu.cn

Abstract. Protein homology detection is a key problem in computational biology. In this paper, a novel building block for protein called N-nary profile which contains the evolutionary information of protein sequence frequency profiles has been presented. The protein sequence frequency profiles calculated from the multiple sequence alignments outputted by PSI-BLAST are converted into N-nary profiles. Such N-nary profiles are filtered by a feature selection algorithm called chi-square algorithm. The protein sequences are transformed into fixed-dimension feature vectors by the occurrence times of each N-nary profile and then the corresponding vectors are inputted to support vector machine (SVM). The latent semantic analysis (LSA) model, an efficient feature extraction algorithm, is adopted to further improve the performance of this method. When tested on the SCOP 1.53 data set, the prediction performance of N-nary profile method outperforms all compared methods of protein remote homology detection. The ROC50 score is 0.736, which is higher than the current best method for nearly 4 percent.

Keywords: remote homology; N-nary profiles; chi-square algorithm; latent semantic analysis.

1 Introduction

Protein homology detection is one of the most intensively researched problems in bioinformatics, which refers to the detection of structural homology in protein when there is little or no sequence similarity. The aim is to predict structural or functional properties of protein by means of homologies, these properties are important for the classification of proteins into functional and structural classes.

Many powerful methods and algorithms have been proposed to detect homology between proteins. Early methods were based on the pairwise similarities between protein sequences. Among those algorithms, the Smith-Waterman dynamic programming algorithm [1] which finds an optimal score for similarity according to a predefined objective function is among the most successful methods. Some heuristic algorithms, such as BLAST [2] and FASTA [3] trade reduced accuracy for improved efficiency. These methods do not perform well for remote homology detection, for the alignment score falls into a twilight zone when the protein sequences similarity is below 35% at the amino acid level [4]. The later methods challenged this problem by incorporating the family information. These methods are based on a proper representation of protein families and can be split into two groups [5]: generative models for protein families and discriminative classifiers. Generative models which provide a probabilistic measure of association between a new sequence and a particular family, such as profile hidden Markov models (HMM) [6], can be trained iteratively in an unsupervised manner using both positively labeled and unlabeled examples by pulling in close homology and adding them to the positive set [7]. The discriminative algorithms such as Support Vector Machine (SVM) [8] provided state-of-the-art performance with appropriate kernel. In contrast to generative methods, the discriminative algorithms focus on learning a combination of the features that discriminate between the classes. These algorithms are trained in a supervised manner using both the positive and negative samples to establish a discriminative model. The first discriminative method is the SVM-Fisher [9], which represents each protein sequence by a vector of Fisher scores. SVM-pairwise [10] is another successful method, in which each protein sequence is represented as a vector of pairwise similarities to all protein sequences in the training set. Many other SVM-based methods also have been proposed, such as SVM-k-spectrum [11], Mismatch-SVM [12], SVM-I-sites [13], SVM-n-peptide [14], Monomer-dist [5], GPkernel [15], SVM-LA and SVM-SW [16]. A comparison of SVM-based methods has been performed by Saigo *et al.* [17].

Sequence homologs are an important source of information about proteins. Multiple sequence alignments of protein sequences contain much information regarding evolutionary processes. This information can be detected by analyzing the output of PSI-BLAST [18, 19]. Since protein sequence frequency profiles are a richer encoding of protein sequences than the individual sequence, it is of great significance to use such evolutionary information for protein remote homology detection.

In our previous study [20], we have introduced a discriminative method called binary profiles method to use the protein sequence frequency profiles for protein remote homology detection, in which protein sequence frequency profile is converted into binary profiles with a probability threshold. In detail when a given amino acid is lager than the threshold it is converted into an integral value 1, otherwise it is converted into 0. However, this simple method omits a lot of important evolutionary information of the protein sequence frequency profiles. Here, we present a novel method called N-nary profiles method to use the evolutionary information of the protein sequence frequency profiles, which can contain more evolutionary information of the protein sequence frequency profiles than binary profiles method. The protein sequence frequency profiles calculated from the multiple sequence alignments outputted by PSI-BLAST are converted into N-nary profiles. Such N-nary profiles are filtered by a feature selection algorithm called chi-square algorithm. The protein sequences are transformed into fixed-dimension feature vectors by the occurrence times of each N-nary profile and then the corresponding vectors are inputted to support vector machine (SVM). The method is further improved by applying an efficient feature extraction algorithm from natural language processing, namely, LSA model [21]. N-nary profile method has been compared with other seven methods. When tested on the SCOP data set, the prediction performance of the new method outperforms all related methods. In terms of computational efficiency, the LSA approach of the new method outperforms SVM-pairwise [10] and SVM-LA [16].

The rest of this paper is organized as follows. Section 2 introduces the data set and N-nary profiles. Experimental results and discussion are shown in Section 3. Finally, conclusions are drawn in Section 4.

2 Methods and Algorithms

2.1 Data Set

We use a common data set for protein remote homology detection [22] to evaluate the performance of our method. The data set is available at http://www1.cs.columbia.edu/compbio/svm-pairwise. Because this data set [5, 16, 21, 22] has been used by many studies of remote homology detection methods, it can provide good comparability with previous methods. The data set contains 54 families and 4352 proteins from SCOP version 1.53 which are extracted from the Astral database [23] and include no pair with a sequence similarity higher than an E-value of 10^{-25}. Because the PSI-BLAST [18] is unable to generate profiles on short sequences, the protein sequences with lengths less than 30 are removed. For each family, the proteins within the family are taken as positive test samples, and the proteins outside the family but within the same superfamily are taken as positive training samples. Negative samples are selected from outside of the superfamily and are separated into training and test sets.

2.2 Generation of N-nary Profiles

2.2.1 Protein Sequence Frequency Profiles

A protein sequence frequency profile can be represented as matrix ***M***, the dimensions of ***M*** are $L{\times}N$, where L is the length of the protein sequence and N is the number of all standard amino acids which is a constant value of 20. Each element of ***M*** is target frequency which indicates the probability of an amino acid in a specific position of a protein sequences during evolution. The rows of ***M*** are amino acid frequency profiles. For each row the frequencies add up to one. Each column of ***M*** corresponds to one of the 20 standard amino acids. The calculation of target frequency is similar to that implemented in PSI-BLAST [18]. The protein sequence frequency profiles are calculated from the multiple sequence alignments outputted by PSI-BLAST. The parameter values of PSI-BLAST are set to default except for the number of iterations set to 10. The database for PSI-BLAST to search against is nrdb90 database from EBI [24]. A subset of multiple sequence alignments with sequence identity less than 98% is used to calculate the protein sequence frequency profiles. We use the position-based sequence weight method [25] to assign the sequence weight. Formula (1) is used to calculated the pseudo-count for amino acid i.

$$g_i = \sum_{j=1}^{20} f_i * (q_{ij} / p_j) \qquad (1)$$

Where f_i is the observed frequency of amino acid i, p_j is the background frequency of amino acid j, q_{ij} is the score of amino acid i being aligned to amino acid j in BLOSUM62 substitution matrix, which is the default score matrix of PSI-BLAST.

The target frequency is calculated with the pseudo-count as:

$$Q_i = (\alpha f_i + \beta g_i)/(\alpha + \beta) \tag{2}$$

Where β is a free parameter set to a constant value of 10 which is initially used by PSI-BLAST and α is the number of different amino acids in a given column minus one.

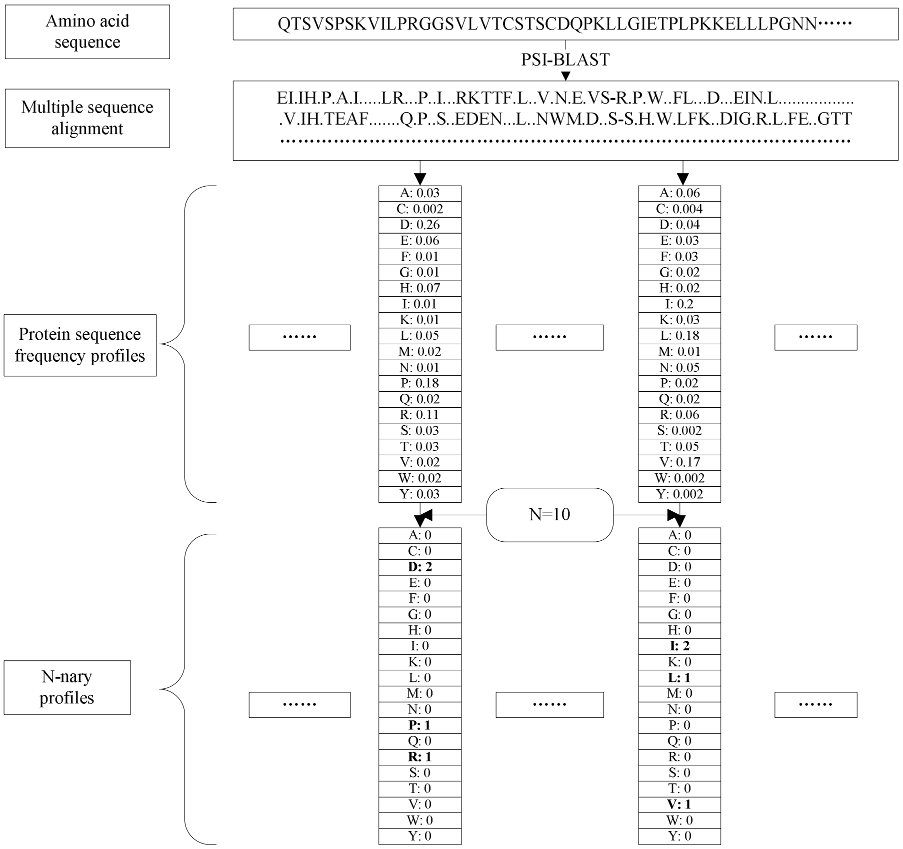


Fig. 1. The flowchart of calculating and converting protein sequence frequency profile. The multiple sequence alignment is obtained by PSI-BLAST. The protein sequence frequency profile is calculated on the multiple sequence alignment and converted to N-nary profiles.

2.2.2 Convert Frequency Profile into N-nary Profiles

Because a protein sequence frequency profile is a matrix of frequencies of 20 standard amino acids, it cannot be used directly. To solve this problem, the protein sequence frequency profiles are converted into N-nary profiles. Details of converting protein sequence frequency profile into N-nary profiles are as follow:

The frequencies for all amino acids belong to interval [0, 1], so this interval can be divided into N equal size intervals. N different integers from 0 to N-1 are used to represent the *N* different equal size intervals respectively (i.e. for *N*=4, interval [0, 1] is divided into 4 equal size intervals: [0, 0.25), [0.25, 0.5), [0.5, 0.75), [0.75, 1] and four integers including 0, 1, 2 and 3 are used to represent the four different equal size intervals respectively). When a given amino acid frequency belongs to a specific interval, the corresponding integer value of the interval is assigned to the amino acid. This process is iterated until each of the 20 standard amino acids is represented as a corresponding integer. By collecting the values of all the 20 standard amino acids, an amino acid frequency profile can be converted into a vector with dimensions of 20, in which each element represents one standard amino acid and can take the value from 0 to *N*-1. These elements discriminate the frequencies of all the 20 standard amino acids. The bigger the value of the element is, the more probable the corresponding amino acid occurs during evolution. We call such vectors N-nary profiles. The above process is iterated until all amino acid frequency profiles in the protein sequence frequency profile are converted into N-nary profiles. In other words, a protein sequence frequency profile can be converted into *k* N-nary profiles, where *k* is the length of protein sequence. Compared with binary profiles method, N-nary profiles method contains more evolutionary information of the protein sequence frequency profiles. The process of generating and converting the protein sequence frequency profile into N-nary profiles is shown in Fig. 1.

2.3 Chi-Square Feature Selection

In theory, the total number of N-nary profiles is N^{20}. In fact, only a small part of the N-nary profiles occur in the protein sequence. However, when *N* increases the total number of N-nary profiles increases rapidly. Furthermore, most machine learning algorithms do not scale well to high-dimensional feature spaces [26]. Thus the so called chi-square feature algorithm [27], one of the most effective feature selection methods in document classification task [28], is used to reduce the dimension of the feature space by removing non-informative and redundant features. The maximum of 8000 N-nary profiles are selected by chi-square feature algorithm. Details of chi-square algorithm are available in a related study [27].

2.4 Construction of SVM Classifiers and Classification

In order to exclude differences owing to particular realizations of the SVM-based learning algorithm and for best comparability with other methods, we employ the publicly available Gist SVM package (http://svm.sdsc.edu) for remote homology detection. The SVM parameters are used by default of the Gist Package except for the kernel function is set as Radius Basis Function (RBF). Radius Basis Function is as follow:

$$k^{\wedge}(X,Y) = e^{-((k(X,X)-2k(X,Y)+k(Y,Y))/2\sigma^2)} \tag{3}$$

Where the width σ is the median Euclidean distance from any positive training sample to the nearest negative sample and *k*(.,.) is the normalized base kernel acting as a similarity score between the pair of input vectors *X* and *Y*; i.e.,

$$k(X,Y)=\frac{X\cdot Y}{\sqrt{(X\cdot X)(Y\cdot Y)}} \tag{4}$$

The training protein sequences are transformed into fixed-dimension feature vectors by the occurrence times of each N-nary profile and then the vectors are inputted to SVM to construct the classifier for a specified family. The test sequences are vectorized in the same way as the training sequences and fed into the classifier constructed for a given family to make separation between the homology and non-homology samples. The SVM assigns each protein in the test set a discriminative score which indicates a predicted level of homology. The proteins with discriminative scores higher than a threshold are classified as homologs and the others as non-homologs. The above process is iterated until each family is tested.

2.5 Latent Semantic Analysis

The Latent Semantic Analysis (LSA) [21] is adopted to get better performances. Recently, latent semantic analysis (LSA) was introduced in computational biology, it was used to predict the secondary structure of protein [29] and detect protein remote homology [21]. LSA is used to extract and represent the context-usage meaning of words by statistical computations applied to a large corpus of text [30]. The process of LSA is as follow:

Firstly, a word-document matrix $\boldsymbol{W}$ of co-occurrences between words and documents is constructed. The elements of $\boldsymbol{W}$ indict the number of times each word appears in each document, so the dimensions of $\boldsymbol{W}$ is M×N, where M is the total number of words and N is the number of given documents. Each word count is normalized to compensate the differences in document lengths and overall counts of different words in the document and collection [30]. Secondly, singular value decomposition is performed on the word-document matrix $\boldsymbol{W}$, as follows:

$$W=USV^{T} \tag{5}$$

Where $\boldsymbol{U}$ is left singular matrix with dimensions (M×K), K is the total ranks of $\boldsymbol{W}$, $\boldsymbol{S}$ is diagonal matrix of singular values with dimensions (K×K), and $\boldsymbol{V}$ is right singular matrix with dimensions (N×K). Thirdly, the top R (R<<Min (M, N)) dimensions are selected for further processing. The dimensions of reduced matrices $\boldsymbol{U}$, $\boldsymbol{S}$ and $\boldsymbol{V}$ are M×R, R×R and N×R respectively. More details of latent semantic analysis are available in [21].

In this paper, values of R in the range [50, 500] are selected. Five building blocks are treated as the "word", including N-grams [11], patterns [27], motifs [31], binary profiles [20] and N-nary profiles. The protein sequences are viewed as the "documents". Through collecting the weight of each word in the documents, the word-document matrix is constructed and then the latent semantic analysis is performed on the matrix to produce the latent semantic representation vectors of protein sequences in order to remove noise and compress data. The latent semantic representation vectors are inputted into SVM to give the final result.

2.6 Evaluation Methodology

Three methods are used to evaluate the quality of the methods: the Receiver Operating Characteristic (ROC) scores [32], the ROC50 scores [32], and the Median Rate of False Positive (M-RFP) scores [9]. A ROC score is the normalized area under a curve that plots true positives against false positives for different classification thresholds. A score of 1 denotes perfect separation of positive samples from negative ones, whereas a score of 0 indicates that none of the sequences selected by the algorithm is positive. A ROC50 score is the area under the ROC curve up to the first 50 false positives. The M-RFP score is the fraction of false positives scoring as high as or better than the median score of true positives. The bigger the M-RFP score is, the worse the result is.

3 Results and Discussion

3.1 Comparative Results of Various Methods

Table 1 summarizes the average values of ROC scores, ROC50 scores and M-RFP scores of eight different methods, including PSI-BLAST [18], pairwise [10], LA [16], N-gram [11], pattern [27], motif [31], binary profiles [20], and the present method (N-nary profiles). For a detail setup procedures of these methods for comparison, please refer to two related studies [20, 21].

As shown in the table, our method performs well for *N* from 6 to 12. The detection performance increases significantly for *N* from 2 to 5, since when *N*<6, there are less than 249 N-nary profiles containing limited evolutionary information of protein

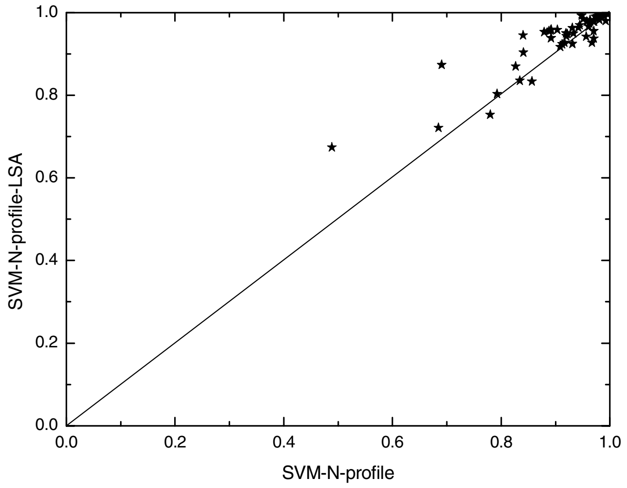


Fig. 2. Family by family comparison of the methods with LSA and those without LSA. The coordinates of each point in the plot are the ROC scores for one SCOP family, obtained by the two methods labeled near the axis.

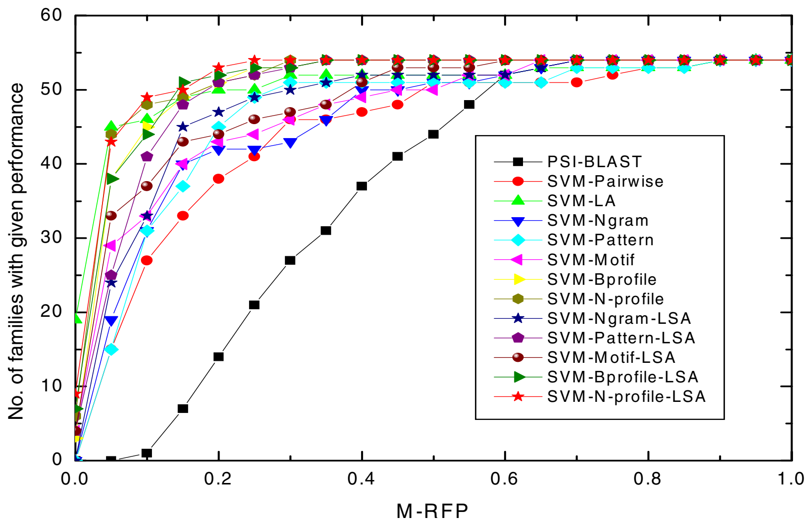


Fig. 3. M-RFP score distribution for different methods. Each series corresponds to one of the homology detection methods described in the text.

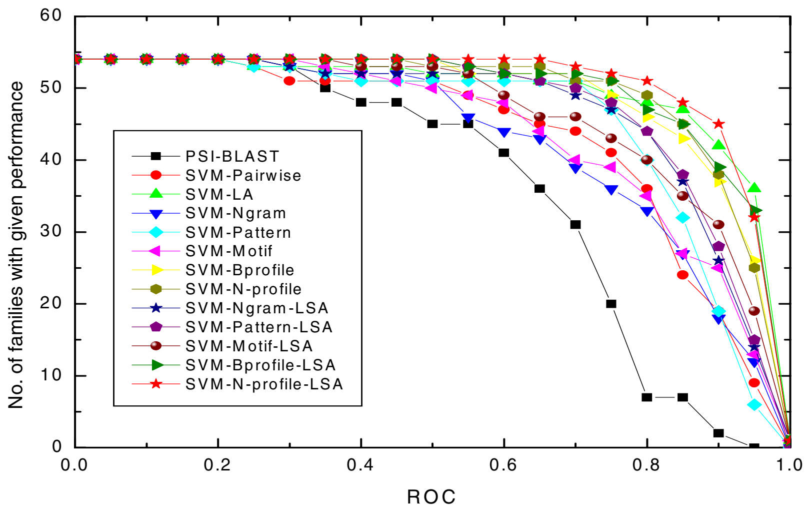


Fig. 4. ROC score distribution for different methods

sequence frequency profiles. When *N*>12, 8000 N-nary profiles are selected by chi-square feature algorithm from more than 28407 N-nary profiles which maybe contain much noise, this is possibly the reason for decreasing in the detection performance.

The latent semantic analysis model is adopted to further improve the performance of this method. Fig. 2 shows the family-by-family comparison of the ROC scores

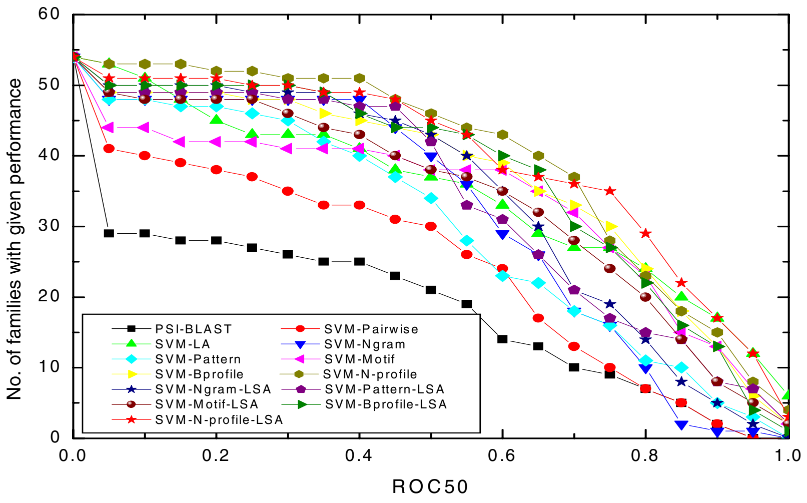


Fig. 5. ROC50 score distribution for different methods

between the method with LSA and without LSA when the N-nary profiles are taken as the basic building blocks. Each point on the graph corresponds to one of the tested SCOP families. When the families are in the left-upper area, it means that the method labeled by y-axis outperforms the method labeled by x-axis on this family. Obviously, when the N-nary profiles are taken as the basic building blocks, the method with LSA can significantly outperform the method without LSA. Such conclusion is also suitable for other building blocks: N-grams, patterns, motifs and binary profiles [20, 21].

In order to further investigate the results, these methods are compared by their relative performances by plotting the number of families for a given method above a given threshold ROC scores, M-RFP score, or ROC50 score ranging from 0 to 1. Fig. 3-5 compare the performance of the new method for *N*=11 with other methods. In each graph, a higher curve corresponds to more accurate performance.

SVM-Ngram, SVM-Pattern, SVM-Motif, SVM-Bprofile and SVM-N-profile refer to the SVM-based methods on the five building blocks: N-grams, patterns, motifs, binary profiles and N-nary profiles respectively. The methods with LSA suffix refer to the corresponding method after latent semantic analysis. Results of SVM-LA method are taken from (Saigo et al. 2004, Bioinformatics, 20: 1682-1689). Bold numbers indicate the best results in each column.

As show in Fig. 3, SVM-LA performs well for M-RFP score thresholds less than 0.05 with a higher number of included families, while both N-nary-profile-based methods show an improved performance for a increasing score threshold, especially for M-RFP scores between 0.05 and 0.1. Fig. 4 shows SVM-LA performs well for ROC score thresholds bigger than 0.95, while SVM-N-profile-LSA method outperforms the compared methods for a decreasing score threshold. Fig. 5 shows that SVM-N-profile method outperform other methods for ROC50 score thresholds between 0 and 0.7 and SVM-N-profile-LSA method outperforms compared methods for

Table 1. Average ROC, ROC50 and M-RFP scores over all families for different methods

Average ROC, ROC50 and M-RFP scores			
Methods	ROC	ROC50	M-RFP
PSI-BLAST	0.6754	0.330	0.3253
SVM-Pairwise	0.8259	0.446	0.1173
SVM-LA (β = 0.5)	**0.9250**	0.649	0.0541
SVM-Ngram	0.7914	0.584	0.1441
SVM-Pattern	0.8354	0.589	0.1349
SVM-Motif	0.8136	0.616	0.1246
SVM-Bprofile (*Ph* = 0.13)	0.9032	0.681	0.0682
SVM-N-profile			
$N = 2$	0.5081	0.276	0.5446
$N = 3$	0.7718	0.534	0.1441
$N = 4$	0.8369	0.652	0.0747
$N = 5$	0.8556	0.701	0.0831
$N = 6$	0.8886	0.699	0.0613
$N = 7$	0.8908	0.695	0.0626
$N = 8$	0.8939	0.645	0.0625
$N = 9$	0.8883	0.665	0.0712
$N = 10$	0.8801	0.636	0.0780
$N = 11$	0.9151	**0.733**	**0.0419**
$N = 12$	0.8928	0.698	0.0614
$N = 13$	0.8716	0.666	0.0743
$N = 14$	0.8517	0.648	0.1068
$N = 15$	0.8520	0.639	0.0915
$N = 16$	0.8397	0.578	0.1012
SVM-Ngram-LSA	0.8595	0.628	0.1017
SVM-Pattern-LSA	0.8789	0.626	0.0703
SVM-Motif-LSA	0.8592	0.628	0.0995
SVM-Bprofile-LSA(*Ph*=0.13)	0.9210	0.698	0.0459
SVM-N-profile-LSA(N =11)	**0.9402**	**0.736**	**0.0327**

score thresholds between 0.7 and 0.9, which demonstrates LSA approach can improve the performance for a increasing ROC50 score threshold with a higher number of included families.

In summary, table 1 and Fig. 3-5 demonstrate that the SVM-N-profile-LSA method significantly outperforms all other given methods. Especially, the ROC50 score of SVM-N-profile-LSA method is 0.736, which is higher than the current best method for nearly 4 percent. When the LSA model is not used, the SVM-N-profile method is highly comparable with SVM-LA method and outperforms other methods. The PSI-BLAST [18] method that is based on search techniques and sequence alignment gets the lowest performance. The discriminative methods, SVM-LA based on string alignment kernels and SVM-Pairwise [10] achieve improved performances. The performances of the SVM methods based on N-gram, Pattern, Motif and Binary profiles are lower than that of SVM-LA. When the LSA model is used, the SVM methods based on the five basic building blocks get better performance. The

SVM-LA method outperforms many other methods such as Mismatch-SVM [12] and SVM-Fisher [9] as well as FPS [33] and SAM [34], so SVM-LA method is one of state-of-the-art methods. Because the N-nary profile method outperforms SVM-LA, it is an efficient feature representation scheme of protein sequence for remote homology detection.

3.2 Computational Efficiency

One important aspect of any homology detection method is its computational efficiency. In this regard, although the training times of LSA-based methods are longer than those of the SVM-based method, the testing times of LSA-based methods are much shorter than those of the SVM-based method. Compared with other given methods, LSA approaches are better than SVM-pairwise and SVM-LA, but a little worse than the methods without LSA and PSI-BLAST [21].

The time complexity of running PSI-BLAST is O (nN), where N is the size of the database. In the current situation, N is approximately equal to nl, where n is the number of training examples and l is the length of the longest training sequence. The important steps in any SVM-based method are the vectorization step and optimization step. The vectorization step has a complexity of O (n^2l^2) in SVM-pairwise. The time complexity of calculation of LA-ekm kernel matrix is same as that of SVM- pairwise [16]. The time complexity of the vectorization step of the SVM-N-profile method is O (nml), where m is the total number of the words. The SVD process of LSA method roughly takes time of O (nmt), where t is the minimum of n and m. The optimization steps of SVM-based methods take time of O (n^2p), where p is the length of the latent semantic representation vector. For SVM-pairwise, p is equal to n, resulting in a total time of O (n^3). In the LSA method, p is equal to R, while in the method without LSA, p is equal to m. Since $R << \mathrm{Min}(n, m)$, The SVM optimization step of LSA method is much faster than SVM-pairwise and SVM-LA.

4 Conclusion

In this paper, we present a simple and efficient building block to represent the protein sequences, which contains evolutionary information of protein sequence frequency profiles. The new method has been successfully applied for remote protein homology detection task and tested on the SCOP 1.53 data set. In comparison with the existing methods, the present method significantly outperforms all other related methods. On the other hand, the computational efficiency of the LSA approach of the new method is better then SVM-pairwise and SVM-LA. Further work will aim at applying the new building block for protein to other protein classification tasks.

Acknowledgments. The authors would like to thank Minghui Li for his helpful discussions. Financial support is provided by the National Natural Science Foundation of China (60673019 and 60435020) and the 863 High Technology Research and Development Programme of China (2006AA01Z197).

References

[1] Smith, T.F., Waterman, M.S.: Identification of Common Molecular Subsequences. J. Mol. Biol. 147, 195–197 (1981)

[2] Altschul, S.F., Gish, W., Miller, W., Myers, E.W., Lipman, D.J.: Basic Local Alignment Search Tool. J. Mol. Biol. 215, 403–410 (1990)

[3] Pearson, W.R.: Rapid and Sensitive Sequence Comparison with Fastp and Fasta. Methods Enzymol. 183, 63–98 (1990)

[4] Rost, B.: Twilight zone of protein sequence alignments. Protein Eng. 12, 85–94 (1999)

[5] Thomas, L.: Remote homology detection based on oligomer distances. Bioinformatics 22, 2224–2231 (2006)

[6] Karplus, K., Barrett, C., Hughey, R.: Hidden Markov Models for Detecting Remote Protein Homologies. Bioinformatics 14, 846–856 (1998)

[7] Qian, B., Goldstein, R.A.: Performance of an Iterated T-Hmm for Homology Detection. Bioinformatics 20, 2175–2180 (2004)

[8] Vapnik, V.N.: Statistical Learning Theory. New York (1998)

[9] Jaakkola, T., Diekhans, M., Haussler, D.: A Discriminative Framework for Detecting Remote Protein Homologies. J. Comput. Biol. 7, 95–114 (2000)

[10] Li, L., Noble, W.S.: Combining Pairwise Sequence Similarity and Support Vector Machines for Detecting Remote Protein Evolutionary and Structural Relationships. J. Comput. Biol. 10, 857–868 (2003)

[11] Leslie, C., Eskin, E., Noble, W.S.: The Spectrum Kernel: A String Kernel for svm Protein Classification. In: Pacific Symposium on Biocomputing, pp. 566–575 (2002)

[12] Leslie, C., Eskin, E., Cohen, A., Weston, J., Noble, S.W.: Mismatch String Kernels for Discriminative Protein Classification. Bioinformatics 20, 467–476 (2004)

[13] Hou, Y., Hsu, W., Lee, M.L., Bystroff, C.: Efficient Remote Homology Detection Using Local Structure. Bioinformatics 19, 2294–2301 (2003)

[14] Ogul, H., Mumcuoglu, E.: A discriminative method for remote homology detection based on n-peptide compositions with reduced amino acid alphabets. BioSystems 87, 75–81 (2007)

[15] Håndstad, T., Hestnes, A.J., Sætrom, P.: Motif kernel generated by genetic programming improves remote homology and fold detection. BMC Bioinformatics 8, 23 (2007)

[16] Saigo, H., Vert, J.P., Ueda, N., Akutsu, T.: Protein Homology Detection Using String Alignment Kernels. Bioinformatics 20, 1682–1689 (2004)

[17] Saigo, H., Vert, J.P., Akutsu, T., Ueda, N.: Comparison of Svm-Based Methods for Remote Homology Detection. Genome Informatics 13, 396–397 (2002)

[18] Altschul, S.F., Madden, T.L., Schaffer, A.A., Zhang, J.H., Zhang, Z., Miller, W., Lipman, D.J.: Gapped Blast and Psi-Blast: A New Generation of Protein Database Search Programs. Nucleic Acids Research 25, 3389–3402 (1997)

[19] Dowd, S.E., Zaragoza, J., Rodriguez, J.R., Oliver, M.J., Payton, P.R.: Windows.Net Network Distributed Basic Local Alignment Search Toolkit (W.Nd-Blast). BMC Bioinformatics. 6, 93 (2005)

[20] Dong, Q.W., Lin, L., Wang, X.L.: Protein Remote Homology Detection Based on Binary Profiles. In: Hochreiter, S., Wagner, R. (eds.) BIRD 2007. LNCS (LNBI), vol. 4414, pp. 212–223. Springer, Heidelberg (2007)

[21] Dong, Q.W., Wang, X.L., Lin, L.: Application of Latent Semantic Analysis to Protein Remote Homology Detection. Bioinformatics 22, 285–290 (2006)

[22] Liao, L., Noble, W.S.: Combining pairwise sequence similarity and support vector machines for remote protein homology detection. In: 6th Annual International Conference on Research in Computational Molecular Biology, pp. 225–232 (2002)

[23] Chandonia, J.M., Hon, G., Walker, N.S., Conte, L.L., Koehl, P., Levitt, M., Brenner, S.E.: The astral compendium in 2004. Nucleic acids research 32, 189–192 (2004)

[24] Holm, L., Sander, C.: Removing near-neighbour redundancy from large protein sequence collections. Bioinformatics 14, 423–429 (1998)

[25] Henikoff, S., Henikoff, J.G.: Position-Based Sequence Weights. J. Mol. Biol. 243, 574–578 (1994)

[26] Andreeva, A., Howorth, D., Brenner, S.E., Hubbard, T.J.P., Chothia, C., Murzin, A.G.: Scop Database in 2004: Refinements Integrate Structure and Sequence Family Data. Nucleic Acids Research 32, 226–229 (2004)

[27] Dong, Q.W., Lin, L., Wang, X.L., Li, M.H.: A Pattern-Based svm for Protein Remote Homology Detection. In: 4th international conference on machine learning and cybernetics, GuangZhou, China, pp. 3363–3368 (2005)

[28] Yang, Y., Pedersen, J.A.: A comparative study on feature selection in text categorization. In: 14th international conference on machine learning, San Francisco, USA, pp. 412–420 (1997)

[29] Ganapathiraju, M., et al.: Characterization of protein secondary structure, Application of latent semantic analysis using different vocabularies. IEEE Signal Processing Magazine 21, 78–87 (2004)

[30] Landauer, T.K., Foltz, P.W., Laham, D.: Introduction to Latent Semantic Analysis. Discourse Processes 25, 259–284 (1998)

[31] Ben-Hur, A., Brutlag, D.: Remote homology detection: A motif based approach. Bioinformatics 19(suppl. 1), i26–i33 (2003)

[32] Gribskov, M., Robinson, N.L.: Use of Receiver Operating Characteristic (Roc) Analysis to Evaluate Sequence Matching. Computers and Chemistry 20, 25–33 (1996)

[33] Bailey, T.L., Grundy, W.N.: Classifying Proteins by Family Using the Product of Correlated P-Values. In: 3rd international conference on computational molecular biology (RECOMB 1999), pp. 10–14 (1999)

[34] Krogh, A., Brown, M., Mian, I.S., Sjolander, K., Haussler, D.: Hidden Markov Models in Computational Biology: Applications to Protein Modeling. Journal of Molecular Biology 235, 1501–1531 (1994)

Searching for Supermaximal Repeats in Large DNA Sequences

Chen Na Lian, Mihail Halachev, and Nematollaah Shiri

Dept. of Computer Science & Software Engineering,
Concordia University, Montreal, Quebec, Canada
`{cn_lian, m_halach, shiri}@cse.concordia.ca`

Abstract. We study the problem of finding supermaximal repeats in large DNA sequences. For this, we propose an algorithm called SMR which uses an auxiliary index structure (POL), which is derived from and replaces the suffix tree index STTD64 [1]. The results of our numerous experiments using the 24 human chromosomes data indicate that SMR outperforms the solution provided as part of the Vmatch [2] software tool. In searching for supermaximal repeats of size at least 10 bases, SMR is twice faster than Vmatch; for a minimum length of 25 bases, SMR is 7 times faster; and for repeats of length at least 200, SMR is about 9 times faster. We also study the cost of POL in terms of time and space requirements.

Keywords: DNA sequences, supermaximal repeats, suffix tree, performance.

1 Introduction

Searching for repetitive patterns in biological sequences is a major problem in bioinformatics research [3]. Repetitive substrings occur to a striking extent in an organism genome, especially in higher-order organisms such as eukaryotes. For example, [4] indicates that families of repeated sequences account for about one third of the human genome. Although the role of these repetitive patterns is still mainly unknown, some have been linked to several inherited diseases, and are thought to play a role in major events of evolution [4, 5]. A related concern in the repetitive structures is based on the notion of maximality. A *maximal repeat* in a sequence *S* is a substring that occurs at least twice in *S*, and that cannot be further extended to the left and/or right without destroying it being a repeat. For example, *S* = A**ACGT**CG**ACGT**TA**ACGT**C is a DNA sequence which includes two maximal repeats: ACGT, which occurs three times (starting at positions 1, 7, and 13, shown in boldface), and AACGTC, which occurs twice (at positions 0 and 12, shown with gray background). Some applications may consider other types of repetitive structures [3]. For instance, a *supermaximal* repeat is a maximal repeat that never occurs as a substring of any other maximal repeat. In our example, AACGTC is a supermaximal repeat, while ACGT is not, since it occurs as a substring of AACGTC. Further, providing a required minimum length of supermaximal repeats helps in improving both the search performance and the result quality by filtering out the abundant relatively shorter repeats that mostly occur by chance and are believed not to carry structural or functional information.

Searching for supermaximal repeats is among the basic analysis tasks a biologist may perform for finding repeated patterns in a new DNA sequence. Considering the exponential rate at which new DNA sequences are being acquired, we need more efficient search techniques to find repetitive structures. While recognizing this need, we would like to avoid a common pitfall of providing a separate, standalone technique for each sequence search problem. Our approach has been providing a basis for supporting versatile search tasks for sequence analysis. In this context, our proposed solution here for finding supermaximal repeats (SMR) can be viewed as yet another extension of our previous work on searching in biological sequences, in which we introduced a suffix tree index structure STTD64 [1], and showed that it is efficient and scalable for various search tasks, including exact and approximate (k-mismatch) search, and search for structured motifs. Our solution to SMR problem in this paper is part of our ongoing project FASST, available online at http://sepehr.cs.concordia.ca.

The contributions of this work are two. First, we propose an auxiliary index structure, called parent-of-leaf (POL). POL is derived from and possibly replaces the STTD64 index (when supermaximal repeats of some minimum lengths are only desired). Based on the significantly smaller POL index, we develop the SMR algorithm for supermaximal repeats search. Second, we perform extensive experiments using real-life data and show that SMR outperforms the enhanced suffix array based solution included as part of the Vmatch software tool [2]. We also study the cost of building POL in terms of time and space.

The rest of this paper is organized as follows. Section 2 provides the necessary background and review works related to supermaximal repeats problem. In Section 3, we introduce the POL index as well as its construction algorithm, and present our SMR search algorithm. In Section 4, we present the experiments done and report the results. Concluding remarks and future work are provided in Section 5.

2 Background and Related Work

In this section, we review the STTD64 index representation which is used in this work for constructing the POL index. We then recall the supermaximal search algorithm proposed in [3] and discuss its shortcomings. Last, we review research related to searching for supermaximal repeats.

2.1 STTD64 Index Representation

Given an input sequence S of size n characters, the suffix tree (ST) is a rooted directed tree with exactly n leaves. Each internal node (except the root) has at least two children and each edge is labeled with a nonempty substring of S. No two edges out of a node can have labels which start with the same character. The key feature of the ST is that for any leaf node i, the concatenation of the edge-labels on the path from the root to node i exactly spells out the suffix of S that starts at position i, i.e., $S\,[i..n]$.

To illustrate the STTD64 index, consider an example sequence S = AGAGAG\$, where \$ is used as a terminal symbol to ensure no suffix of S is a prefix of another suffix. A graphical representation of the ST for this sequence is shown in Fig. 1, in which the numbers in squares indicate the order in which the ST nodes are evaluated and recorded. The number below each leaf node shows the starting position of the suffix of *S* represented by this leaf node, and this suffix is encoded by the edge labels on the path from the root to this leaf node.

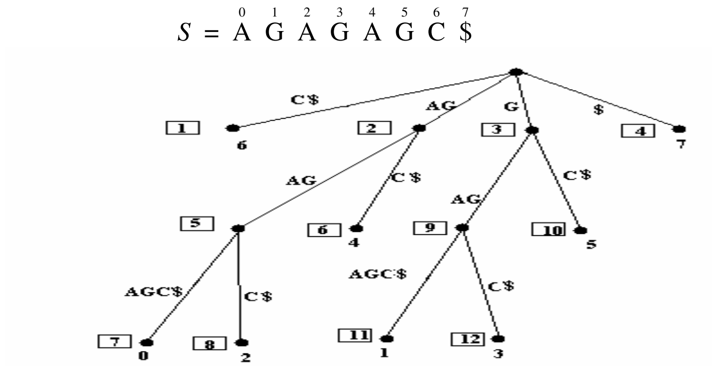
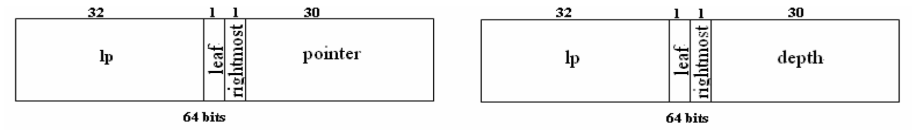


Fig. 1. Suffix Tree (ST) for *S*

In Fig. 2, we show the design of branch and leaf nodes in STTD64, and Fig. 3 shows the actual STTD64 nodes for our example sequence *S*. Each STTD64 node, regardless of being a branch or a leaf node, occupies 64 bits. For each node *v* of either of these two types, we store in the first 32 bits its left pointer value (*lp*). This value is the sum of the leftmost starting location in *S* of the substring encoded from the root of the ST to node *v* plus the *depth* of node *v*. The *depth* of a node is defined as the number of characters from the root to the parent of *v*. For example, for node 9 in Fig. 1, the path from the root to this node encodes the substring "GAG". The leftmost occurrence of this substring in *S* is at location 1. The depth of node 9 is 1, computed as the number of characters from the root to node 3 (i.e., the parent of node 9). Thus, the *lp* value for node 9 which we record in STTD64 is 1+1= 2 (Fig. 3).

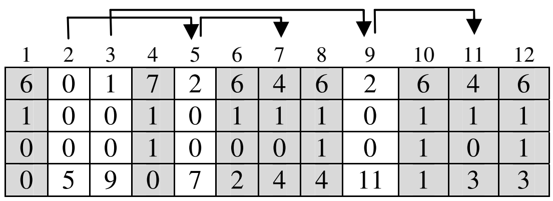


(a) Branch node (b) Leaf node

Fig. 2. STTD64 Representation

For each STTD64 node shown in Fig 2, in the second field of size 1 bit, we store its leaf value. A leaf value 0 indicates that the current node is a branch node, while a leaf value 1 indicates the node is a leaf. For each ST node, in the third field of size 1 bit, we store its rightmost value. Rightmost value 1 indicates that the current node is the rightmost child of its parent. In the fourth field of size 30 bits, we store different information depending on whether the current node is a branch node or a leaf. In case it is a branch node, in the fourth field we store a pointer to the location in the STTD64 index where the first child of this branch node is stored. This is used for traversing the tree. In case of a leaf node, in the fourth field we store the depth of the leaf node.

Fig. 3 shows the complete STTD64 index for our example sequence *S*. The numbers on the top (used for illustration purpose only) indicate the node number in the suffix tree of Fig. 1. The 32 bits *lp* values are shown in the second row and the following two rows indicate leaf bit and rightmost bit, respectively. The pointer/depth values are shown in the last row. For clarity, leaf nodes are shown in gray and branch node pointers are illustrated by the arrows above Fig. 3. For more details on the STTD64 index structure, interested reader is referred to [1].



1	2	3	4	5	6	7	8	9	10	11	12
6	0	1	7	2	6	4	6	2	6	4	6
1	0	0	1	0	1	1	1	0	1	1	1
0	0	0	1	0	0	0	1	0	1	0	1
0	5	9	0	7	2	4	4	11	1	3	3

Fig. 3. STTD64 Representation of ST for Sequence *S*

As already pointed out, our proposed POL index is constructed using the information stored in the STTD64 index. In order to facilitate our description later of the POL index construction algorithm, here we explain how to compute the starting position of a suffix represented by a leaf node. While shown in Fig. 1 by the number below each leaf node, the starting locations in *S* of the suffixes are not explicitly stored in the STTD64 representation (Fig. 3), but rather calculated as follows. For each leaf node *x*, the starting position of the suffix represented by *x* is determined by subtracting the *depth* value of *x* from its *lp* value. For example, the starting location of the suffix that is encoded by the path from the root to node 8 is computed by subtracting the *depth* value of node 8 from its *lp* value, i.e., the starting location of the suffix "AGAGC\$" is 6 – 4 = *S*[2] (see Figures 1 and 3). Storing the required information needed for this calculation in a single node eliminates unnecessary ST traversals, leading to a significant improvement on the number of disk I/O operations during the POL index construction. This explains our choice of STTD64 index representation using which we build the POL index efficiently.

2.2 A Supermaximal Repeats Search Algorithm

In this section, we review a supermaximal repeats search algorithm, taken from Gusfield [3]. The algorithm takes a sequence *S*, its suffix tree index *ST*, and optionally a constraint on the minimum length of the supermaximal repeats as the inputs, and returns the starting positions of all supermaximal repeats in *S* that satisfy the minimum length constraint together with their lengths. For each position *i* in sequence *S*, the character at position *S*[*i*-1] is called the left character of *i*. A node *v* is called left diverse if there are at least two leaves in the suffix tree subtree rooted at *v* which represent suffixes that have different left characters. A left diverse internal node *v* represents a supermaximal repeat if and only if all the children of *v* are leaves, and each suffix has a distinct left character.

Gusfield's algorithm performs two major steps while traversing the ST index. In the first step, it examines all the branch nodes in ST, checking if a particular branch node *v* has only leaf node children and if the length of the possible repeat encoded from the root to *v* is not less than the minimum length required. If yes, the second step is executed, in which it compares the left characters of all the suffixes represented by the children of *v*, i.e., the left diversity check. If successful, the starting positions of the suffixes represented by the leaf nodes of *v* are the starting positions of a supermaximal repeat, and these positions are returned to the user together with the length of the repeat, encoded by the edge labels on the path from the ST root to *v*.

This algorithm has to traverse and examine all ST nodes, even if a minimum length constraint is provided. This results in a significant amount of disk I/Os in order to read into main memory the whole ST index from disk, which even if implemented efficiently [6], is still an order of magnitude larger than the sequence. Our proposed solution addresses this issue and minimizes the disk I/O operations, as explained later.

2.3 Related Work

Recently, the suffix trees (ST) and suffix arrays (SA) received considerable attention as suitable data structures for indexing large biological sequences. One major drawback of these index structures is their considerable size, especially evident for ST. To improve the situation, there have been proposals for compressed suffix arrays and suffix trees representations [7, 8, 9]. Such compressed and compact index representations are of smaller size, allowing them to fit entirely in the main memory available on regular desktop computers. However, this gain in space requirements comes at a cost of less efficient search. As studied in [10], compressed suffix array [7] and FM-index [8] are much slower than suffix tree and suffix array for exact match search. Perhaps this explains why these compressed indexes were not used for finding repeats; we could not find any such work.

There are several software tools for finding repeats, including REPuter [11], RepeatMatch [12], RepeatMasker [13], MaskerAid [14]. REPuter, which is based on suffix tree indexing, is a popular tool for computing various kinds of repeats, including supermaximal repeats, for which it uses a variation of Gusfield's algorithm [3] mentioned above. RepeatMatch also finds repeats based on suffix trees, however it does not support finding supermaximal repeats. RepeatMasker is an implementation of Smith-Waterman-Gotoh algorithm [15]. To find repeats, it performs an exact or approximate search to match the input sequence data against known repetitive patterns already stored in its library. MaskerAid is another implementation of the same approach as RepeatMasker. They both look for repeats of known patterns, however, this is different from the problem we address in this paper, namely finding supermaximal repeats in a sequence without further knowledge.

Vmatch [2] searches for supermaximal repeats based on the enhanced suffix array (ESA) index [16]. Vmatch subsumes REPuter in maximal and supermaximal repeat search for its better space requirements and faster search performance [2]. This is a reason why we compare our work in this paper with Vmatch. Further, Vmatch is a general software tool for solving various search problems in sequence data, including supermaximal repeat search. Since our ongoing project FASST also aims at providing a tool for various types of search tasks in large biological sequences, we consider its comparison with Vmatch on performance of supermaximal repeat search as yet another opportunity to evaluate the two competing multipurpose alternatives.

3 Our Technique for Supermaximal Repeats Search

In the previous section, we discussed Gusfield's algorithm for supermaximal repeats search, which returns the starting positions and the lengths of existing supermaximal repeats in a sequence, by performing sequential access to its entire ST index. However, very often in practice, biologists are interested in repeats of size longer than a particular threshold value. For example, [17] studies different species for repeats of size longer than 200 base pairs. Since the algorithm mentioned above examines the entire ST index, which is about 13 times larger than the data sequence, and cannot take advantage of this additional information on threshold size, it performs a constant and significant amount of disk I/O operations.

The supermaximal repeats search technique (SMR) that we propose here addresses this problem. It uses an index structure, called Parent-Of-Leaf (POL), which is derived from and replaces the STTD64 index. The new POL index is considerably smaller than the STTD64 index. We organize and store the information in POL in such a way that the number of required disk I/O operations is much reduced, resulting in considerably improved search time. Next, we present the structure of the POL index, followed by a description of its construction algorithm. At the end of this section we present our SMR algorithm, which searches for supermaximal repeats using the POL index.

3.1 POL Index Structure

As discussed earlier, each ST node v whose children are all leaf nodes, is a candidate that has to be further examined. If all the suffixes represented by the leaves of v have distinct left characters (i.e., the suffixes are left diverse), then a supermaximal repeat is found, which consists of the common prefix of all the leaf nodes (i.e., the characters on the path from the root to v).

Our POL index is a collection of records related to such candidate nodes v. Each record consists of two parts, the *header* and *data*, as shown in Fig. 4. In the first 27 bits of the header, we store the number of characters on the path from the root of STTD64 index to node v, which represents the length of the potential supermaximal repeat. In the remaining 5 bits of the header, we record y, the number of the leaf children of v, which is equivalent to the number of occurrences of the repeat in the sequence S. The data part of the record for candidate node v contains exactly y blocks, each of size 32 bits. In each block, we store the start location in sequence S at which the suffix represented by a particular leaf of v occurs.

The chosen sizes of the index fields allows for POL indexing of DNA and protein sequences of sizes up to 2^{32} characters (4GB), in which the length of the longest supermaximal repeat is at most 134 million characters. First, the 4 GB limit is due to the fact that each data block, in which we record a sequence location, is of size 32 bits, i.e., the address space is limited to 2^{32}. Next, recall from the ST definition, that no two edges out of a node can have labels which start with the same character. Thus, the number of children of any ST node is bounded by the alphabet size, i.e., 6 for DNA data (including N and the terminal symbol \$) and 24 for proteins. In order for POL index to be applicable for both DNA and protein data, we allocate 5 bits for the second part of the header in which we record the value *y* (at most 32) for the number of leaf children for each candidate node *v*. Last, the remaining 27 bits in the header part are used for recording the length of the repeat, which leads to the limitation on the supermaximal repeat length of 2^{27} characters, i.e., around 134 million nucleotides or amino acids.

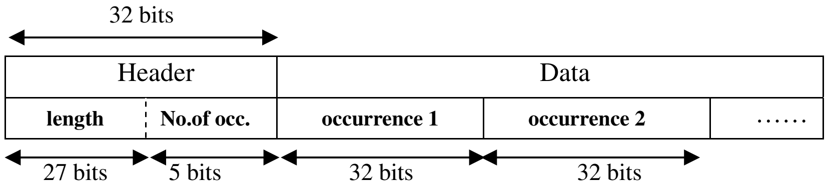


Fig. 4. POL Index Representation

The POL index is implemented as an array of 32 bit blocks. Fig. 5 shows the POL index for our example sequence *S* = AGAGAGC\$. The information about candidate nodes 5 and 9 (in Fig. 1) is recorded respectively in the first 3 and the last 3 POL blocks. The length of the repeat represented by node 5 is 4 nucleotides (i.e., size of "AGAG"), and node 5 has 2 leaf children. Thus, in the header of the record for node 5 (block 1 in Fig. 5), we store values 4 and 2. The next 2 blocks, as indicated by the last 5 bits of the header, are used for recording the starting locations in the sequence at which this repeat occurs. The children of node 5 are nodes 7 and 8, which represent suffixes starting at locations 0 and 2, stored in the second and third POL block. Similarly, since the length of the repeat "GAG" represented by node 9 is 3 nucleotides, and node 9 has 2 leaf children, we store values 3 and 2 in the header of the record for node 9 (block 4 in Fig. 5). The children of node 9 are nodes 11 and 12, which represent suffixes starting at locations 1 and 3, stored in the last 2 POL blocks.

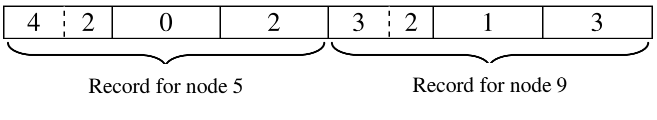


Fig. 5. The POL Index for Sequence *S*

To further improve the support for searching supermaximal repeats of length greater than a particular threshold value, we store the POL records in descending order, with respect to the length of each potential supermaximal repeat (recorded in the first 27 bits of its header). In this way, the SMR algorithm would terminate exploring the POL candidate nodes, once the length of a particular supermaximal candidate becomes smaller than the threshold value.

We remark that in POL we do not record all ST candidate nodes, since otherwise this results in large POL index sizes (comparable to the size of the STTD64 index) and consequently slow POL construction times, without improving much the search performance. In our experimental study, we consider 4 POL index structures for different repeated lengths: POL10, POL25, POL100, and POL200. In POL10 we record all candidate nodes with repeat length of at least 10 nucleotides. This allows for improving the search time for supermaximal repeats of size at least 10 nucleotides. Similarly, POL25, POL100, and POL200 will result in faster search for supermaximal repeats of at least 25, 100, and 200 nucleotides, respectively. In Section 4, we study the cost of these 4 indexes, both in terms of construction time and storage space. However, our implementations of the POL construction algorithm and SMR search algorithm are flexible and allow for creating and using a POL index with repeat lengths that are relevant to the requirements of a particular application.

3.2 POL Index Construction Algorithm

The POL index construction algorithm is presented in Fig. 6. The algorithm takes as input (1) the STTD64 index of the sequence to be searched for supermaximal repeats, and (2) a user-defined minimum repeat length (*len*) of the candidate nodes that are to be recorded. The output is the POL_*len* index, which supports efficient search for supermaximal repeats of size at least *len*.

```
Algorithm POL_len construction (index STTD64, minimum repeat length len)
    //scan all ST nodes;
    node = first ST node;
1.  while (there are unexamined ST nodes)
2.      if (node is a branch) node ++, go to step 1;
3.      else // node is a leaf
4.          if (node.depth < len) node++, go to step 1;
5.          else
6.              while (node is not a rightmost node)
7.                  node++; // move to its sibling
8.                  if (node is branch) node++, go to step 1;
9.                  else
10.                     counter of leaves ++, go to step 6;
11.             end while;
12.             header.length = depth (node);
....            header.number of occurrences = counter;
........        data = starting positions of suffixes;
13.             node++, go to step 1;
14.         end else
15.     end else
16. end while
17. sort records in descending order on header.length;
End
```

Fig. 6. POL Index Construction Algorithm

The construction algorithm traverses the STTD64 index sequentially, examining the leaf nodes. For a leaf node *x*, the algorithm compares its *depth* to *len* (Step 4) to eliminate ineligible nodes. Recall that the *depth* of a node is defined as the number of characters on the path from the root to *the parent of the node*. Thus, this step correctly identifies if a branch node *v* - the parent of *x*, meets the minimum length criterion. In Steps 6 to 11, the algorithm checks if all siblings of *x* are leaf nodes. If positive, a POL record for this candidate is created (Step 12). The POL construction method is completed in Step 17 by sorting the POL records.

Consider the suffix tree in Fig. 1 with *len* = 2. For node 7, which is a leftmost leaf with *depth* > 2, the algorithm performs the while loop in Step 6 to check if node 8 (node 7's right sibling) is a leaf node. Since this is the case and since node 8 is a rightmost child, the algorithm goes to Step 12 to create a record representing the branch node 5 - the parent of leaf nodes 7 and 8 (see Fig. 5). The same steps are executed when node 11 is processed, which results in creating a record in the POL index representing node 9. Last, the candidate node records are sorted based on the repeat length in descending order, but in our example this is already the case. Fig. 5 shows the final POL index for this example.

Assuming that the STTD64 index has been already created, the POL construction algorithm reads the entire STTD64 index in strictly sequential order, which results in $O(n)$ constant time operations, where n is the number of nodes in STTD64. The sorting in Step 17 is done in $O(r \log r)$ time, where r is the number of records in POL. Since r is much less than n, the overall time complexity of POL construction algorithm is $O(n \log n)$.

3.3 Our Supermaximal Repeats Search Algorithm

Our proposed SMR algorithm is presented in Fig. 7. The algorithm takes as input the sequence to be searched, its POL index, and a user-defined parameter *threshold*, which indicates the requested minimum length of the supermaximal repeats. The output of the algorithm contains the starting positions in the sequence and the lengths of all supermaximal repeats of size equal to or greater than the *threshold* value.

As shown in Fig. 7, starting from the first POL record, the SMR algorithm compares the length of each candidate to the *threshold* value (Step 2). If the current record represents a candidate with a length at least equal to threshold, the algorithm reads the repeat occurrence positions from the data blocks of this record (Step 4). In Step 7, the left diversity of these occurrences is examined, and if successful, the discovered supermaximal repeats are produced as output (Step 8). The process of examining POL records continues either to the last POL record or until the length of a candidate becomes smaller than the threshold. Since the records are sorted in descending order, no other supermaximal repeats with length at least equal to threshold may exist and the SMR algorithm correctly terminates.

Let us consider the running sample sequence *S* and a *threshold* value 4. The SMR algorithm starts with reading the first POL record, which represents the candidate repeat at node 5, whose length being 4 characters satisfies the threshold value. Then SMR reads the two subsequent data blocks (as instructed by $y = 2$) and retrieves the two positions $S[0]$ and $S[2]$, where the candidate supermaximal repeat starts. Since left character of the suffix starting at position $S[0]$ of any sequence is different by default from any characters in the sequence, the two suffixes are left diverse and hence SMR outputs the supermaximal repeat found, which is of length 4 and its two occurrences start at positions *S*[0] and *S*[2]. The algorithm then reads the next POL block, which is the header for the candidate repeat at node 9. Since its length is 3, which is less than *threshold,* there is no need to further examine the sorted POL index and the search process terminates.

```
Algorithm SMR(Sequence S, index POL, requested minimum length threshold)
    // read index POL and sequence S into memory
1.  POL record = first;
2.  while (POL.header.length > threshold)
3.      read header.number of occurrences
4.      loop:
5.          find a starting position of an occurrence from POL.data
6.          retrieve left character for the occurrence from sequence S
        end loop
7.      compare left characters of all occurrences
8.      if (all left characters are distinct)
          record a supermaximal repeat of size POL.header.length nucleotides is
          found at starting locations: S[POL.data[1]], ..., S[POL.data[y]])
9.          record++;
10. end while
End
```

Fig. 7. The SMR algorithm using a POL index

The main advantage of our SMR algorithm over the Gusfield's algorithm [3] is that it does not read the entire ST but rather accesses only a considerably smaller index. Using POL, which replaces the ST index, avoids an initial major step of Gusfield's algorithm for finding suitable parent nodes from ST. This leads to considerable decrease in the number of disk I/O operations, which in turn results in significant decrease in search times for SMR. We next study performance of the SMR technique.

4 Experiments and Results

In this section, we first study the cost of construction of POL both in terms of time and storage space. We then experimentally evaluate the performance of our proposed SMR technique on real-life DNA sequences, using four POL indexes of different repeat lengths. We compare our search times with Vmatch.

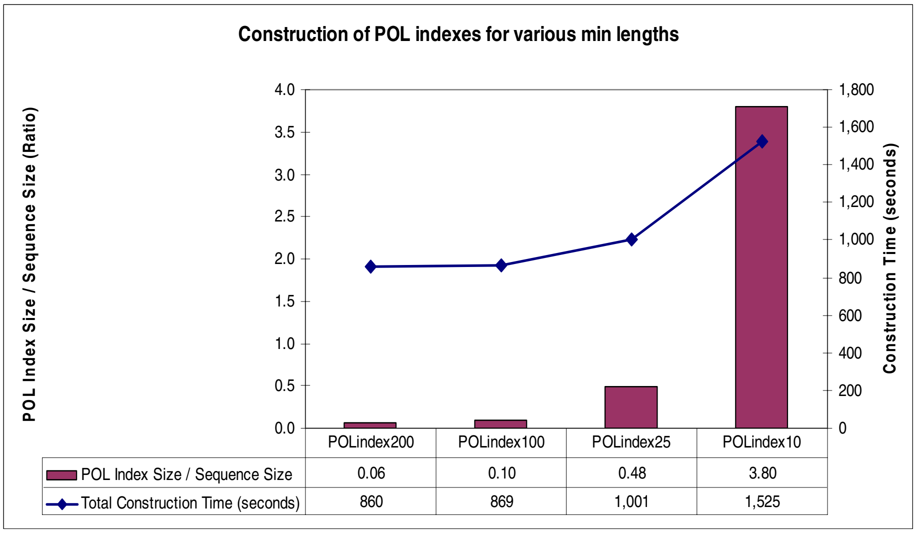
In our experiments, we used the 24 human chromosomes as input sequences to be searched for supermaximal repeats. The data was obtained from [18]. We removed all the unknown nucleotides (indicated by character N), resulting in sequences of size range 26 to 238 million bases.

All experiments are performed on a typical desktop computer with Intel Pentium 4@3GHz, 2GB RAM, 300GB HDD, and 2MB L2 cache, running Linux kernel

2.6.14. The construction and search times reported are real times in seconds (measured using the *time* command in Linux). The POL construction and SMR search algorithms are implemented in C. The SMR search service is available online for experiments and use from the project web site at http://sepehr.cs.concordia.ca.

4.1 POL Index Construction

As discussed in Section 3.1, recording "all" candidate nodes from ST will result in huge POL indexes. Instead, we consider 4 alternative minimum repeat lengths, i.e., 10, 25, 100, and 200 nucleotides, for which we construct the corresponding POL indexes POL10, POL25, POL100, and POL200. Fig. 8 reports the construction times (in seconds) and storage requirements for these four indexes.



	POLindex200	POLindex100	POLindex25	POLindex10
▬ POL Index Size / Sequence Size	0.06	0.10	0.48	3.80
—◆— Total Construction Time (seconds)	860	869	1,001	1,525

Fig. 8. Construction of POL Indexes for Various Minimum Repeat Lengths

The POL construction times in this figure are the sum of the index construction times for all the 24 chromosomes. For the index size, the figure shows the average index size for all the 24 chromosomes. As can be seen, the POL200 index has the fastest construction time and the smallest storage requirements - on average 6% of the sequence size, or 200 times smaller than the STTD64 index. This index records all candidate ST nodes whose repeat length is at least 200 nucleotides. POL200 can be used to improve the search performance of SMR only if the threshold value on the supermaximal repeats length is at least 200 nucleotides. POL100, POL25, and POL10 can be used for smaller threshold values, at the cost of increased construction time and storage space, since decreasing minimum repeat length leads to more candidate nodes to be identified and recorded in POL. For example, POL25 requires additional 140 seconds in order to record 8 times more candidate nodes compared to POL200, but it supports efficient SMR search for threshold value 25 or more nucleotides.

Constructing the POL10 requires almost twice the construction time of POL200 and results in a 60 times larger index (but still 3 times smaller than STTD64), which supports searching for supermaximal repeats of almost all practical sizes.

The choice of an "appropriate" minimum repeat length for the POL index construction is application dependent. Our construction algorithm allows the user to define a value for this parameter that will best fit the needs of a particular application, and provide a suitable trade-off between construction time and storage space on one hand, and search time on the other. However, we note that for the 24 human chromosomes considered, searching for repeats with minimum lengths smaller than 10 nucleotides is not practical in general, as discussed later in Section 4.3.

4.2 SMR Search Performance

In our second set of experiments, we evaluate the search time performance of SMR when using the 4 POL indexes. In these experiments, we used 14 different threshold values for the supermaximal repeats, ranging from 1 to 10,000 nucleotides. If the threshold is smaller than the minimum repeat length in a particular POL index, the SMR algorithm uses the general STTD64 index instead. Fig. 9 reports the measured cumulative search times (for all 24 chromosomes) for the four SMR runs and Vmatch.

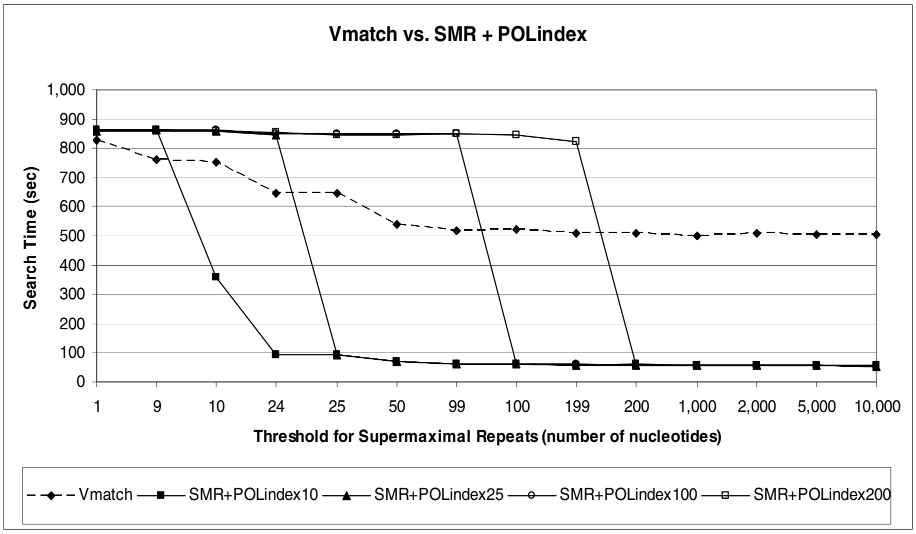


Fig. 9. Vmatch vs. SMR with different POL index

We make the following two important observations. First, if the requested repeats length is greater than the minimum length of two or more POL indexes, the SMR algorithm provides the same search time performance, regardless of which POL index is used. For example, SMR exhibits very similar search times using POL10 and POL25 for threshold values larger than 24 nucleotides. Also, SMR exhibits identical performance using any of the 4 POL indexes for threshold values above 199 nucleotides. This is explained by noting that regardless of which particular POL index is used, the number of candidate nodes that represent supermaximal repeats of desired lengths is the same. Since the records in POL about the candidate nodes are kept in descending order, the SMR algorithm processes the same number of POL records, which leads to identical search times using any of the 4 POL indexes. This observation implies that if a particular search application requires only finding supermaximal repeats of size hundreds or thousands of nucleotides, then POL200 would be suitable index choice due to its fast construction and small storage requirements.

Second, we note that SMR provides significantly faster supermaximal repeats search compared to Vmatch, for many of the cases considered. For example, for a threshold value of 10 nucleotides, SMR with POL10 is 2 times faster than Vmatch; for a threshold value 25, SMR is 7 times faster using either POL10 or POL25. SMR is more than 8 times faster for a threshold value 100 using any of POL10, POL25, or POL100 indexes; and above 9 times faster for threshold values at least 200 nucleotides, using any of the four POL indexes. On the other hand, for threshold values less than 10 nucleotides, the construction of a POL index is not recommended for being too costly. While cases with threshold values less than 10 may not be frequent in practice, our proposed SMR algorithm in such cases can directly use the STTD64 index, resulting in only about 10% slower times compared to Vmatch.

The above results are based on the assumption that a POL index is already constructed and available to SMR. However, an important practical question is: how many requests for computing supermaximal repeats should be posed against a particular sequence so that the cost of POL construction is justified and amortized, thus making SMR preferable to Vmatch solution? For lack of space we do not show our analysis here, but recommend our approach if: (i) there are 2 or more searches with minimum repeat length of 100 nucleotides; (ii) 3 or more searches with length at least 25 nucleotides; or (iii) 4 or more searches with length at least 10 nucleotides are to be performed on the same sequence.

4.3 Number of Supermaximal Repeats

In our last set of experiments, we investigated the number of supermaximal repeats found in the 24 human chromosomes and present the results in Fig. 10.

Increasing the minimum repeat length in the range from 1 to 10 nucleotides does not lead to a significant decrease in the number of supermaximal repeats found (in Fig. 10 these two lines overlap). For such small threshold values, we find almost half a billion repeats in the collection of 24 chromosomes with total size of around 2.8 billion bases. Such answer size contradicts with the idea of supermaximal repeats search, which is intended as a high-level and concise investigation tool for initial analysis of repetitive structures in biological sequences. Further, there is a high possibility that most supermaximal repeats of size less than 10 nucleotides found in sequences containing tens and hundreds million bases occur purely by chance, and thus do not carry any structural or functional information. For these reasons, we believe searching with threshold values less than 10 nucleotides should not be viewed as a primary application of supermaximal repeats search in large DNA sequences, and about a 10% drop in performance of SMR in such cases would not pose a restriction in its use.

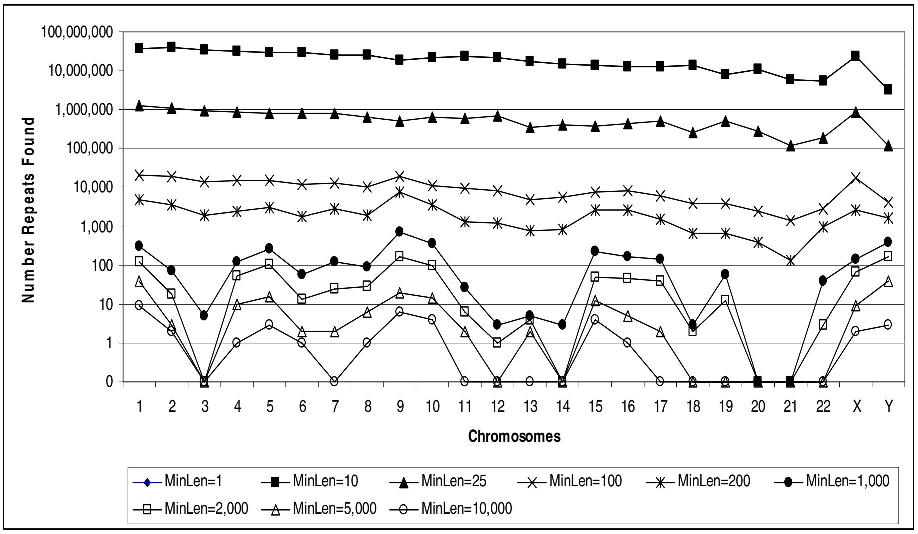


Fig. 10. Occurrences found for various threshold values

5 Conclusions and Future Work

We studied the problem of finding exact supermaximal repeats in large DNA sequences, a major task in bioinformatics applications. Our proposed technique consists of two parts: (i) a new index structure, POL (parent-of-leaf); and (ii) an efficient search algorithm SMR, which uses the POL index. A web-based interface for SMR service is available online at http://sepehr.cs.concordia.ca.

The POL index is derived from and designed to replace a more general, but considerably larger suffix tree based index. Our experiment results indicate that a practical POL index for large DNA sequences, such as the 24 human chromosomes, can be constructed very efficiently by processing the STTD64 index [1] and has 3 to 200 times smaller space requirements. Further, our results show that by using the POL index, our proposed SMR algorithm significantly outperforms the ESA based solution, provided as part of the Vmatch package. The search time improvement achieved by SMR over Vmatch was in the range to 2 to 9 times, when searching for supermaximal repeats of size at least 10 and at least 200 nucleotides, respectively.

Other advantages of our technique are its flexibility and applicability. The POL index can be tailored towards the needs of a specific supermaximal repeats search application. Depending on a desired minimum length for supermaximal repeats in a sequence, the user has control over the amount of information stored during the construction process of the POL index, thus selecting a suitable trade-off between index construction time and storage space on one hand, and the search time performance on the other. Further, a POL index created for a specific minimum repeats length, *len*, can be used for searching repeats of any length equal to or greater than *len*. This feature could be extremely useful in the process of iterative supermaximal repeats search, until a desired trade-off between the number of repeats found and their lengths is reached.

As part of our ongoing research, we are investigating the use of POL index for other types of repetitive structures search, such as maximal repeats, tandem repeats, etc. We also plan to extend our technique for approximate repeat search.

Acknowledgements

This work was partially supported by Natural Sciences and Engineering Research Council (NSERC) of Canada, Genome Quebec, and by Concordia University. We also thank S. Kurtz for providing us the executables of Vmatch.

References

1. Halachev, M., Shiri, N., Thamildurai, A.: Efficient and scalable indexing techniques for biological sequence data. In: Hochreiter, S., Wagner, R. (eds.) BIRD 2007. LNCS (LNBI), vol. 4414, pp. 464–479. Springer, Heidelberg (2007)
2. Vmatch: large scale sequence analysis software, `http://www.vmtach.de`
3. Gusfield, D.: Algorithms on strings, trees and sequences: computer science and computational biology. Cambridge University Press, New York (1997)
4. Korf, B.R.: Human Genetics: A Problem-Based Approach. Blackwell, Boston (2000)
5. Watson, J., Hopkins, N., Roberts, J., Steitz, J., Weiner, A.: Molecular Biology of the Gene, 6th edn. Benjamin-Cummings, Menlo Park (2007)
6. Kurtz, S.: Reducing the space requirement of suffix trees. Software Practice and Experience 29(13), 1149–1171 (1999)
7. Grossi, R., Vitter, J.S.: Compressed Suffix Arrays and Suffix Trees with Applications to Text Indexing and String Matching. SIAM Journal on Computing 35(2), 378–407 (2005)
8. Ferragina, P., Manzini, G.: Opportunistic data structures with applications. In: 41st IEEE Symposium on Foundations of Computer Science, pp. 390–398 (2000)
9. Valimaki, N., Gerlach, W., Dixit, K., Makinen, V.: Compressed suffix tree - a basis for genome-scale sequence analysis. Bioinformatics 23(5), 629–630 (2007)
10. Hon, W.-K., Lam, T.W., Sung, W.-K., Tse, W.-L., Wong, C.-K., Yiu, S.-M.: Practical Aspects of Compressed Suffix Arrays and FM-index in Searching DNA Sequences. In: 6th Workshop on Algorithm Engineering and Experiments, pp. 31–38 (2004)
11. Kurtz, S., Schleiermacher, C.: REPuter: Fast Computation of Maximal Repeats in Complete Genomes. Bioinformatics 15, 426–427 (1999)
12. RepeatMatch, `http://mummer.sourceforge.net/manual/#repeat`
13. RepeatMasker, `http://repeatmasker.org/`
14. Bedell, J.A., Korf, I., Gish, W.: MaskerAid: a Performance Enhancement to RepeatMasker. Bioinformatics 16(11), 1040–1041 (2000)
15. Gotoh, O.: An Improved Algorithm for Matching Biological Sequences. Journal of Molecular Biology 162(3), 705–708 (1982)
16. Abouelhoda, M.I., Kurtz, S., Ohlebusch, E.: Replacing suffix tree with enhances suffix arrays. Journal of Discrete Algorithms 2(1), 53–86 (2004)
17. Miki, B.L., Neelin, J.M.: DNA repeat lengths of erythrocyte chromatins differing in content of histones H1 and H5. Nucleic Acids Res. 8(3), 529–542 (1980)
18. National Center for Biotechnology Information, `http://www.ncbi.nim.nih.gov`

Motif Location Prediction by Divide and Conquer

Mohammed Alshalalfa and Reda Alhajj

Department of Computer science, University of Calgary, Calgary, Alberta, Canada
{msalshal,alhajj}@cpsc.ucalgary.ca

Abstract. Motif discovery recently received considerable interest from both computational biologists and computer scientists. Identifying motifs is greatly significant for understanding the mechanism behind regulating gene expressions. Although many algorithms have been proposed to solve this problem, only some of them use prior information about motifs. In this paper, we propose a method to limit the search space of the existing methods for motif discovery. Our method is based on the following observation: *if some elements are conserved, then these elements may be part of a conserved motif.* Further, the proposed approach is based on the divide and conquer concept, where we divide each DNA sequence into four subsequences, one subsequence per each of the four letters, representatives of the nucleotides, namely $\{A, C, G, T\}$. Then, we consider the subsequences for G as the major source for deciding on candidate motifs because G is found in almost all the transcription factors binding sites; the decision is supported and enhanced by the subsequences of the other three letters. We have applied this idea to yst04 and hm03r datasets; the results are encouraging as we have successfully predicted the locations of some of the motifs hidden within the analyzed sequences.

Keywords: motif discovery, gene expression data, sequence analysis, conserved motifs, divide and conquer.

1 Introduction

Transcription regulation is arguably one of the most important foundation of the cellular function, since it exerts the most fundamental control over the abundance of virtually all of cell's functional macromolecules. A predominant feature of transcription regulation is the binding of regulatory proteins, transcription factors (TFs), to cognate DNA binding sites in the genome, known as transcription factor binding sites (TFBS). The computational identification of TFBS through the analysis of DNA sequence data has emerged in the last decade as a major new technology for the elucidation of transcription regulatory networks.

With the increasing volume of the available biological sequences, finding regularities among the sequences as motifs could be identified as one of the emerging important biocomputing problems that received considerable attention in the research community. A motif, in the context of biological sequence analysis, is a consensus pattern of DNA bases or amino acids which accurately captures a conserved feature common to a group of DNA or protein sequences. DNA motifs are sometimes termed signals: examples are regulatory sequences, scaffold attachment sites, and messenger RNA splice sites.

Motif discovery is the act of identifying and characterizing motifs; the process underlies a number of important biomedical activities. In other words, the problem of motif discovery may be stated as follows. Given a set of sequences, which we have good reason to believe that they share a common binding motif, i.e., they have the same function, the target is how do we go about extracting these often degenerate patterns from the set of sequences. For example, the identification of regulatory signals has applications for gene finding in sequenced genomes, understanding of regulatory networks, and the design of drugs for regulating specific genes; and protein motifs are routinely used to identify the functions of newly-sequenced genes and to understand the basis of protein's cellular function.

The automatic motif discovery problem is a multiple sequence local alignment problem under the assumption that the motif model gives the optimal score for some appropriate scoring function. There are several cases for this problem:

***The simple sample*:** each sequence in the data set contains exactly one motif instance,
***The corrupted sample*:** an instance may not appear in every sequence,
***The invaded sample*:** more than one instance may exist in some sequences,
***The multiple patterns*:** the sequences may contain more than a single common motif.

There are many methods already described in the literature and widely used by researchers to discover motifssuch as MEME [1,2], BioProspector [17], AlignACE [12], Consensus [9], MDScan [18], and Weeder [27], among others. In real-life applications, a biologist would select one of these tools, but the main question that may arise is which one a biologist has to use?

There is no comparative study which compares the significance of motifs extracted from different motif discovery tools. A recent study published in *NATURE BIOTECHNOLOGY* [31] highlighted the statistical significance of the motifs discovered by the currently available tools. The authors have shown that Weeder is the most significant tool. In their study, they allowed only the best motif to represent the analyzed data set. They compared the statistical significance of the discovered motifs. However, by considering the real case from biological perspective, a gene can be regulated by more than one transcription factor. This means that there may be more than one motif per one dataset. Another interesting study described in [23] showed that the sequence feature of DNA binding sites reveal structural class of associated transcription factor. Given a data set of the upstream sequences, they could classify those sequences into different classes. Each class contains the genes which are regulated by transcription factors of the same structure.

In this study, we propose a new motif discovery method which may be classified as a divide and conquer based approach. The proposed approach consists of several steps. Since the genes within each of the two datasets yst04r and hm03r have the same function and are regulated by the same transcription factor, we applied the motif discovery process directly to these datasets. Further, realizing the complexity of the motif discovery process in general, we decided to apply a divide and conquer type of approach that helps reducing the search space for a target solution. Our motivation is the argument that a motif is a trend that occurs within each sequence in a cluster, and hence each of the four letters, namely $\{A, C, G, T\}$, should follow a certain trend within the motif. Based on this, we split each sequence into 4 subsequences, one per letter. The result will be four different matrixes, one per letter to keep track of its positions within each sequence. In other words, each row in a matrix corresponds to a sequence to reflect with the sequence the locations of the particular letter represented by the matrix. Note that it is not necessary for a letter to have same number of occurrences within all sequences. So, we consider the number of occurrences of the letter within each sequence and take the maximum as the number of columns; for rows with less number of occurrences, the extra entries are assigned the value -1. Finally, we apply the following process on each of the four matrixes.

We find subsequences of certain length such that the two extremes of the subsequence are p positions away from each other in the original sequence; in this study we consider p between 5 and 10. Then, we find the average distance from the head instance of the subsequence to each of its remaining instances. A subsequence is anticipated to be part of a candidate motif when it has the same length p and almost the same average distance within each subsequence of a particular letter. This way, analysis of the four matrixes returns possible subsequences of different candidate motifs. After this point, we are interested only in their positions within the original sequences. We consider subsequences for one of the letters as the major deciding factor with the subsequences of the other three letters as providers for supportive information to help in better ranking the former subsequences. The major subsequences in this study are those that correspond to the letter G. Letter G is selected as the major factor because it is found in almost all the TF binding sites. Also, many studies showed that some classes of TF, like Cys2His2, have conserved binding sites of G elements only. Locations of the subsequences that have similar length are ranked by the number of supporting rows within the same matrix and then by the number of other matrixes (of the other 3 letters) supporting them. Finally, we scan the ranked list from top to bottom by feeding the anticipated motif candidates to three of the existing tools, namely MEME [1,2], AlignACE [12], and Weeder [27]. Candidates confirmed by these tools are reported as the actual motifs. To sum up, the main contribution of this paper is the divide and conquer approach combined with ranking to identify positions of potential motifs.

The rest of this paper is organized as follows. Section 2 presents the related work. Section 3 covers the proposed approach. Section 4 reports the experimental results. Section 5 is discussion and conclusions.

2 Related Work

Motif discovery has received considerable attention as evident by the large number of related approaches described in the literature, e.g., [14,21]. Different statistics and machine learning techniques have been successfully adapted to the motif discovery process, including clustering, regression, joint probability, genetic algorithms, hidden Markov model, etc. For instance, some approaches, e.g., [4,5],are based on the fact that genes with similar expression levels should have homogeneous motifs, and hence their basic step is to cluster the genes first and then investigate motifs within each cluster by employing some of the existing motif discovery tools like MEME [1,2], BioProspector [17], AlignACE [12], Consensus [9], MDScan [18], and Weeder [27], among others. Other researchers employed regression on gene expression data for motif discovery, e.g., the Reduce method of Bussemaker *et al* [13], the Rim-Finder system of Zilberstein *et al* [34], the linear regression method called Motif Regressor of Conlon *et al* [7], the Logic-Motif approach of Keles *et al* [15] which uses two-step logistic regression on a single gene expression experiment. Finally, joint probabilistic approaches have also demonstrated successful in motif discovery, e.g., [3,10].

Expectation maximization (EM) can be used to simultaneously optimize a position weight matrix (PWM) description of a motif, and the binding probabilities for its associated sites [1]. Another approach, which is a stochastic implementation of EM, is Gibbs sampling [30]. The motif is typically initialized with a randomly selected set of sites, and every site in the target sequences is scored against this initial motif model. At each iteration, the algorithm probabilistically decides whether to add a new site and/or remove an old site from the motif model, weighted by the binding probability for those sites.

Weeder [27] is a commonly used tool for motif discovery. The idea of this algorithm is to enumerate all the oligos of or up to a given length, in order to determine which ones appear, with possible substitution, in a significant fraction of the input sequences, and finally to rank them according to statistical measure of significance.

Various approaches and tools have been proposed to handle the motif discovery problem using genetic algorithms (GA). For instance, both single and multi-objective GA have been used for discovering motifs in multiple unaligned DNA sequences [32].Stine *et al* [32] presented a structured GA to evaluate candidate motifs of variable length. Fitness values were assigned as a function of high scoring alignment performed with BLAST. Many other recent studies are using suffix trees for motif discovery, e.g., [20], and they have showed promising results.

One of the most common approaches to handle the motif discovery problem is multiple sequence alignment (MSA) [3]. Those approaches are found using dynamic programming. This approach has both advantages and drawbacks. The main advantage of MSAs for motif discovery is that they do not lose any information contained in each sequence. However, they have several disadvantages: they do not generalize the sequence data, and therefore are of limited help for biological understanding.

Hidden Markov Models (HMMs) have proved particularly useful for describing conserved sequences [8], being both flexible and better able to identify distant homologues than other approaches. Neural Networks have also been used to recognize and classify patterns in biological sequences [32]. Neural networks pose two significant problems when applied to motif discovery: (i) their relative inflexibility when mapping sequence data to inputs; and (ii) the difficulty of interpreting neural models, particularly when hidden layers are present within the network. Finally, three interesting studies for motif discovery have been recently published by Hartemink *et al* [23,24,25]. They used prior information about the TF binding sites to predict the location of the motif. In addition to this, they have shown that information nucleosome occupancy may improve the motif discovery process.

3 The Proposed Methodology

In this section, we describe the methodology developed to discover motifs in DNA sequences. The process consists of several steps. First, each sequence is split into 4 different subsequences, one per letter; and then, the subsequences are analyzed to identify candidate motifs which are fed into three motif discovery tools to report the actual motifs. Finally, it is worth mentioning that a preprocessing stage is required to cluster the genes to obtain sequences that share common motifs. However, such step is not described in this paper; we merely concentrate on how to derive motifs from a set of homogeneous sequences, which may equivalent to a certain cluster. In particular, we apply a divide and conquer based approach as described next.

3.1 Identifying Positions of Potential Candidate Motifs

The basic step of the method for motif discovery works by considering each set of homogeneous sequences (this may be considered equivalent to a cluster). For each group of homogeneous sequences, Algorithm 1 is invoked to identify and return positions for potential candidate motifs within each of the considered sequences. Algorithm 1 first splits the original sequences and produces four matrixes, one per each of the four letters $\{A, C, G, T\}$. Each of the four matrixes is used as input to Algorithm 2. The latter algorithm returns the positions for the potential candidate motifs. Here, it is worth noting that it is possible to report multiple potential motifs per sequence. However, at the end of the process only strong motifs are retained after employing the three motif discovery tools AlignACE, MEME and Weeder on all the returned potential motifs.

As we base the analysis on the indices of the letter G as the major deciding factor, we will use the letter G to illustrate the process of Algorithm 1. First a matrix called G is constructed to contain one row per sequence; each row contains the positions of the letter G within the corresponding sequence. Algorithm 1 invokes Algorithm 2 which finds from each row of matrix G all subsequences of positions such that each subsequence has h occurrences of G, where $6 < h < 15$. Since each subsequence contains positions, we determine the average

Algorithm 1. Identifying Positions of Potential Candidate Motifs in One Cluster

```
1: Given the input cluster $R$, consider four matrixes $A$, $C$, $G$, and $T$;
2: **for** each sequence $S_j$ in cluster $R$, $(j = 1, size(R))$ **do**
3:    Let $d_A$=0, $d_C$=0, $d_G$=0, $d_T$=0;
4:    **for** $i$=1 to length($S_j$) **do**
5:       **if** $(s_j = `A')$ **then**
6:          $A[j, d_A] = i$;
7:          $d_A = d_A + 1$;
8:       **else**
9:          **if** $(s_j = `C')$ **then**
10:            $C[j, d_C] = i$;
11:            $d_C = d_C + 1$;
12:         **else**
13:            **if** $(s_j = `G')$ **then**
14:               $G[j, d_G] = i$;
15:               $d_G = d_G + 1$;
16:            **else**
17:               **if** $(s_j = `T')$ **then**
18:                  $T[j, d_T] = i$;
19:                  $d_T = d_T + 1$;
20:               **end if**
21:            **end if**
22:         **end if**
23:      **end if**
24:   **end for**
25: **end for**
26: $W_A$=Find-Subsequences($R, A$); (Algorithm 2)
27: $W_C$=Find-Subsequences($R, C$);
28: $W_G$=Find-Subsequences($R, G$);
29: $W_T$=Find-Subsequences($R, T$);
30: Rank all the returned subsequences for the letter $G$ according to the following
    criteria:
    1) appears in more sequences within cluster $R$,
    2) supporting in majority of the other three results $W_A$, $W_C$, and $W_T$;
31: Return the final ranked list of potential positions for candidate motifs;
```

distance from the first position in the subsequence to all following positions within the same subsequence. For instance, the following result reflects for each of the seven sequences in the yst04 data, S-1 to S-7, the information related to all subsequences with 8 occurrences of the letter G; we mainly report the position P of the first occurrence of the letter G and the average distance D within the analyzed subsequence.

S-1 (P-407.D-7) (P-626.D-8) (P-636.D-7)
S-2 (P-4.D-6) (P-387.D-7) (P-749.D-5) (P-766.D-5) (P-817.D-7)
S-3 (P-1.D-6) (P-32.D-5) (P-83.D-5) (P-93.D-8) (P-121.D-6) (P-150.D-6) (P-443.D-7) (P-795.D-6)
S-4 (P-102.D-7) (P-106.D-6) (P-292.D-7) (P-432.D-9)

Algorithm 2. Find-Subsequences(R, M)

```
1: $R$ is the input cluster and M is one of the four matrixes;
2: for $i = 0$ to $size(R)$ do
3:    for $j = 0$ to NumberOfColumns($M$) do
4:       Find from row $i$ each subsequence of length between 6 and 15;
5:       For each such subsequence, find the average distance from its head to each
         instance in the subsequence;
6:    end for
7:    for $j = 6$ to 15 do
8:       if (there exist in each row of $M$ at least one subsequence of length $j$ such that
         all have similar average distances) then
9:          Report the position of each subsequence as location for potential candidate
            motif within the corresponding original sequence $i$ in cluster $R$;
10:      end if
11:   end for
12: end for
13: Return all the reported potential positions;
```

S-5 (P-88.D-6) (P-131.D-5) (P-187.D-7) (P-207.D-7) (P-282.D-6) (P-292.D-7) (P-312.D-4) (P-352.D-8) (P-366.D-6) (P-405.D-5) (P-455.D-8) (P-488.D-8)
S-6 (P-220.D-5) (P-294.D-6) (P-415.D-5) (P-416.D-8) (P-524.D-5) IP-694.D-6) (P-695.D-8) (P-795.D-5)
S-7 (P-169.D-6) (P-174.D-7) (P-186.D-6) (P-317.D-6) (P-321.D-6) (P-674.D-7)

The first pair of values in S-1 may be read as follows: there is in sequences S-1 a subsequence that contains 8 G's, it starts at position 407 in S-1 and 7 is the average distance for the distribution of G within the subsequence. This way, we are able to identify, in the rows of matrix G, subsequences that have the same length and the same average distance, regardless of the actual positions included in each subsequence. Algorithm 1 initially identifies pairs of length and average distance that repeat at least once in most of the rows of matrix G as frequent candidates for further investigation. Frequent candidates are located in the other three matrixes by the same way. Algorithm 1 ranks frequent candidates from matrix G based on the occurrence of some supportive frequent candidates from the other three matrixes. Supportive frequent candidates do not need to have same length and average distance pair as the corresponding frequent candidates in G; they are rather providers of supportive information as the actual positions within the sequence are concerned. In other words, a frequent candidate from matrix G and all its supportive frequent candidates from the other three matrixes should have close-by corresponding positions in the original sequence. After predicting the first positions of the highly ranked subsequences that have similar distribution of G, we extend the subsequences by 5 positions in both directions because it is not necessary to have G occurring on both ends of a motif. The point here is to reduce the large sequences into small subsequence of shorter length which have one nucleotide aligned correctly in all the sequences. After we classify and obtain the extended subsequences, we feed them to MEME, ALignACE and Weeder.

Then, we get the motifs discovered by these three tools and we find the TF binding site using TFSEARCH.

4 Results

To justify the effectiveness and applicability of the divide and conquer process described in this study, we have conducted experiments using yst04r and hm03r datasets, which were downloaded from the TRANSFAC database. These datasets were used because they are very well studied and corresponding results are reported in the literature. We have used the TFSEARCH tool for TF binding site discovery. Also, we have used three motif discovery tools; in particular, we decided to use AlignACE, MEME and Weeder because it has been shown by Tompa *et al* [31] that these three tools give the statistically significant motifs, in addition to their popularity among researchers who are interested in Motif discovery.

In the first part of the experiments, we want to demonstrate that the three tools, namely AlignACE, MEME and Weeder, are not always capable of discovering motifs which are biologically significant. So,we applied the three tools on yst04r and hm03r and obtain all the candidate motifs produced by these tools. Then, we used the TFSEARCH tool for testing the biological significant of the reported motifs. For each produced motif, we just checked whether it has a corresponding TF binding site or not. Explicitly, we have used TFSEARCH to discover the significant TF binding sites within the yst04 and hm03r data sets. TFSEARCH returned the binding sites of the following TFs: HSF, ADR1, GCR1 and RAP1 in yst03r, and cdxA, oct-1, GATA, SRY, AML-1a, c/EBPB, deltaE, Nkx-2, AP-1 and MZF1 for hm03r. When we feed TFSEARCH with the candidate motifs from the tools we used, we discovered that the three tools could only discover binding sites of HSF and GCR1 transcription factors in the yst04 data. As the hm03r data is concerned, Weeder was able to discover the binding sites of CdxA, Nkx-2, USF,and SRY. On the other hand, MEME could only discover SRY and AP-1 binding sites. We see from these results that none of the tools could discover all the TF binding sites; more than that, none of them explicitly consider the biological aspect in finding the motifs. As a result of this step, we have shown that, although the three tools could discover some significant motifs, none of them is able to discover all the TF binding sites hidden within the upstream sequences. Based on these results, we may argue that none of the tested three motif discovery tools is fully capable of producing biologically significant results. However, we realized that Weeder is still the tool which could discover more than the others, so we decided to use Weeder for motif discovery.

For the sake of the experiments conducted to demonstrate the power of the proposed approach, we assumed that if there is a set of sequences which have conserved element at specific position, then the other elements may be conserved in this set of sequences. We found all the subsequences of length from 6 to 15, which have 3 or 4 Gs and rank high based on the ranking criteria applied in this paper. In other words, we consider subsequences of G that repeat in most if not all the given sequences, and which are maximally supported by subsequences of the other three letters. Maximally supported means that most if not all of the three letters have some trends that repeat in most if not all of the sequences and in the same areas discovered by G; having positions of the trends returned for the four letters overlapping is the best targeted case. Here it is worth mentioning that it is trends for the same letter should have the same length, but the length should not be reserved between letters.

The 3 or 4 length was selected for G based on the notion that most TF binding sites are 15-20 bp. We tried to discover the core of the motif first, and then extend this core in both directions, along the right and left sides. If we find a set of subsequences of the same length and have the same average distance value D, then these subsequences were candidate locations to search for motifs. When we tested the binding sites of GCR1 reported by TRANSFAC, we observed that the scores are between 86-95 according to TFSEARCH. Our candidate motifs could predict binding sites of GCR1 of scores between 70-88. We have ranked the candidate location of motifs based on the support from the other elements, namely A, T, and C. The obtained results showed that when we set the length to be 6, having 3 Gs and D is 4, supportive elements were 4-5. This means that there are 7-8 aligned elements in those locations. All the candidate locations have GCR1 binding site of scores between 70-84. Other candidate locations of supportive values of 4, which have GCR1 binding site were predicted, such as motifs of length 7 and 8, which have 3 Gs and D is taken as 5 and 7, respectively. But, it was observed that only 4 out of the seven predicted locations have the GCR1 binding site. We have noticed that, for subsequences of length 6-8 and bounded by Gs and have one G within the subsequences have high rank and high supportive values. They also showed to have the target TF binding sites. Furthermore, our candidate locations showed to hide the TF binding sites of HSF and ADR1. These results are in agreement with Scot *et al* [33]. Apart from this, we noticed that the predicted locations of GCR1 are 0-100bp apart from the real GCR1 binding site location. For some locations, there was overlap, but others are 60-100 apart.

5 Discussion and Conclusions

In this work, we tried to investigate the biological importance of the motifs discovered by three of the most frequently used motif discovery tools, namely MEME, AlignACE, and Weeder. It has been shown that, although the results they produce are statistically significant, they are not biologically significant, in general. None of the tools consider the biological aspect, while searching for motifs.

Most of the algorithms dealing with the motif discovery problem do not use prior information on the location of the motif. Recently, many studies have been shown to be using prior information, either from Nucleotide occupancy or from TF binding sites [23,24,25]. Inhere, we tried to benefit from divide and conquer based algorithm and from Similar distribution of one element supported by the other elements, namely G and supported by A, C and T.

We can predict the location of a motif by dividing the data into four subsets, each has the indexes of one element, then benefiting from the similar distribution of one element in substrings from different sequences. In other words, we use information from the distribution of G to predict motif locations. We looked for subsequences of length 6-15, which are bounded by Gs and hide some Gs within it because most of the reported motifs have length of 10-20. Also, we have set the length of the candidate location to be 20 for the same reason. This idea showed to be promising to predict the location of motifs. In this work, we could discover the location of HSF and ADR1 transcription factors. This result is as good as the other tools. Our approach could discover the binding site of ADR1 which has been shown to have zinc fingers, which agree with our prior information about the binding sites of zinc finger transcription factors. We have also predicted locations for the real transcription factor in the considered data, which is GCR1. The predicted locations were so close to the real locations.

In conclusion, in this study we have shown that using prior information from the distribution of one element is a promising idea for further works; the discovery becomes stronger when supported by information related to the other three elements. Using this approach we could reduce the research space of the motif discovery tools to 20-100 bp instead of 1000 bp. We are expecting our approach to be more efficient for data with longer motifs (30-70), since that will reduce the false positive motifs, which have the same distribution of an element.

References

1. Baily, T.L., et al.: MEME: discovering and analyzing DNA and protein sequence motifs. Nucleic Acids Research 34, 369–373 (2006)
2. Bailey, T.L., Elkan, C.: The Value of Prior Knowledge in Discovering Motifs with MEME. In: Proc. of ISMB, Menlo Park, CA (1995)
3. Battle, A., Segal, E., Koller, D.: Probabilistic Discovery of Overlapping Cellular Processes and Their Regulation. In: Proc. of RECOMB, San Diego, CA (2004)
4. Beer, M.A., Tavazoie, S.: Predicting gene expression from sequence. Cell 117, 185–198 (2004)
5. Brazma, A., Jonassen, I., Vilo, J., Ukkonen, E.: Predicting Gene Regulatory Elements in Silico on a Genomic Scale. Genome Research 8, 1202–1215 (1998)
6. Bussemaker, H.J., Li, H., Siggia, E.D.: Regulatory Element Detection using Correlation with Expression. Nature Genetics 27, 167–171 (2001)
7. Conlon, E.M., Liu, X.S., Lieb, J.D., Liu, J.S.: Integrating Regulatory Motif Discovery and Genome-wide Expression Analysis. PNAS 100(6), 3339–3344 (2003)
8. D'haeseleer, P.: How does DNA sequence motif discovery work? Nature Biotechnology 24(8) (2006)
9. Hertz, G.Z., Stormo, G.D.: Identifying DNA and Protein Patterns with Statistically Significant Alignments of Multiple Sequences. Bioinformatics 15(7/8), 563–577 (1999)
10. Holmes, I., Bruno, W.J.: Finding regulatory elements using joint likelihoods for sequence and expression profile data. In: Proc. of International Conference of Intelligent Systems for Molecular Biology, pp. 202–210 (2000)
11. Hu, J., Li, B., Kihara, D.: Limitations and potentials of current motif discovery algorithms. Nucleic Acids Research 33(15), 4899–4913 (2005)

12. Hughes, J.D., Estep, P.W., Tavazoie, S., Church, G.M.: Computational Identification of Cis-regulatory Elements Associated with Groups of Functionally Related Genes in Saccharomyces Cerevisiae. Journal of Molecular Biology 296, 1205–1214 (2000)
13. Jensen, S.T., Shen, L., Liu, J.S.: Combining phylogenetics motif discovery and motif clustering to predict co–regulated genes. Bioinformatics 21(20), 3832–3839 (2005)
14. Kechris, K.J., van Zwet, E., Bickel, P.J., Eisen, M.B.: A Boosting Approach for Motif Modeling using ChIP-chip Data. Bioinformatics 21(11), 2636–2643 (2005)
15. Keles, S., van der Laan, M.J., Vulpe, C.: Regulatory Motif finding by Logic Regression. U.C. Berkeley Biostatistics Working Paper Series, (145) (2004)
16. Kundaje, A., Middendorf, M., Gao, F., Wiggins, C., Leslie, C.: Combining sequence and time series expression data to learn transcriptional modules. IEEE Transactions on Computational Biology and Bioinformatics 2(3), 194–202 (2005)
17. Liu, X., Brutlag, D.L., Liu, J.S.: Bioprospector: Discovering Conserved DNA Motifs in Ppstream Regulatory Regions of Co-expressed Genes. In: Proc. of Pacific Symposium on Biocomputing (2001)
18. Liu, X.S., Brutlag, D.L., Liu, J.S.: An Algorithm for Finding Protein-DNA Binding Sites with Applications to Chromatin-Immunoprecipitation Microarray Experiments. Nature Biotechnology (20), 835–839 (2002)
19. Lones, M.A., Tyrrell, A.M.: The evolutionary computation approach to motif discovery in biological sequences. In: Proc. of GECCO workshop (2005)
20. Marsan, L., Sagot, M.: Algorithms for extracting structured motifs using a suffix tree with an application to promoter and regulatory site consensus identification. Journal of computational Biology 7(3/4), 345–362 (2000)
21. Middendorf, M., Kundaje, A., Shah, M., Freund, Y., Wiggings, C.H., Leslie, C.: Motif Discovery through Predictive Modeling of Gene Regulation. In: Proc. of RECOMB, Cambridge, MA (2005)
22. Moreau, Y., Thijs, G., Marchal, K., De Smet, F., Mathys, J., Lescot, M., Rombauts, S., Rouze, P., De Moor, B.: Integrating Quality-based Clustering of Microarray Data with Gibbs Sampling for the Discovery of Regulatory Motifs. JOBIM, 75–79 (2002)
23. Narlikar, L., Hartemink, A.: Sequence features of DNA binding sites reveal structural class of associated transcription factor. Bioinformatics 22, 157–163 (2006)
24. Narlikar, L., Gordan, R., Hartemink, A.J.: Nucleosome occupancy information improves de novo motif discovery. In: Speed, T., Huang, H. (eds.) RECOMB 2007. LNCS (LNBI), vol. 4453. Springer, Heidelberg (2007)
25. Narlikar, L., Gordan, R., Ohler, U., Hartemink, A.J.: Informative priors based on transcription factor structural class improve de novo motif discovery. Bioinformatics 22, 384–392 (2006)
26. Paul, T.K., Iba, H.: Identification of weak motifs in multiple biological sequences using genetic algorithm. In: Proc. of GECCO 2006 (2006)
27. Pavesi, G., et al.: 'Weeder Web: discovery of transcription factor binding sites in a set of sequences from co-regulated genes. Nucleic Acid Research 32, 199–203 (2004)
28. Segal, E., Yelensky, R., Koller, D.: Genome-wide Discovery of Transcriptional Modules from DNA Sequence and Gene Expression. Bioinformatics 19(1), 273–282 (2003)
29. Segal, E., Barash, Y., Simon, I., Friedman, N., Koller, D.: From Promoter Sequence to Expression: A Probabilistic Framework. In: Proc. of RECOMB, Washington, DC (2001)

30. Thompson, W., Rouchka, E.C., Lawrence, C.E.: Gibbs recursive sampler: finding transcription factor binding sites. Nucleic Acids Research 31(13), 3580–3585 (2003)
31. Tompa, M., et al.: Assessing computational tools for the discovery of transcription factor binding sites. Nature Biotechnology 23(1), 137–144 (2005)
32. Stine, M., et al.: Motif discovery in upstream sequences of coordinately expressed genes. In: Proc. of CEC, USA, pp. 1596–1603 (2003)
33. Wolfe, S.A., Nekludova, L., Pabo, C.O.: DNA Recognition by Cys_2 His_2 Zinc Finge Proteins. Annu. Rev. Biophys. Biomol. Stru. 3, 183–212 (1999)
34. Ben-Zaken Zilberstein, C., Eskin, E., Yakhini, Z.: Sequence Motifs in Ranked Expression Data. Technion CS Dept. Technical Report (CS-2003-09) (2003)
35. Zhang, Y., Chen, Y., Ji, X.: Motif Discovery as a multiple instance problem. In: Proc. of IEEE ICTAI, pp. 805–809 (2006)
36. Zhu, Z., Pilpel, Y., Church, G.M.: Computational Identification of Transcription Factor Binding Sites via a Transcription-factor-centric Clustering (TFCC) Algorithm. Journal of Molecular Biology (318), 71–81 (2002)

Translational Control by RNA-RNA Interaction: Improved Computation of RNA-RNA Binding Thermodynamics

Ulrike Mückstein[1,⋆], Hakim Tafer[1,⋆], Stephan H. Bernhart[2], Maribel Hernandez-Rosales[2], Jörg Vogel[3], Peter F. Stadler[1,2,4,5], and Ivo L. Hofacker[1]

[1] Institute for Theoretical Chemistry, University of Vienna, Währingerstrasse 17, A-1090 Vienna, Austria
{ulim,htafer,ivo}@tbi.uvivie.ac.at
http://www.tbi.univie.ac.at/∼ivo/RNA/

[2] Bioinformatics Group, Department of Computer Science, and Interdisciplinary Center for Bioinformatics, University of Leipzig, Härtelstrasse 16-18, D-04107 Leipzig, Germany
{bstephan,maribel,studla}@bioinf.uni-leipzig.de

[3] RNA Biology, Max Planck Institut für Infektionsbiologie, Charitéplatz 1, Campus Charité Mitte, D-10117 Berlin, Germany
vogel@mpiib-berlin.mpg.de

[4] RNomics Group, Fraunhofer Institut for Cell Therapy and Immunology (IZI), Deutscher Platz 5e, D-04103 Leipzig, Germany

[5] Santa Fe Institute, 1399 Hyde Park Rd., Santa Fe, NM 87501, USA

Abstract. The thermodynamics of RNA-RNA interaction consists of two components: the energy necessary to make a potential binding region accessible, i.e. unpaired, and the energy gained from the base pairing of the two interaction partners. We show here that both components can be efficiently computed using an improved variant of `RNAup`. The method is then applied to a set of bacterial small RNAs involved in translational control. In all cases of biologically active sRNA target interactions, the target sites predicted by `RNAup` are in perfect agreement with literature. In addition to prediction of target site location, `RNAup` can also be used to determine the mode of sRNA action. Using information about target site location and the accessibility change resulting from sRNA binding we can discriminate between positive and negative regulators of translation.

1 Introduction

A series of high-throughput transcriptomics projects, among them ENCODE [1] and FANTOM [2] have demonstrated that mammalian genomes are pervasively transcribed, and that a large fraction of the transcripts does not code for proteins. Concurrently, small RNAs, in particular microRNAs and siRNAs have been identified as crucial regulators of gene expression, reviewed e.g. in [3].

⋆ The first two authors contributed equally to this work.

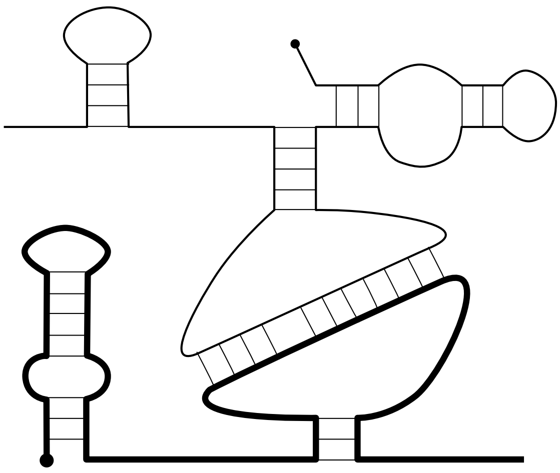


Fig. 1. Interaction between two RNAs of comparable length. Since each molecule forms intramolecular structures, the accessibility for an interaction differs along the molecule: Unstructured regions can easily take part in an interaction. Regions that are involved in an intramolecular structure, e.g. the left hand side of the molecule drawn as a bold line, are not easily accessible for intermolecular binding.

Genome-wide mapping of small ncRNAs [4] revealed novel classes of ncRNAs, implying that ncRNAs act by several, if not many, different mechanisms.

MicroRNAs, siRNAs and snoRNAs require the direct interaction of ncRNAs and their target by means of base-pairing [5]. The same is true for many of the bacterial small RNAs discovered during the last decade, see e.g. [6]. Computational evidence [7] suggests, furthermore, that a significant fraction of RNA candidates with evolutionary conserved RNAs [8] binds to mRNAs.

These observations have triggered increasing interest in methods to predict "targets" via the evaluation of RNA-RNA interactions. For microRNAs, the available tools are almost too numerous to list (see [9,10] for recent reviews), `targetRNA` [11] is frequently used for bacteria, and a specific heuristic for orphan snoRNAs was presented recently [12]. In the most simple case, only the base pairing between the two interacting partners is taken into account [13,14,15,16,11]. In most cases, however, RNA-RNA interaction does not cover the entire target. This is maybe most evident in the case of short siRNAs or miRNAs targeting long mRNAs. It becomes necessary in such cases, to explicitly consider the structure of the target. In [17], anti-sense targets are predicted as unpaired regions on the target molecules. For siRNA and microRNA it was shown that the accessibility of the target site correlates directly with the efficiency of cleavage [18,19].

Instead of treating the target independent of its binding partner, it seems more appealing to compute the structure of the interaction complex. Just as the folding problem with pseudoknots [20], finding the energetically optimal interaction structure is NP-complete [21]. It is, however, not even desirable to solve the general "RIP" problem, because too highly entangled structures typically are not formed in nature. Practical approaches therefore restrict the set of interaction structures that are searched. So far, four classes of structures have been investigated in some detail:

1. Only base-pairs between the interacting RNAs are considered, no base pairs are allowed within each structure. As argued above, disregarding the internal structure of the interaction partners may be too crude an approximation.
2. Interactions between the two molecules are restricted to the external bases of the two partners. Such structures can be computed by means of a straightforward generalization of the usual pseudoknot-free folding algorithm [22,23]. This class of structures, however, is still too restrictive as it rules out frequent motifs such as kissing-hairpins [24].
3. The other extreme is to consider all "tangle-free" interaction structures. This leads to a rather expensive algorithm with a runtime $\mathcal{O}(m^3 \cdot n^3)$, where m and n are the lengths of the interacting sequences, and quartic memory consumption [25,21,26,27], which is prohibitive for many large-scale applications. Another problem is that the interaction structures contain many types of complex loops for which energy parameters are unknown.
4. The `RNAup` approach [28] restricts the region of interaction to a single interval on each of the interaction partners, while arbitrary pseudoknot free structures are allowed elsewhere, see Fig. 1. This model is sufficient for most but not all known RNA-RNA interactions. For example, the OxyS–fhlA interaction [29] contains two separate kissing complexes and therefore can not be predicted using `RNAup`. Most bacterial sRNAs however show one well defined interaction with a typical interaction length from 9 bp up to 60 bp and variable degrees of complementarity between ncRNAs and target sequence [30,31]. In [28], only the target molecule was assumed to be structured, while the ncRNA partner was assumed to be a miRNA or siRNA without internal structure. Here we will drop this restriction.

Instead of directly computing the interaction structure, `RNAup` decomposes the problem into three steps: For each subsequence (with bounds i and j) of an RNA, we compute the probability $P[i, j]$ that it is unpaired. This probability is equivalent to the free energy of *making* the binding regions accessible. The optimal interaction structure is then computed by assessing all possible combinations of binding sites of both partners.

This conceptual decomposition of RNA/RNA binding into an unfolding and an interaction contribution has most recently been adopted by several groups. Long *et al.* [32] developed a model for modeling the interaction between a miRNA and a target as a two-step hybridization reaction: nucleation at an accessible target site, followed by hybrid elongation to disrupt local target secondary structure and formation of the complete miRNA-target duplex. Lu & Mathews [33] predicted the cost of opening base pairs in the mRNA for hybridization to siRNA by calculating the structure once without constraints and then once with the constraint that the nucleotides in the hybridization site are forced to be single-stranded. A similar approach is taken in Tafer et al. [34] where accessibility is computed using the RNAplfold program [35]. Kertesz *et al.* [19] devised a parameter-free model for microRNA-target interaction that computes the difference between the free energy gained from the formation of the microRNA-target duplex and the energetic cost of unpairing the target to make it accessible to the microRNA.

In the following sections we first describe an algorithmic improvement in the computation of $P[i,j]$ that leads to a significant speed-up of `RNAup`. Then we show how to include secondary structure information of both interaction partners in the computation of the free energy of binding. In the results section, we report how these improvements allow us to more precisely describe translational control by bacterial sRNA.

2 Algorithm

`RNAup` calculates the energetics of RNA-RNA interactions in a stepwise process. The free energy of binding ΔG consists of the "breaking energies" ΔG_u that are necessary to render the binding site on each molecule accessible and a contribution ΔG_h that describes the energy gain due to hybridization:

$$\Delta G = \Delta G_u^A + \Delta G_u^B + \Delta G_h, \tag{1}$$

where A and B denote the two interacting molecules. In principle, Eq. 1 has to be evaluated for every possible combination of interacting regions in molecule A and B. In practice, our algorithm first computes the accessibilities ΔG_u for all regions up to a maximum size w and then combines these regions to compute the hybridization energies ΔG_h.

In order to compute free energies of binding we cannot rely on finding a single optimal structure only. Instead, we have to compute the partition functions associated with these three free energy terms. This can be done with (suitably modified) variants of the algorithm introduced by McCaskill [36] and implemented in the `Vienna RNA package` [37]. Recall that the equilibrium partition function is defined as

$$Z = \sum_S \exp(-\beta F(S)) \,, \tag{2}$$

where $F(S)$ is the free energy of a secondary structure S, and $\beta = 1/(RT)$ is the inverse of the temperature times Boltzmann's constant (here expressed as the gas constant, i.e. for energies per mol). Note that individual secondary structures are assigned temperature dependent free energies with entropic contributions arising from the ensemble of microscopic conformations that are assigned to a single secondary structure as macro state. Energy parameters used here are taken from [38].

2.1 Calculation of Accessibility

Partition functions for subsequences contain the information necessary to compute the frequency of structural motifs, in the simplest case individual unpaired bases or base pairs [36].

Here, we are interested in the probability $P_u[i,j]$ that the sequence interval $[i,j]$ is unpaired, which is equivalent to the energy $\Delta G_u[i,j] = -RT\ln(P_u[i,j])$ necessary to make the subsequence from i to j single-stranded. An unpaired interval $[i,j]$ is either "exterior", i.e. not enclosed by a basepair, or there exists an enclosing base pair (p,q) such that $p < i < j < q$ and there is no other pair (s,t) such that $p < s < i < j < t < q$. We can therefore express $P_u[i,j]$ in terms of restricted partition functions for these two cases:

$$P_u[i,j] = \frac{Z(1,i-1)Z(j+1,n) + \sum_{p<i}\sum_{j<q}\hat{Z}(p,q)Z_{pq}[i,j]}{Z(1,n)} \tag{3}$$

where $\hat{Z}(p,q)$ is the partition function outside base pair (p,q), and $Z_{pq}[i,j]$ the partition function inside a base pair (p,q) given that the interval $[i,j]$ is unpaired. Here we introduce an improved recursion for $\hat{Z}(p,q)Z_{pq}[i,j]$ that reduces the CPU requirements of the previous implementation of RNAup [39] from $\mathcal{O}(n^3 \cdot w)$ to $\mathcal{O}(n^3)$, where n is the length of the sequence and w is the maximal size of the unstructured region $[i,j]$.

As in [39], we start from the observation that $Z_{pq}[i,j]$ consists of three contributions, of which the summation of all multi-loop energies is the most complex one. This multi-loop part is again split into three parts, depending on whether the unpaired region is to the left or to the right of all components of a multi-loop or in between them, Fig. 2:

$$\begin{aligned} Z^{mult}[i,j] = \sum_{p<i<j<q} & \hat{Z}(p,q)\times \\ & \left(\underbrace{Z^{M2}(p+1,i-1)e^{-\beta c(q-i)}}_{\text{left}} \quad + \quad \underbrace{Z^{M2}(j+1,q-1)e^{-\beta c(j-p)}}_{\text{right}} \right. \\ & \left. \quad + \quad \underbrace{Z^{M}(p+1,i-1)e^{-\beta c(j-i+1)}Z^{M}(j+1,q-1)}_{\text{in-between}} \right) \end{aligned} \tag{4}$$

The crucial improvement is obtained by replacing the double sum in Eq. 3 by two separate summation steps. For the last, "in-between", summand we use the auxiliary variables

$$Z^{MM}(q)[i] = \sum_{1\le p<i} \hat{Z}(pq)Z^{M}(p+1,i-1) \tag{5}$$

For $Z_l^M(q)[i]$ where the unpaired region $[i,j]$ is to the left of all multi-loop components, we introduce

$$Z_l^M(q)[i] = \sum_{1\le p<i} \hat{Z}(p,q)Z^{M2}(p+1,i-1)e^{-\beta c(q-i)} \tag{6}$$

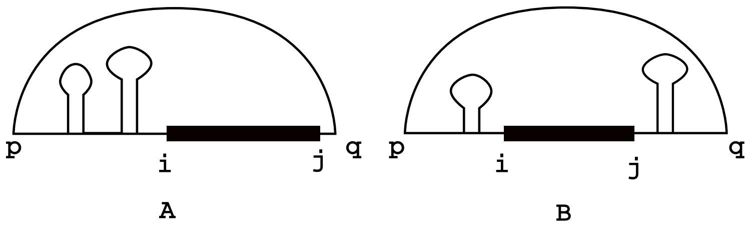


Fig. 2. Decomposition for calculating multiloop contributions: Base pair $[p, q]$ that includes the unpaired region $[i, j]$ is drawn as an arc connecting bases p and q. The unpaired region $[i, j]$ is drawn as a bold black line. In the one-sided multiloop case (A) a structured region containing *at least* two structure components is on one side of the unpaired region. In case (B) the unpaired region $[i, j]$ is between two structured regions. In case (B) we have to take care to make a unique decomposition of the multiloop into a 3' part that contains exactly one component and a 5' part with at least one component.

and an analogous term is used for the "right" contribution. Computing these values costs $\mathcal{O}(n^3)$. By using them, we can compute

$$\begin{aligned} Z^{mult}[i,j] = &\sum_{j<q} Z^{MM}(q)[i] e^{-\beta c(j-i+1)} Z^M(j+1,q-1) \\ &+ \sum_{p<i} Z_r^M(p)[j] \\ &+ \sum_{j<q} Z^M(j+1,q-1) + Z_l^M(q)[i] \end{aligned} \tag{7}$$

in $\mathcal{O}(n^2 \cdot w)$ time, i.e., the entire algorithm is $\mathcal{O}(n^3)$. The computations for hairpin and interior loop contributions are handled in the same way.

In comparison to McCaskill's partition function algorithm, `RNAup` needs to store five additional matrices (Z^{M2}, Z^{MM}, Z_l, Z_r and one additional matrix for the interior loop case). Hence we buy the speed-up by $\mathcal{O}(w)$ by increasing the memory requirements by only about a factor of 2. A comparison of the execution times of the old and the new version of `RNAup` shows that the new version is 20 times faster for the default settings ($w = 25$) and sequence lengths below 400 nucleotides. For sequence lengths between 400 and 2000 nucleotides the speed up decreases with increasing sequence length, but the new version is at least 12 times faster. This substantial performance gain considerably facilitates large-scale applications.

2.2 Free Energy of Interaction

In [39] we used $P_u[i, j]$ for the (long) target mRNA only, assuming that the siRNA or miRNA is unstructured due to its short length. This approximation cannot be justified for most bacterial small RNAs, however. Hence, we extended `RNAup` to take the secondary structure of both interacting molecules into account.

Suppose the interaction region covers the intervals $[i^*, j^*]$ and $[i, j]$ in the two RNAs. As in `RNAhybrid` and related programs, we allow interior loops and bulges in the interaction region. The partition function over all these binding conformations is obtained by the following recursion:

$$Z^I[i,j,i^*,j^*] = \sum_{\substack{i<k<j \\ i^*>k^*>j^*}} Z^I[i,k,i^*,k^*] e^{-\beta I(k,k^*;j,j^*)}. \tag{8}$$

where $I(k,k^*;j,j^*)$ is the energy contribution for the interior loop delimited by the base pairs (k,k^*) and (j,j^*).

As we want to avoid having to keep track of a four dimensional array, we compute the partition function $Z^*[i,j]$ over all structures where region $[i,j]$ in the *longer* molecule is involved in the interaction. While doing this, we keep track of the region where $Z^I[i,j,i^*,j^*]$ is maximal. The recursion for the calculation of $Z^*[i,j]$ is shown in Eq 9.

$$Z^*[i,j] = P_u^A[i,j] \sum_{i^*>j^*} P_u^B[i^*,j^*] Z^I[i,j,i^*,j^*]. \tag{9}$$

From $Z^*[i,j]$ we can readily compute $\varDelta G[ij]$, the free energy of binding given the binding site is in region $[i,j]$. For visual inspection, $\varDelta G[ij]$ can be reduced to the optimal free energy of binding $\varDelta G[i]$ at a given position i, see Eq 10. The memory requirement for these steps is $\mathcal{O}(n \cdot w^3)$, the required CPU time scales as $\mathcal{O}(n \cdot w^5)$, which, at least for long target RNAs, is dominated by the first step, i.e., the computation of the $P_u[i,j]$.

$$\begin{aligned} \varDelta G[i,j] &= -RT \ln Z^*[i,j]. \\ \varDelta G[i] &= \min_{k \le i \le l}\{\varDelta G[k,l]\}. \end{aligned} \tag{10}$$

The positional free energy, $\varDelta G[i]$, referring to position i in the target molecule, is written to a file. For the region with maximal $Z^I[i,j,i^*,j^*]$, we use `RNAduplex` to print out the optimal interaction structure.

3 Results

To test whether the changes in `RNAup` improve its applicability, we studied experimentally verified interactions between bacterial small RNAs (sRNAs) and their targets [30]. Bacterial sRNAs are ideally suited to examine the usefulness of the inclusion of the secondary structure of both interaction partners into the free energy calculations, since sRNAs are long enough to be highly structured. Furthermore the binding region usually spans only part of the sRNA binds. Therefore, the secondary structure of the sRNA will critically influence the exact location of the binding site.

As a first test we compared the binding sites predicted by the old version of `RNAup`, which neglects sRNA structure, with the predictions of the new version that computes the contributions of both structures. As expected, when omitting the structure within the sRNA the binding energy was markedly higher (mean -24.97 ± 5.97) than in the new version (mean -15.54 ± 1.99).

When comparing binding site location with the location of experimentally verified binding sites, see Table 1, we found that the new version predicts binding sites more accurately than the old version. In the new version 3 binding sites were predicted with perfect accuracy (the predicted binding site did not deviate by more than one base pair from the binding site reported in literature), and 7 binding sites deviate by at most 17 base pairs, see Table 1. Neglecting sRNA structure, on the other hand, predicts no binding site with perfect accuracy, 9 binding sites show a deviation between 4 to 45 base pairs, $(4, 11, 12, 16, 27, 33, 39, 39, 45)$, and one binding site prediction was wrong, i.e. far away from the site reported in literature.

This comparison emphasises the importance of the inclusion of secondary structure information of both binding partners when predicting sRNA-mRNA interactions. Neglecting the structure of the sRNA results in an overestimation of the length of the predicted interaction and in most cases hinders the clear localization of the proper target site boundary.

In addition to the location of the binding site, the regulatory effects upon binding of the sRNA to the its target mRNA was studied. We used a data set consisting of 9 small regulatory RNAs from *E.Coli*, their 9 reported mRNA targets and the fold-change in protein concentration induced by all 81 possible mRNA-ncRNA interactions [30]. Among those interactions, 8 targets were downregulated, 2 were upregulated, and no or only marginal changes were detected for the others (see Table 1). Downregulation usually occurs when the hybridisation of the ncRNA with its cognate mRNA blocks the ribosome entry sites on the target (for a review see [40]). In contrast, upregulation typically takes place when the sRNA-mRNA hybridization disrupts intrinsic inhibitory structures that sequester the ribosome binding site and/or the start codon [41,42,43]. In many cases the sRNA-mRNA interactions are assisted by the RNA chaperone protein *Hfq* [44].

Target prediction was performed with the mRNA constructs (117-689 nts) described in [30] and the full length sRNAs (69-220 nts). The mRNA constructs included a long 5'UTR sequence (57-565 nts) and a comparably short fragment of the CDS (35-139 nts). Both the hybridisation energy and the target site position were computed with `RNAup` for all sRNA-mRNA combinations.

For each sRNA we tested which of the mRNA constructs was predicted to bind most strongly. To our satisfaction the most favorable binding energy for each sRNAs was found for its cognate target (see Table 1).

Since the most common mechanism of translational control is to influence ribosome binding at the Shine-Dalgarno (SD) sequence, we checked the position and structural effects of the predicted interactions. For each of the 8 interactions that resulted in downregulation, we found the binding site to be at or close to the Shine-Dalgarno sequence. This type of inhibition can thus be predicted by comparing `RNAup` predictions with sequence features that are easy to recognize in bacterial genomic sequences.

Our data set contains only two examples of upregulation, namely binding of *DsrA* and *RprA* to *rpoS*. In both cases, binding leads to the disruption of a

Table 1. Binding site summary for the 10 functional interactions published by Urban et.al [30]. Column $\Delta\Delta G$ shows the optimal binding energy calculated with `RNAup`. Column Position gives the binding position relative to the start codon. Column Position lit. gives the binding position found in the literature.

mRNA	sRNA	regulation	$\Delta\Delta G$	Position	Pos.lit.	cite
RyhB	sodB	-	-11.50	-18,+4	-4,+5	[45]
DsrA	hns	-	-14.60	-10,+11	+7,+19	[46]
MicA	ompA	-	-13.60	-21,-6	-21,-6	[47]
MicC	ompC	-	-15.80	-30,-15	-30,-15	[48]
MicF	ompF	-	-17.80	-11,+9	-11,+10	[48]
Spot42	galK	-	-17.00	-18,+30	-19,+21	[49]
SgrS	ptsG	-	-17.33	-28,-10	-28,+4	[50]
GcvB	dppA	-	-17.30	-30,-7	-31,-14	[31]
DsrA	rpoS	+	-14.52	-126,-97	-119,-97	[42]
RprA	rpoS	+	-15.90	-134,-94	-117,-94	[42]

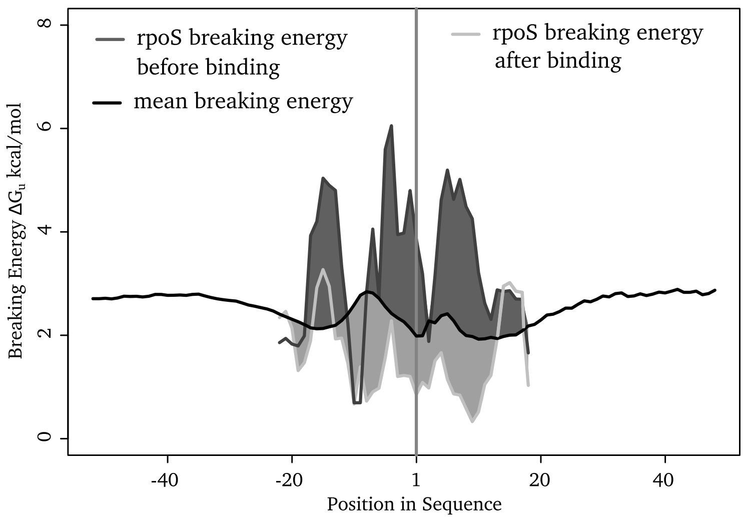


Fig. 3. Opening energy, ΔG_u plotted versus sequence position for the interaction of *DsrA* with textitrpoS. The vertical gray line marks the position of the start codon. The black line represents the average breaking energy for all *E. Coli* mRNAs. The dark gray line represents the opening energy of unbound *rpoS*, the light gray line the opening energy after binding *DsrA*. Unbound *rpoS* is less accessible than average (dark gray area), while bound *rpoS* is more accessible than average (light gray area).

helix which normally sequesters the Shine-Dalgarno sequence as well as the start codon. We remark that this is an example of the modifier RNA mechanism that was proposed in [51,52].

To assess the ability of `RNAup` to predict upregulating interactions we first compared the accessibility of the region around the start codon of all 9 mRNAs, with the mean accessibility of all 4463 genes in the E.Coli genome. Mean accessibility was computed for regions of 401 nts, centered at the start codon. For comparability we used the same 401 nts regions of our 9 target genes rather than the constructs used above. The accessibilities and corresponding opening energies were computed with RNAup for unpaired regions of length 4. The screen against the E.coli genome with all 9 sRNAs took 16 CPU days on one core of an Intel Core2 duo CPU with 2 GB RAM running at 2.40GHz.

With a local opening energy of 4.51 kcal/mol *rpoS* is the most inaccessible transcript among the 9 transcripts presented here. Genome-wide only 8.8% of the transcripts have a less accessible start codon than *rpoS*. In contrast, the eight downregulated transcripts showed a higher than average (2.23 kcal/mol) accessibility, ranging from 0.30 kcal/mol for ompA to a maximum of 1.27 kcal/mol for ryhB.

After binding *DsrA*, the accessibility of the *rpoS* start codon changes dramatically. With only 1.40 kcal/mol, bound *rpoS* is much more accessible than the average transcript and belongs to the 33% most accessible genes, see fig. 3. The same effect is seen upon binding with *RprA*, with a local accessibility after binding of 1.90 kcal/mol. Technically, accessibilities after binding can be computed easily by adding the constraint that nucleotides in the binding site remain single stranded.

4 Conclusion

Translational control by sRNAs is an important regulatory function throughout all bacteria. In contrast to e.g.micro RNAs, these regulatory RNAs are mostly structured. We have improved `RNAup` to take both target and sRNA structure into account. As we have also increased the speed of `RNAup`, it is now suitable for the computational identification of mRNA targets of bacterial sRNAs.

Furthermore, we find that `RNAup` can be used to predict the regulatory effect of sRNA binding by investigating the location of the binding site and the structural changes induced by binding in the vicinity of the start codon of the mRNA. A predicted binding close to the start codon or the Shine-Dalgarno sequence is a clear indicator for downregulation. While results look promising for upregulation, a bigger data set is needed to confirm that `RNAup` can also accurately predict it.

Our algorithm captures the most common types of interaction between regulatory RNAs and their targets, even though more complicated types of interactions, such as H/ACA snoRNA with their target rRNAs or OxyS–fhlA, are neglected. The speed of `RNAup` is clearly sufficient for genome wide searches for sRNA–mRNA interactions in bacteria. In principle, the approach is equally applicable to interaction search in higher organisms. However, the larger genome size and longer UTR regions pose challenges both in terms of computation time and false positives.

Acknowledgments

This work has been funded, in part, by the Austrian GEN-AU projects bioinformatics integration network and non-coding RNA, the FP-6 EMBIO project, the *Deutsche Forschungsgemeinschaft* Proj No STA 850/7-1 as part of SPP-1258 "Sensory and Regulatory RNAs in Prokaryotes" and Siemens.

References

1. The ENCODE Project Consortium: Identification and analysis of functional elements in 1% of the human genome by the ENCODE pilot project. Nature 447, 799–816 (2007)
2. Maeda, N., Kasukawa, T., Oyama, R., Gough, J., Frith, M., Engström, P.G., Lenhard, B., Aturaliya, R.N., Batalov, S., Beisel, K.W., Bult, C.J., Fletcher, C.F., Forrest, A.R., Furuno, M., Hill, D., Itoh, M., Kanamori-Katayama, M., Katayama, S., Katoh, M., Kawashima, T., Quackenbush, J., Ravasi, T., Ring, B.Z., Shibata, K., Sugiura, K., Takenaka, Y., Teasdale, R.D., Wells, C.A., Zhu, Y., Kai, C., Kawai, J., Hume, D.A., Carninci, P., Hayashizaki, Y.: Transcript annotation in FANTOM3: Mouse gene catalog based on physical cdnas. PLoS Genetics 2, e62 (2006), doi:10.1371/journal.pgen.0020062.
3. Mattick, J.S., Makunin, I.V.: Non-coding RNA. Hum. Mol. Genet. 15, 17–29 (2006)
4. Kapranov, P., Cheng, J., Dike, S., Nix, D., Duttagupta, R., Willingham, A.T., Stadler, P.F., Hertel, J., Hackermüller, J., Hofacker, I.L., Bell, I., Cheung, E., Drenkow, J., Dumais, E., Patel, S., Helt, G., Madhavan, G., Piccolboni, A., Sementchenko, V., Tammana, H., Gingeras, T.R.: RNA maps reveal new RNA classes and a possible function for pervasive transcription. Science 316, 1484–1488 (2007)
5. Schubert, S., Gruenweller, A., Erdmann, V.A., Kurreck, J.: Local RNA target structure influences siRNA efficacy: systematic analysis of intentionally designed binding regions. J. Mol. Biol. 348(4), 883–893 (2005)
6. Vogel, J., Wagner, E.G.: Target identification of small noncoding RNAs in bacteria. Curr. Opin. Microbiol. 10, 262–270 (2007)
7. The Athanasius F. Bompfünewerer RNA Consortium: Backofen, R., Flamm, C., Fried, C., Fritzsch, G., Hackermüller, J., Hertel, J., Hofacker, I.L., Missal, K.: RNAs everywhere: Genome-wide annotation of structured RNAs. J. Exp. Zool. B: Mol. Dev. Evol. 308B, 1–25 (2007)
8. Washietl, S., Hofacker, I.L., Lukasser, M., Hüttenhofer, A., Stadler, P.F.: Mapping of conserved RNA secondary structures predicts thousands of functional noncoding RNAs in the human genome. Nature Biotech. 23, 1383–1390 (2005)
9. Doran, J., Strauss, W.M.: Bio-informatic trends for the determination of miRNA-target interactions in mammals. DNA Cell Biol. 26, 353–360 (2007)
10. Maziére, P., Enright, A.J.: Prediction of microRNA targets. Drug Discov. Today 12, 452–458 (2007)
11. Tjaden, B., Goodwin, S.S., Opdyke, J.A., Guillier, M., Fu, D.X., Gottesman, S., Storz, G.: Target prediction for small, noncoding RNAs in bacteria. Nucleic Acids Res. 34, 2791–2802 (2006)
12. Bazeley, P.S., Shepelev, V., Talebizadeh, Z., Butler, M.G., Fedorova, L., Filatov, V., Fedorov, A.: `snoTARGET` shows that human orphan snoRNA targets locate close to alternative splice junctions. Gene 408, 172–179 (2008)

13. Rehmsmeier, M., Steffen, P., Hochsmann, M., Giegerich, R.: Fast and effective prediction of microRNA/target duplexes. RNA 10(10), 1507–1517 (2004)
14. Zuker, M.: Mfold web server for nucleic acid folding and hybridization prediction. Nucleic Acids Res. 31(13), 3406–3415 (2003)
15. Dimitrov, R.A., Zuker, M.: Prediction of hybridization and melting for double-stranded nucleic acids. Biophys. J. 87(1), 215–226 (2004)
16. Hodas, N.O., Aalberts, D.P.: Efficient computation of optimal oligo-RNA binding. Nucleic Acids Res. 32(22), 6636–6642 (2004)
17. Ding, Y., Lawrence, C.E.: Statistical prediction of single stranded regions in RNA secondary structure and application to predicting effective antisense target sites and beyond. Nucl. Acids Res. 29, 1034–1046 (2001)
18. Ameres, S.L., Martinez, J., Schroeder, R.: Molecular basis for target RNA recognition and cleavage by human RISC. Cell 130(1), 101–112 (2007)
19. Kertesz, M., Iovino, N., Unnerstall, U., Gaul, U., Segal, E.: The role of site accessibility in microRNA target recognition. Nat. Genet. 39(10), 1278–1284 (2007)
20. Akutsu, T.: Dynamic programming algorithms for RNA secondary structure with pseudoknots. Discrete Applied Mathematics 104, 45–62 (2000)
21. Alkan, C., Karakoç, E., Nadeau, J.H., Sahinalp, S.C., Zhang, K.: RNARNA interaction prediction and antisense RNA target search. J. Comp. Biol. 13, 267–282 (2006)
22. Andronescu, M., Zhang, Z.C., Condon, A.: Secondary structure prediction of interacting RNA molecules. J. Mol. Biol. 345(5), 987–1001 (2005)
23. Bernhart, S.H., Tafer, H., Mückstein, U., Flamm, C., Stadler, P.F., Hofacker, I.L.: Partition function and base pairing probabilities of RNA heterodimers. Algorithms Mol. Biol. 1, 3 (2006)
24. Wagner, E.G.H., Simons, R.W.: Antisense RNA control in bacteria, phage, and plasmids. Annu. Rev. Microbiol. 48, 713–742 (1994)
25. Pervouchine, D.D.: `IRIS`: Intermolecular RNA interaction search. Proc. Genome Informatics 15, 92–101 (2004)
26. Aksay, C., Salari, R., Karakoc, E., Alkan, C., Sahinalp, S.C.: taveRNA: a web suite for RNA algorithms and applications. Nucleic Acids Res. 35, W325–W329 (2007)
27. Kato, Y., Akutsu, T., Seki, H.: A grammatical approach to RNA-RNA interaction prediction. In: CMLS 2007: 2007 International Symposium on Computational Models of Life Sciences. AIP Conf. Proc., vol. 952, pp. 197–206 (2007)
28. Mückstein, U., Tafer, H., Hackermüller, J., Bernhard, S.B., Stadler, P.F., Hofacker, I.L.: Thermodynamics of RNA-RNA binding. Bioinformatics 22, 1177–1182 (2006)
29. Argamana, L., Altuvia, S.: fhla repression by Oxys RNA: kissing complex formation at two sites results in a stable antisense-target RNA complex. J. Mol. Biol. 300(5), 1101–1112 (2000)
30. Urban, J.H., Vogel, J.: Translational control and target recognition by Escherichia coli small RNAs in vivo. Nucleic Acids Res. 35(3), 1018–1037 (2007)
31. Sharma, C.M., Darfeuille, F., Plantinga, T.H., Vogel, J.: A small RNA regulates multiple ABC transporter mRNAs by targeting C/A-rich elements inside and upstream of ribosome-binding sites. Genes Dev. 21(21), 2804–2817 (2007)
32. Long, D., Chan, C.Y., Ding, Y.: Analysis of microRNA-target interactions by a target structure based hybridization model. In: Pac. Symp. Biocomput., pp. 64–74 (2008)
33. Lu, Z.J., Mathews, D.H.: Efficient siRNA selection using hybridization thermodynamics. Nucleic Acids Res. 36(2), 640–647 (2008)

34. Tafer, H., Ameres, S.L., Obernosterer, G., Gebeshuber, C.A., Schroeder, R., Martinez, J., Hofacker, I.L.: The impact of target site accessibility on the design of potent siRNAs. Nature Biotech. 26(5) (in press, 2008)
35. Bomfünewerer, A.F., Backofen, R., Bernhart, S.H., Hertel, J., Hofacker, I.L., Stadler, P.F., Will, S.: Variations on RNA folding and alignment: Lessons from benasque. J. Math. Biol. 56, 119–144 (2008)
36. McCaskill, J.S.: The equilibrium partition function and base pair binding probabilities for RNA secondary structure. Biopolymers 29(6-7), 1105–1119 (1990)
37. Hofacker, I., Fontana, W., Stadler, P., Bonhoeffer, S., Tacker, M., Schuster, P.: Fast folding and comparison of RNA secondary structures. Monatsh. Chem. 125, 167–188 (1994)
38. Mathews, D.H., Sabina, J., Zuker, M., Turner, D.H.: Expanded sequence dependence of thermodynamic parameters improves prediction of RNA secondary structure. J. Mol. Biol. 288(5), 911–940 (1999)
39. Mueckstein, U., Tafer, H., Hackermueller, J., Bernhart, S.H., Stadler, P.F., Hofacker, I.L.: Thermodynamics of RNA-RNA binding. Bioinformatics 22(10), 1177–1182 (2006)
40. Gottesman, S.: Micros for microbes: non-coding regulatory RNAs in bacteria. Trends Genet. 21(7), 399–404 (2005)
41. Majdalani, N., Cunning, C., Sledjeski, D., Elliott, T., Gottesman, S.: DsrA RNA regulates translation of RpoS message by an anti-antisense mechanism, independent of its action as an antisilencer of transcription. Proc. Natl. Acad. Sci. U S A 95(21), 12462–12467 (1998)
42. Majdalani, N., Hernandez, D., Gottesman, S.: Regulation and mode of action of the second small RNA activator of RpoS translation, RprA. Mol. Microbiol. 46(3), 813–826 (2002)
43. Prévost, K., Salvail, H., Desnoyers, G., Jacques, J.F., Phaneuf, E., Massé, E.: The small RNA RyhB activates the translation of shiA mRNA encoding a permease of shikimate, a compound involved in siderophore synthesis. Mol. Microbiol. 64(5), 1260–1273 (2007)
44. Valentin-Hansen, P., Eriksen, M., Udesen, C.: The bacterial Sm-like protein Hfq: a key player in RNA transactions. Mol. Microbiol. 51(6), 1525–1533 (2004)
45. Geissmann, T.A., Touati, D.: Hfq, a new chaperoning role: binding to messenger RNA determines access for small RNA regulator. EMBO J. 23(2), 396–405 (2004)
46. Lease, R.A., Cusick, M.E., Belfort, M.: Riboregulation in Escherichia coli: DsrA RNA acts by RNA:RNA interactions at multiple loci. Proc. Natl. Acad. Sci. U S A 95(21), 12456–12461 (1998)
47. Rasmussen, A.A., Eriksen, M., Gilany, K., Udesen, C., Franch, T., Petersen, C., Valentin-Hansen, P.: Regulation of ompA mRNA stability: the role of a small regulatory RNA in growth phase-dependent control. Mol. Microbiol. 58(5), 1421–1429 (2005)
48. Chen, S., Zhang, A., Blyn, L.B., Storz, G.: MicC, a second small-RNA regulator of Omp protein expression in Escherichia coli. J. Bacteriol. 186(20), 6689–6697 (2004)
49. Moeller, T., Franch, T., Udesen, C., Gerdes, K., Valentin-Hansen, P.: Spot 42 RNA mediates discoordinate expression of the E. coli galactose operon. Genes Dev. 16(13), 1696–1706 (2002)
50. Kawamoto, H., Koide, Y., Morita, T., Aiba, H.: Base-pairing requirement for RNA silencing by a bacterial small RNA and acceleration of duplex formation by Hfq. Mol. Microbiol. 61(4), 1013–1022 (2006)

51. Meisner, N.C., Hackermüller, J., Uhl, V., Aszódi, A., Jaritz, M., Auer, M.: mRNA openers and closers: A methodology to modulate AU-rich element controlled mRNA stability by a molecular switch in mRNA conformation. Chembiochem. 5, 1432–1447 (2004)
52. Hackermüller, J., Meisner, N.C., Auer, M., Jaritz, M., Stadler, P.F.: The effect of RNA secondary structures on RNA-ligand binding and the modifier RNA mechanism: A quantitative model. Gene 345, 3–12 (2005)

A Symmetry-Free Subspace for *Ab initio* Protein Folding Simulations

Xiangchao Gan[1], Leonidas Kapsokalivas[1], Andreas A. Albrecht[2], and Kathleen Steinhöfel[1]

[1] King's College London, Department of Computer Science
Strand, London WC2R 2LS, UK
[2] University of Hertfordshire, Department of Computer Science
Hatfield, Herts AL10 9AB, UK
{xiang.gan, kathleen.steinhofel, leonidas.kapsokalivas}@kcl.ac.uk

Abstract. A*b initio* protein structure prediction usually tries to find a ground state in an extremely large phase space. Stochastic search algorithms are often employed by using a predefined energy function. However, for each valid conformation in the search phase space, there are usually several counterparts that are reflective, rotated or reflectively rotated forms of the current conformation, imprecisely called isometric conformations here. In protein folding, these isometric conformations correspond to the different rotation states caused by admissible protein structure transitions. In structure prediction, these isometric conformations, owning the same energy value, not only significantly increase the search complexity but also degrade the stability of some local search algorithms. In this paper, we will prove that there exists a subspace that is unique (no two conformations in the space are isometric) and complete (for any valid conformation, there exists a corresponding conformation in the subspace that is a reflective or rotated form of it). We demonstrate that this subspace, which is about 1/24 of the conventional search space in the 3D lattice model and 1/8 in the 2D model contains the lowest energy conformation, and all other isometric lowest energy forms can then be obtained by protein rotation. Our experiments show that the subspace can significantly speed up existing local search algorithms.

1 Introduction

Proteins support life by carrying out important biological functions that are primarily determined by their structures. Methodological advances in DNA sequencing resulted in a dramatic increase of data about protein sequences, whereas only a small portion of protein sequences have their structure experimentally determined exactly. Thus, structure prediction is an important first step of the sequence-to-structure-to-function paradigm and has been widely applied to protein design, Biocatalysis and Bioengineering.

Given a polypeptide chain and the molecular potential, how can one find the native state conformation? *Ab initio* methods represent a physical approach to predict the structure of a target protein only from its amino acid sequence. The methods are based on an energy function, which is often the contact potentials, and the native state conformation of a protein is the free energy minimum for the amino acid sequence. Computational methods are employed to find the global minimum of the energy function for the target protein. *Ab initio* methods often need a mathematical model to represent a protein. The lattice model is one of the most simple and popular options. Though the simplified lattice model clearly loses the details of protein structures and functions, it does not lose the physical essence of protein folding features, such as the existence of cooperativity and the primary sequence pattern that determines its uniquely defined native 3D conformation.

Even in the simplest lattice model, the protein folding problem has been recognized to be "NP-complete" [1] and is therefore assumed to be not solvable in polynomial time. In most cases, the number of conformations in the search space is extremely large and it is impossible to traverse all the conformations in order to find the optimum solution. Attempts to alleviate the difficulty have been mainly into two directions. One is using powerful optimization methods, such as simulated annealing [2, 3], genetic algorithm [4, 5], Monte Carlo-minimization [4] and Tabu search [6]. The other one is to develop appropriate move sets.

In this paper, different from the above two classes of methods, we try to reduce the conventional phase space instead. By phase space, we mean all 3D protein structures regardless of the types of the amino acids. A big advantage of our method is that almost all existing search algorithms can benefit from this reduction on the search space. Our method comes from our investigation to the symmetry in the conventional phase space of protein folding.

In protein folding space, for each valid conformation, there are often several counterparts that are reflective, rotated or reflectively rotated forms of the current conformation, imprecisely called isometric conformations here. These isometric conformations correspond to the different rotation states caused by the protein movement. The isometric conformation exists in almost all available protein models, such as lattice model, off-lattice model and all-atom models.

In the paper, we will prove that there exists a subspace where each conformation is unique (no reflective or rotated forms for any valid conformation) and complete (for any valid conformation, there is a corresponding conformation in this subspace that is a reflective or rotated form of it). We demonstrate that the subspace, which about 1/24 of the conventional search space in the 3D lattice model and 1/8 in the 2D model, contains the lowest energy conformation, and all other isometric conformations can then be obtained using some simple mappings. Our experiments show that the subspace search method can significantly improve existing search algorithms.

2 Stochastic Search Algorithms for Protein Folding

Energy landscapes are a convenient tool to analyze protein folding algorithms. Formally, a landscape is denoted by $L=(f,u,S)$, where f is the energy function to optimize and u is the modification operator (often called Move Set) applied to elements of the search space S. The landscape heavily depends on the move set, which in turn depends on the structure representation. In protein structure prediction algorithms, there are two kinds of representation mechanism that are widely used.

2.1 Available Structure Representations

3D Coordinates

In 3D protein structure representation, 3D coordinates are no doubt the most popular representation. The method can precisely show the distribution of each amino acid. A large number of protein storage and visualization methods, such as PDB format [7, 8], and search algorithms [9] are using this representation. However, the method is very sensitive to coordinate shifting. For example, by shifting the conformation in Fig. 1a we can get Fig.1b. In the 3D coordinate presentation, both structures are different. Thus, when used in search algorithms, 3D coordinate presentation often needs to employ some extra control in order to alleviate redundancies arising from coordinate shifting.

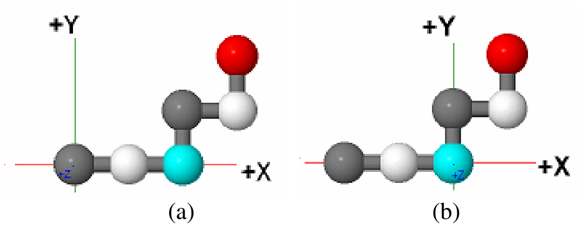


Fig. 1. The 3D coordinate representation of a protein in the lattice model

Torsion Angle Representation (TAR)

In a 3D protein structure, if the length of the bond between two adjacent residues is known, one can define any unique conformation by only providing the torsion angles. In a lattice model, at each point, the chain can only turn 90° left or right, turn 90° up or down or continue ahead relative to the orientation of last bond. For simplicity, we denote these directions with char L, R, U, D and F, respectively. By adding char B, we indicate the backward direction when necessary, and we use six chars F, L, R, U, D and B to represent all the possible conformations of a protein in the lattice model. In addition, this torsion angle representation is independent of the coordinate shifting. For example, the two conformations in Fig.1 are both represented by "FFLRL". Furthermore, the torsion angle representation can be easily transformed into 3D coordinates when necessary. The torsion angle representation has diverse applications in energy landscape analysis algorithms, especially the ones based on the pivot move set.

In theory, if the torsion angle between the residues in a 3D protein structure are classified by discrete states, the alphabet representation can be used to represent any protein model, e.g. such as the off-lattice model [10, 11]. The alphabet representation of real protein structures has already been intensely investigated and widely applied to protein structure matching and browsing [12, 13]. For simplicity, in this paper, we

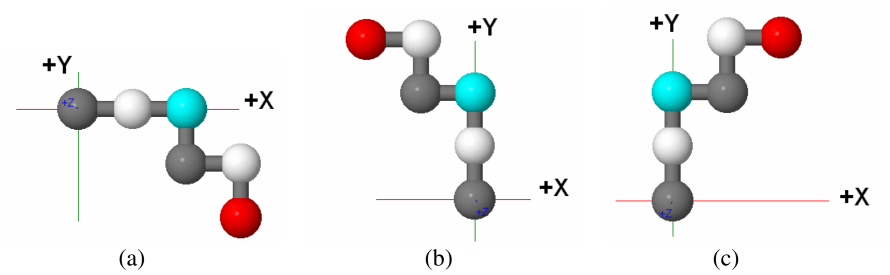


Fig. 2. Three different isometric conformations of the protein from Fig.1.a that have different relative string representations. a) a reflective form of Fig.1.a with TAR as "FFRLR", b) a rotated form of Fig.1a with TAR as "LFLRL" and c) a reflectively rotated form of Fig.1a with TAR as "LFRLR".

represent our algorithm by using the lattice model. However, our conclusions can be easily extended to other models or real proteins by using TAR.

Though torsion angle representation can avoid the redundancy of coordinate shifting, there are still plenty of redundant forms for each protein conformation; for example, the isometric conformation we mentioned before. In structure prediction, these isometric conformations own the same energy value and significantly increase the search complexity.

2.2 Move Sets for Protein Folding

Pivot Move Set

When using the relative string representation, the pivot move set [14, 15] can be simply described by changing one and only one char in a relative string representation of a conformation in order to obtain a new conformation. Since we cannot guarantee that the new conformation is self-avoiding, we must discard the invalid conformations or utilize additional strategies to avoid invalid conformations [16]. An example of pivot move is given in Fig.3.

Pull Move Set

In polymer physics, the motion of a mobile polymer chain moving through a confining environment is governed by slack entering at the ends of the polymer and diffusing along its length [17]. Lesh *et al.* proposed the pull move set [18] based on this observation and the experiments show that the pull move set has a better performance than the pivot move set [18, 19]. The pull move set can guarantee that the new conformation is always valid.

Move Set Comprised of Several Specific Local Mutations

In some papers [20-22] three types of local chain moves are used as the a move set: one-bead terminal swing, one-bead corner flip, and two-bead crankshaft moves. Liang and Wong [23] extended the above move set with an extra *k*-point mutation.

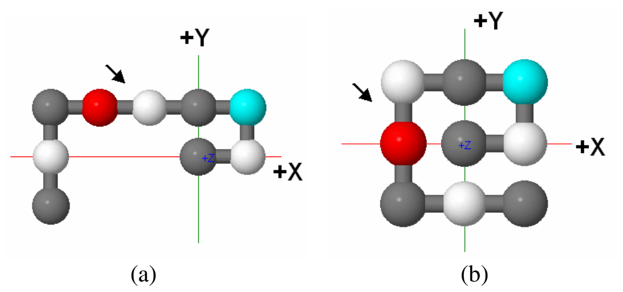


Fig. 3. An example for the pivot move set: with a single pivot move applied to the bond between residue 5 and residue 6 (from "F" to "L"), conformation *a* changes to conformation *b*

2.3 Stochastic Algorithms

Many stochastic search algorithms are a variant of or based on the Metropolis Monte Carlo (MMC) algorithm. MMC algorithms starts from a random conformation S_1 with energy E_1 and then make a single random modification in order to obtain a new conformation $S_2 = \mathrm{u}(S_1)$ with energy E_2 . If $E_2 \leq E_1$, then the transition is accepted; otherwise, acceptance of the transition if

$$Rnd < \exp(\frac{E_1 - E_2}{c_k}),$$

where *Rnd* is a random number between 0 and 1. If c_k is gradually decreased (cooled), the method is called simulated annealing.

3 Proposed Subspace

It is well known that proteins have often a fixed orientation when folding. For protein structure presentation in lattice models, we can state the following:

Lemma 1. *In a lattice model, the chain can turn 90° left or right, turn 90° up or down or continue ahead or backward relative to the orientation of the last segment of the chain. We denote these directions with char L, R, U, D, F and B, respectively. The phase space with F being the first element and L being the first non-F element is unique and complete relative to the protein rotation.*

To prove this lemma, we only need to prove two propositions below.

Proposition 1. *The subspace described above is complete, i.e., for any valid conformation there is a corresponding conformation in this subspace that is its reflective or rotated form.*

Proof. Firstly, it is obvious that any conformation representation starting with non-F char can be rotated to the one with F being the first element. An example of rotating a conformation starting with B into the one starting with F is shown in Fig. 4.

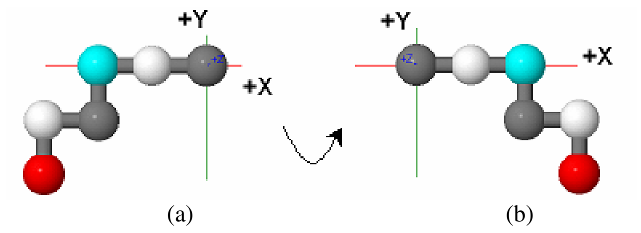


Fig. 4. An example of rotation: a conformation “BFLRL” can be rotated to a conformation “FFRLR”

Secondly, one can see that any conformation representation with F being the first element and the first non-F element being R, U or D can be rotated to a conformation representation with F being the first element and L being the first non-F element. Fig.5 illustrates how a simple protein conformation can be rotated into the one in the proposed subspace and any 3D structure can be treated in the same way. Since B is used only as the first element of conformation representations, we can conclude that any conformation with F being the first element and non-L being the first non-F element can be rotated into a conformation with F being the first element and L being the first non-F element.

Proposition 2. The subspace described above is unique. That is, it is impossible to find two different conformations in this subspace that are isometric.

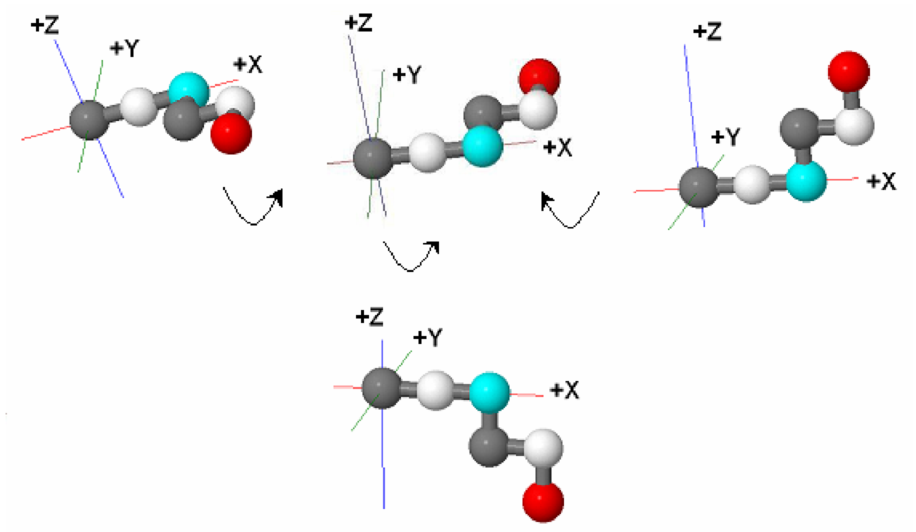


Fig. 5. An example of rotation. Left: Conformation “FFRLR”; Right: Conformation “FFURL”; Bottom: Conformation “ FFDLR”. All of them can be rotated to a conformation “FFLRL”.

Proof. Assume that there are N amino acids. For each residue, there are M different states for each torsion angle. For the proposed subspace, all the possible conformations can be represented as follows, where * denotes M-2 possible choices of U, D, F, L and R:

$$\begin{array}{ccccccc} \multicolumn{7}{c}{\overbrace{\hspace{10em}}^{N-1}} \\ F & L & * & * & \cdots & * & * \\ F & F & L & * & \cdots & * & * \\ & & & \vdots & & & \\ F & F & F & \cdots & F & L & * \\ F & F & F & F & \cdots & F & L \\ F & F & F & F & \cdots & F & F \end{array} \tag{1}$$

Firstly, one can see that two conformations with different number of starting Fs are not isometric. Without loss of generality, we assume that there are two conformations, one with x starting Fs and the other with y starting Fs and $x > y$ as shown below:

$$\begin{aligned} &\text{a. } \overbrace{F \quad F \quad \cdots \quad F}^{x} \quad L \quad * \quad \cdots \\ &\text{b. } \overbrace{F \quad \cdots \quad F}^{y} \quad L \quad * \quad * \quad \cdots \end{aligned} \tag{2}$$

In conformation a, the distance between the first amino acids and the y+1-st amino acids is $\sqrt{y^2+1}$. In conformation b, the distance between the first amino acids and the y+1-st amino acids is y. Since rotation to a conformation does not change the distance between two amino acids inside, conformations a and b are not isometric.

Secondly, we prove that two relative string representations with the same number of starting Fs cannot be rotated to each other. Assume there is a conformation as follows:

$$\overbrace{F \quad F \quad \cdots \quad F}^{x} \quad L \quad * \quad \cdots . \tag{3}$$

The 3D coordinates of the first amino acid are (0,0,0); the x-th amino acid is at (0,0,x) and the $x+1$-th amino acid is at (0,0, x+1). These three amino acids determine a unique plane Γ in the 3D space. If we rotatea a protein in whatever way, plane Γ will also rotate and cannot cover (0,0,0), (0,0,x), (0,0, x+1) any more. Hence, the new conformation obtained by rotation cannot have the form as given in Equation (3). Therefore, we obtain that two relative string representations from the subspace with the same number of starting Fs cannot be rotated to each other.

Lemma 2: *The proposed subspace is a small portion of the whole phase space.*

To prove this, we assume that for fixed-length sequences the ratio of invalid conformations to all *TAR* char combinations is very small and therefore can be omitted, which is true in almost all practical cases. We know that the whole phase space contains $M(M-1)^{N-2}$ conformations. Based on Equation (1), the number of conformations in the proposed subspace is

$$(M-1)^{N-3}+(M-1)^{N-2}+\cdots+(M-1)^{0}+1 \;=\; \frac{(M-1)^{N-2}-1}{M-2}+1 \tag{4}$$

The ratio r of the proposed subspace relative to the whole phase space is

$$r=\frac{(M-1)^{N-2}+M-3}{M-2}\cdot\frac{1}{M(M-1)^{N-2}}=\frac{1}{M(M-2)}+\frac{M-3}{M(M-2)(M-1)^{N-3}}\approx\frac{1}{M(M-2)} \tag{5}$$

Since $N >>1$, the approximation in the above equation holds. In the 2D lattice model, we have $M = 4$ and $r \approx 1/8$. In a 3D case, we have $M = 6$ and therefore $r \approx 1/24$.

4 Results

Since the proposed subspace is much smaller than the whole phase space, all existing algorithms can benefit from this. In this section, we tested the proposed subspace for two stochastic search algorithms.

We have tested the proposed subspace for 3D lattice models. For interacting potentials between amino acids, we use HP and the Miyazawa-Jernigan Matrix [24, 25]. In the HP lattice model, which is often regarded as the simplest model of protein folding, the linear sequence is composed of only two types of amino acids: H(hydrophobic) and P(polar). The energy function is

$$H=\sum_{i<j} e_{v_i v_j}\Lambda(\mathbf{r}_i-\mathbf{r}_j) \tag{6}$$

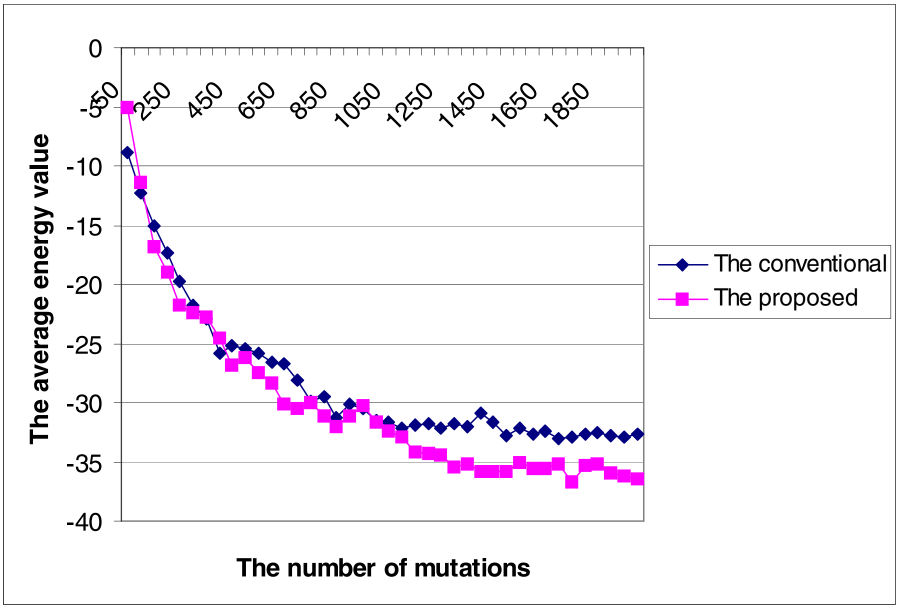
where $\Lambda(\mathbf{r}_i-\mathbf{r}_j)$ is 1 if $\mathbf{r}_i$ and $\mathbf{r}_j$ adjoining non-diagonal lattice sites and 0 otherwise; $e_{v_i v_j}$ can be e_{HH}, e_{HP}, e_{PH} or e_{PP} with e_{HH} = -1 and $e_{HP} = e_{PH} = e_{PP} = 0$. The energy function can be simply described as: -1 for each direct contact of non-bonded hydrophobic-hydrophobic residues and 0 for the others.

In the lattice model with MJ Matrix, the interaction energy for $e_{v_i v_j}$, where v_i and v_j can be any one of the 20 amino acids, taken from the Miyazawa-Jernigan matrix.

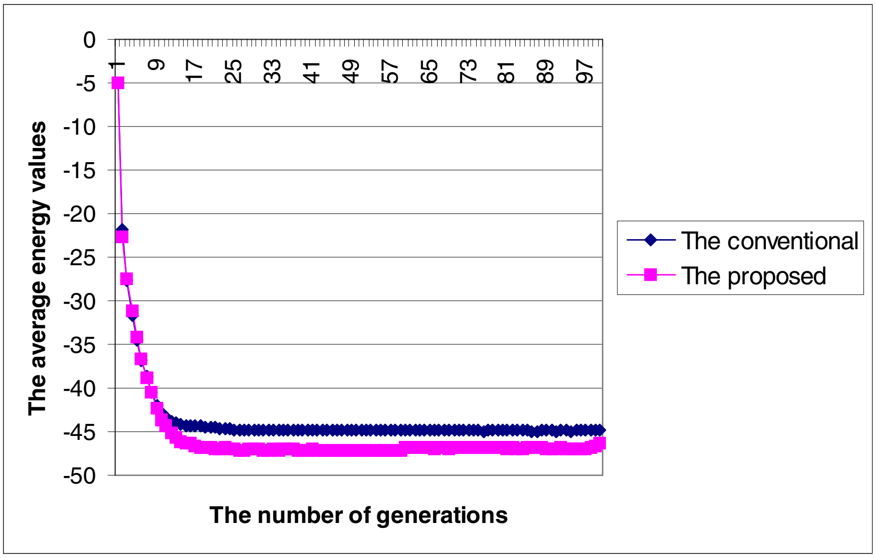
The main factor to be compared is the energy of the current conformation after the same number of successful mutations between two algorithms, one searching the whole phase space, and the other searching the proposed subspace. Since the performance of stochastic search algorithms depends on the random number generator, each experiment is executed multiple times and the average value is taken for the final result.

The following benchmark sequences are analysed in our comparison: in the 3D HP lattice model, a sequence with 64 residues[1] is taken [2, 18]. For the simulated annealing algorithm, the experiment is executed 20 times and the average energy value changes with respect to the number of successful mutations is given in Fig. 4a. The genetic algorithm [2] was also executed 20 times with 100 generations each. There

[1] The sequence is "HHHHHHHHHHHHPHPHPPHHPPHHPPHPPHHPPHHPPHPPHHPPHHPPHPHPHHHHHHHHHHHH".



(a)

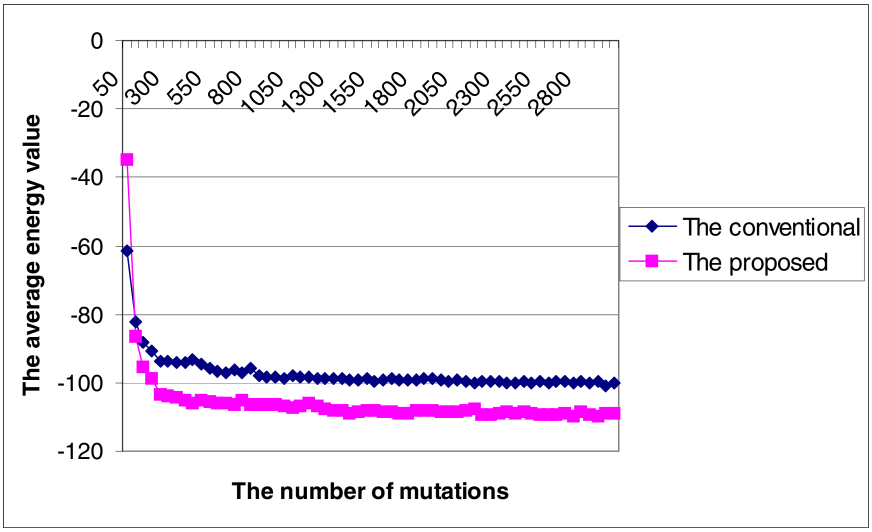


(b)

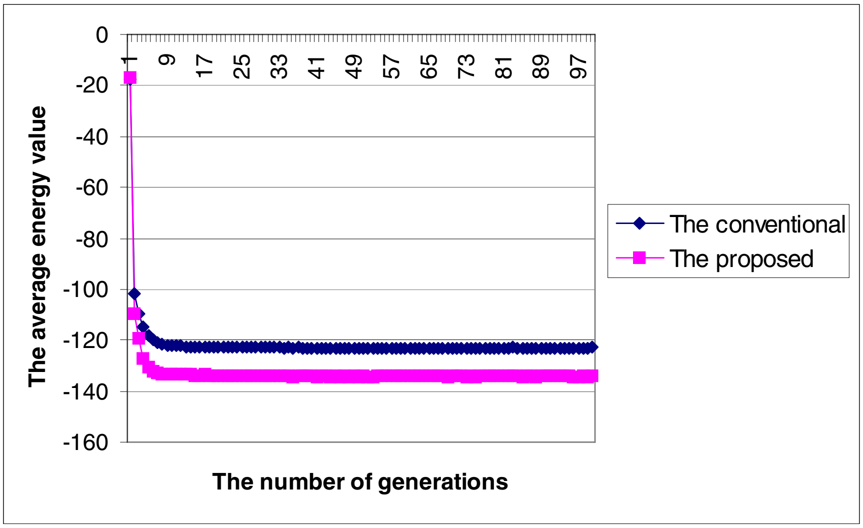
Fig. 6. Performance comparison between algorithms searching in the conventional space and in the proposed subspace by using the 3D HP model: a) the simulated annealing algorithm, b) the genetic algorithm

are 200 successful mutations for each single MC step. The average energy value for every generation after the crossover is given in Fig. 6b.

For the 3D lattice model based upon the MJ matrix, a sequence with 36 residues[2] [26] is tested and the results are given in Fig. 7.



(a)



(b)

Fig. 7. Performance comparison between algorithms searching in the conventional space and the proposed subspace by using the 3D lattice model with MJ matrix: a) the simulated annealing algorithm, b) the genetic algorithm

[2] The sequence is "KMIKDVIERACDHCMHKFVKDVMEHMIKDVCKDCAK".

5 Discussion

It is well known that isometric forms of the protein structure enormously increase the complexity of protein folding prediction. In some papers, the authors try to alleviate the effect of the symmetry on protein folding prediction. A popular idea is to keep a short or long memory of the folding path and to avoid any conformation whose isometric form is in the memory. When simulating a real protein folding process, this idea actually assumes that protein folding somehow keeps track of the folding path. However, there is no proof showing that the protein folding has such an ability. On the contrary, many experiments (see [27] for a review) seem to show that protein folding can be characterized as a "memory-less" stochastic process. Consequently, most of the existing symmetry-reduction algorithms can be seen as computational optimization methods rather than biological simulations. Being completely different from the available algorithms, our method shows that there is no necessity for a protein folding mechanism to remember the folding path in order to avoid isometric conformations.

Though our simulation is successful in the lattice model simulation, the case for real protein folding may be more complicated. For example, it is obvious that there exist several subspaces similar to the one we used. It is likely that the real protein folding does not use one subspace only, but maybe two or several of them in order to arrive in the native state. A more systematic study of real protein folding is under way.

Acknowledgements

We gratefully acknowledge Dr. Gareth Williams from Wolfson Centre for Age-Related Diseases, King's College London, for insightful comments and discussions. The work has been supported by EPSRC grant EP/D062012/1.

References

1. Unger, R., Moult, J.: Finding the lowest free energy conformation of a protein is an NP-hard problem: proof and implications. Bulletin of Mathematical Biology 55, 1183–1198 (1993)
2. Unger, R., Moult, J.: Genetic algorithms for protein folding simulations. Journal of Molecular Biology 231, 75–81 (1993)
3. Chen, W.W., Yang, J.S., Shakhnovich, E.I.: A knowledge-based move set for protein folding. Proteins 66, 682–688 (2007)
4. Li, Z., Scheraga, H.A.: Monte Carlo-minimization approach to the multiple-minima problem in protein folding. Proceedings of the National Academy of Sciences of the United States of America 84, 6611–6615 (1987)
5. Steinbach, P.J.: Exploring peptide energy landscapes: a test of force fields and implicit solvent models. Proteins 57, 665–677 (2004)
6. Paluszewski, M., Hamelryck, T., Winter, P.: Reconstructing protein structure from solvent exposure using tabu search. Algorithms for Molecular Biology 1, 20 (2006)
7. Bernstein, F.C., Koetzle, T.F., Williams, G.J., Meyer Jr., E.F., Brice, M.D., Rodgers, J.R., Kennard, O., Shimanouchi, T., Tasumi, M.: The Protein Data Bank: a computer-based archival file for macromolecular structures. Journal of Molecular Biology 112, 535–542 (1977)

8. Berman, H., Henrick, K., Nakamura, H., Markley, J.L.: The worldwide Protein Data Bank (wwPDB): ensuring a single, uniform archive of PDB data. Nucleic Acids Research 35, 301–303 (2007)
9. Chan, H.S., Dill, K.A.: Transition states and folding dynamics of proteins and heteropolymers. The Journal of Chemical Physics 100, 9238–9257 (1994)
10. Helling, R., Li, H., Melin, R., Miller, J., Wingreen, N., Zeng, C., Tang, C.: The designability of protein structures. Journal of Molecular Graphics and Modelling 19, 157–167 (2001)
11. Park, B.H., Levitt, M.: The complexity and accuracy of discrete state models of protein structure. Journal of Molecular Biology 249, 493–507 (1995)
12. Micheletti, C., Seno, F., Maritan, A.: Recurrent oligomers in proteins: an optimal scheme reconciling accurate and concise backbone representations in automated folding and design studies. Proteins 40, 662–674 (2000)
13. Pandini, A., Bonati, L., Fraternali, F., Kleinjung, J.: MinSet: a general approach to derive maximally representative database subsets by using fragment dictionaries and its application to the SCOP database. Bioinformatics 23, 515–516 (2007)
14. Madras, N., Sokal, A.D.: The pivot algorithm: A highly efficient Monte Carlo method for the self-avoiding walk. Journal of Statistical Physics 50, 109–186 (1988)
15. Madras, N.N., Slade, G.D.: The Self-avoiding Walk. Birkhäuser, Boston (1993)
16. Toma, L., Toma, S.: Contact interactions method: a new algorithm for protein folding simulations. Protein Science 5, 147–153 (1996)
17. de Gennes, P.G.: Reptation of a Polymer Chain in the Presence of Fixed Obstacles. The Journal of Chemical Physics 55, 572–579 (1971)
18. Lesh, N., Mitzenmacher, M., Whitesides, S.: A complete and effective move set for simplified protein folding. In: Proceedings of the Seventh Annual International Conference on Research in Computational Molecular Biology. ACM, Berlin, Germany (2003)
19. Böckenhauer, H.-J., Bongartz, D.: Protein folding in the HP model on grid lattices with diagonals. Discrete Applied Mathematics 155, 230–256 (2007)
20. Dill, K.A.: Polymer principles and protein folding. Protein Science 8, 1166–1180 (1999)
21. Sali, A., Shakhnovich, E., Karplus, M.: Kinetics of protein folding. A lattice model study of the requirements for folding to the native state. Journal of Molecular Biology 235, 1614–1636 (1994)
22. Miller, R., Danko, C.A., Fasolka, M.J., Balazs, A.C., Chan, H.S., Dill, K.A.: Folding kinetics of proteins and copolymers. The Journal of Chemical Physics 96, 768–780 (1992)
23. Liang, F., Wong, W.H.: Evolutionary Monte Carlo for protein folding simulations. The Journal of Chemical Physics 115, 3374–3380 (2001)
24. Miyazawa, S., Jernigan, R.L.: Estimation of effective interresidue contact energies from protein crystal structures: quasi-chemical approximation. Macromolecules 18, 534–552 (1985)
25. Miyazawa, S., Jernigan, R.L.: Residue-residue potentials with a favorable contact pair term and an unfavorable high packing density term, for simulation and threading. Journal of Molecular Biology 256, 623–644 (1996)
26. Betancourt, M.R., Thirumalai, D.: Pair potentials for protein folding: choice of reference states and sensitivity of predicted native states to variations in the interaction schemes. Protein Science 8, 361–369 (1999)
27. Mirny, L., Shakhnovich, E.: Protein folding theory: from lattice to all-atom models. Annual Review of Biophysics and Biomolecular Structure 30, 361–396 (2001)

Comparative Analysis of Disulfide Bond Determination Using Computational-Predictive Methods and Mass Spectrometry-Based Algorithmic Approach

Timothy Lee* and Rahul Singh

Department of Computer Science, San Francisco State University, 1600 Holloway Avenue, San Francisco, CA 94132-4025, U.S.A.
{timtlee@sfsu.edu, rsingh}@cs.sfsu.edu

Abstract. Identifying the disulfide bonding pattern in a protein is critical to understanding its structure and function. At the state-of-the-art, a large number of computational strategies have been proposed that predict the disulfide bonding pattern using sequence-level information. Recent past has also seen a spurt in the use of Mass spectrometric (MS) methods in proteomics. Mass spectrometry-based analysis can also be used to determine disulfide bonds. Furthermore, MS methods can work with lower sample purity when compared with x-ray crystallography or NMR. However, without the assistance of computational techniques, MS-based identification of disulfide bonds is time-consuming and complicated. In this paper we present an algorithmic solution to this problem and examine how the proposed method successfully deals with some of the key challenges in mass spectrometry. Using data from the analysis of nine eukaryotic Glycosyltransferases with varying numbers of cysteines and disulfide bonds we perform a detailed comparative analysis between the MS-based approach and a number of computational-predictive methods. These experiments highlight the tradeoffs between these classes of techniques and provide critical insights for further advances in this important problem domain.

1 Introduction

Cysteine residues have a property unique among the amino acids, in that they can pair to form a covalent bond, known as a disulfide bond. These bonds are so named because they occur when each cysteine's sulfhydryl group becomes oxidized following the reaction

$$\text{S-H} + \text{S-H} \rightarrow \text{S-S} + 2\text{H} \tag{1}$$

Because disulfide bonds impose length and angle constraints on the backbone of a protein, knowledge of the location of these bonds significantly constrains the search-space of possible stable tertiary structures which the protein folds into. The disulfide bond pattern of a protein also can have an important effect on its function. For example, in [1] it is shown that the sterical structure formed by disulfide bonds in ST8Sia

* Equal contributors.

IV is critical for it to catalyze the polysialylation of NCAM, the neural cell adhesion molecule. NCAM has an important role in neuronal development and regeneration.

At the state-of-the-art, techniques for disulfide bond determination can be classified into three primary groups: (1) Crystallographic techniques producing high-resolution three dimensional structures of proteins, where the disulfide bonds can be observed directly. (2) Algorithmic techniques that *predict* (or infer) the disulfide connectivity based on sequence data. (3) Mass-spectrometry-based techniques that detect disulfide bonded peptides by analyzing a mixture of peptides obtained by targeted digestion of an intact protein.

Crystallographic methods can be used to study a subdomain of the protein that is sufficiently soluble and may form crystals. However, such methods can rarely be used in medium or high-throughput settings. Consequently, in the recent past, significant attention has been given to computational methods that can predict disulfide connectivity based on sequence information alone [2-10, 27]. An important advantage of these predictive methods lies in the fact that they require only sequence-level data to make predictions. Recent results in this area have reported high accuracies with Q_p values (fraction of proteins in the test set with disulfide connectivity correctly predicted) in the 70 – 78% range. These methods also report high Q_c (sensitivity) values. However, in interpreting, extrapolating, and understanding these performance values, the following considerations are especially critical:

1. Most of the reported results use a dataset called SP39 of non-redundant sequences derived from the SWISS-PROT database (release no. 39) proposed in [5]. To ensure quality, this dataset only includes sequences containing information from PDB for which intra-chain disulfide bonds are annotated. Further, disulfide assignments described as "by similarity", "probable", or "potential" are excluded. Two issues emanating from the use of this standard dataset need to be emphasized. On one hand, it undeniably leads to uniformity and ease in comparing results. However, it also invariably leads to methods being optimized in context of a fixed standard. For this reason alone, care needs to be taken in extrapolating the performance on SP39 to arbitrary data. It must be noted, that certain methods (such as [7, 8]) have used multiple datasets in addition to SP39, in assessing performance.
2. In many methods, learning and testing have often been done in a cross-validated settings rather than involving independent datasets. This leaves open the issue of training bias and its possible impact on the performance of these methods on completely novel datasets.
3. In SP39 as well as the other datasets used, only a limited disulfide-bonding topology (consisting of intra-bonded cysteines) is considered. This has putative implications regarding the applicability of these methods to more complex bonding topologies.

In contrast to computational-predictive methods, mass spectrometric approaches, which involve direct measurements, provide a conceptually different approach to disulfide bond determination. The choice between these two classes of methods requires studying the tradeoffs between the *model-and-predict* strategy used in predictive methods and the *direct measurement* principle underlying mass spectrometric techniques. The investigations presented in this paper are motivated by the above discussion. Specifically, we pursue two goals. *First*, we investigate some of the key computational challenges associated with mass-spectrometry-based disulfide bond determination. *Second*, through experimental studies conducted on nine eukaryotic Glycosyltransferases with varying numbers of cysteines and disulfide bonds, we investigate the aforementioned tradeoffs between purely predictive methods and an MS-based approach.

2 Previous Work

A variety of techniques have been proposed for determining disulfide bonding patterns including crystallographic approaches, computational predictive strategies, and methods combining mass spectrometric and algorithmic techniques. A comprehensive review of algorithmic methods for this problem is presented in [11]. Broadly speaking, algorithmic approaches can be classified into two major classes: (1) techniques that *predict* (or infer) the disulfide connectivity based on sequence data, and (2) techniques that algorithmically analyze a mixture of peptides obtained by targeted digestion of an intact protein using mass spectrometry and thereby seek to *detect* disulfide bonds.

Techniques based on sequence data are based on characterizing a heuristically defined local sequence environment and address one of two correlated problem formulations. The first, *residue classification*, involves distinguishing the bonded cysteine residues from the free residues. Early techniques for residue classification either analyze the statistical frequency of amino acid residues in neighborhoods around the cysteines [12] or encode the local sequence environment of residues and solve the classification problem using machine learning methods in a supervised setting [13, 14]. Other methods [15], have combined the use of both local and global descriptors and/or hybrid classifiers [16]. While it is difficult to directly compare the prediction performance of these methods due to differences in datasets, most descriptor and classifier choices in the aforementioned works lead to prediction accuracies of greater than 78% with [16] reporting prediction accuracy of 87.4% on chains containing two or more cysteines and 88% overall accuracy. Other techniques, such as [12,15] have also reported prediction accuracies in the 82% - 84% range.

The second formulation, *connectivity prediction*, employs techniques that seek to define the complete disulfide connectivity pattern of a protein. In [17], the connectivity pattern was determined by first constructing a completely connected graph *G*. Four different formulations of contact potential were used for weighting the edges and the disulfide connectivity was defined as the solution of the maximum weight perfect matching problem on *G*. In [18], a recursive neural network (RNN) was used for scoring undirected graphs that represent connectivity patterns by their similarity to the correct graph. The idea of RNN formed the basis of the DISULFIND prediction server [19]. In [20] the notion of utilizing the specificities in the sequence neighborhood of cysteines was extended to take advantage of cysteine distributions in secondary structure elements. In [8], the chain classification problem was addressed using evolutionary information and kernel methods. Other approaches to this problem include the use of cysteine separation profiles [9, 10] and comparisons with an annotated database, as done in the CysView server [22]. The highest Q_p scores (fraction of correctly assigned proteins) reported were in the 70% – 78% range [8, 22].

The basic strategy for determining disulfide bonds using mass spectrometry consists of the following steps: *First*, the protein of interest is cleaved in its non-reduced state between as many of the cysteine residues as possible using proteases like trypsin or chymotrypsin. *Second*, the resultant peptides, including disulfide-linked peptides, are separated and analyzed by electrospray ionization (ESI) or matrix-assisted laser desorption/ionization (MALDI). These mass spectrometry techniques allow peptide and protein molecular ions to be put into the gas phase without fragmentation. The analysis is a two step process and involves measuring the mass-to-charge (*m*/*z*) ratio of the ionized disulfide-linked peptides (also called the parent or precursor ion) along with measurement of the *m*/*z* ratio of the product ions. Subsequently, the peptides' ions are fragmented to confirm the sequence identity of the disulfide-linked peptides and thereby the location of one of the protein's disulfide bonds.

In spite of the seeming simplicity of this process, the determination of disulfide bonds using mass spectra is complex. This is because the number of possible disulfide bonding models grows rapidly with the number of cysteines and the number of expected disulfide bonds. Furthermore, measurements of fragment-based bonding patterns can be influenced by noise and need to be coalesced into an overall connectivity pattern that is physically consistent. These issues constitute the key challenges for an algorithmic approach that seeks to utilize mass-spectrometric data for disulfide-bond determination.

3 Disulfide Bond Determination Using Mass Spectrometry Data

Based on the above discussion, we identify three main computational challenges: (1) efficiently searching the combinatorial space of peptides and fragmented peptides to determine (mass-based) correspondences with the mass spectrum/tandem mass spectrum. These correspondences would indicate putative disulfide bonds. (2) Ranking and filtering the correspondences to exclude effects of noise. (3) Determining the global pattern of disulfide bonds for the molecule.

3.1 Basic Definitions and Computational Formulation

A *cysteine-containing peptide C* is a defined to be a peptide that has at least one of its amino acids identified as a cysteine residue. A *disulfide bonded peptide structure* consists of one or more cysteine-containing peptides that contain one or more disulfide bonds. The *disulfide bond mass space DMS* = $\{Dm_i\}$ is the set of masses of every possible disulfide bonded peptide structure for a protein. During LC/ESI, precursor ions are generated. A *precursor ion* is a peptide or disulfide bonded peptide structure that has been ionized, so that a charge to mass ratio associated with it appears as part of the mass spectrum of a protein. A *precursor ion mass list PML* = $\{Pm_j\}$ is the set of numbers that represent the masses of the precursor ions obtained from a LC/ESI-MS/MS experiment. The *PML* is a representation of the mass spectrum that has been processed to remove noise from the experimental procedure, and has been expanded to address uncertainties in the charge state of the ion, as well as neutral loss. A *precursor match* between *DMS* and *PML* occurs when the difference between their values is below a predefined threshold. We denote the set of precursor matches as the *Initial Match IM* = between *PML* and *DMS*. During the MS/MS step, peptides undergo collision-induced dissociation (CID), generating peptide fragments. The fragments produced are mostly either b-ions containing the peptide's N-terminus or y-ions containing its C-terminus. A *cysteine-containing peptide fragment* is a peptide fragment that has at least one of its amino acids identified as a cysteine residue. A *disulfide bonded peptide fragment structure* consists of one or more cysteine-containing peptide fragments that contain one or more disulfide bonds. The *disulfide bonded fragment mass space FMS* = $\{Fm_i\}$ is the set of the masses of every disulfide bonded fragment structure that can be obtained from a disulfide bonded peptide structure. A *MS/MS mass list TML* = $\{Tm_i\}$ is the set of numbers corresponding to the masses of the peptide fragments obtained from a precursor ion in a LC/ESI-MS/MS experiment. A *MS/MS match* between *TML* and *FMS* occurs when the difference between the corresponding elements of TML and FMS is less than a predefined threshold. We denote the set of MS/MS matches as the *Confirmed Match CM* between *TML* and *FMS*.

The number of elements in a Confirmed Match is an indication of the degree to which the LC/ESI-MS/MS data supports the presence of a particular disulfide bonded peptide fragment. In our case the identification of the peptide structure shows us which cysteine residues are participating in disulfide bonds. Thus, by aggregating the all the Confirmed Matches for a protein analyzed by LC/ESI-MS/MS, we can arrive at the overall disulfide bond pattern for the protein. In doing so, we need to ensure not only that the overall connectivity pattern is physically consistent (no cysteine participates in more than one disulfide bond) but also that the pattern is the most likely one given the data. The primary challenges for determining the disulfide-bond connectivity therefore include:

1. Finding an efficient way to obtain the *initial match IM* between the *PML* and the *disulfide bond mass space DMS*.
2. For each initial match, efficiently determining the *confirmed match* between the *disulfide bonded fragment mass space FMS* and the *TML*.
3. Aggregating the *confirmed matches* into a weighted graph enabling the computation of the overall disulfide bond pattern.

3.1.1 Determining the Initial Match

We first examine how to construct the DMS for a disulfide bond topology consisting of an arbitrary number of peptides. For this analysis, let k denote the number of sites where an arbitrary protein A can be cleaved with a certain protease. As a result, A is divided into k+1 peptides. For most proteins and proteases, each peptide contains at most one cysteine residue. These peptides can form *interbonded* disulfide bonds with other peptides. If a peptide contains two or more cysteines, an *intrabonded* disulfide bond may be present. For this case, the time needed to construct of the DMS equals the time required to search each peptide to determine which peptides contain two or more cysteine residues. Because there are k+1 peptides, the overall complexity to construct the DMS for this topology is $O(k)$. Extending this line of argument, it can be shown that for the n-peptide case, the mass space requires $O(k^n)$ time to compute.

This complexity can be reduced if the data structure used to construct and search the DMS does not require computing the mass of every member of the DMS. Such a data structure can be constructed by identifying every possible disulfide bonded peptide structure and then storing each as an element in a pre-computed state. For example, if a protein has the three cysteine-containing peptides p1, p2, and p3, this space consists of {p1+p2, p1+p3, p2+p3}. Here, each element contains the amino acid sequence of each unique peptide combination. The next step is to arrange these elements in such a way that they are approximately sorted by mass. This can be done without computing the mass of each peptide combination by noting that the number of amino acids in a peptide is directly proportional to its mass. Based on this idea, we store the DMS in a hash table with its key the number of amino acids in the peptide combination. Next the elements of the PML must be converted to an equivalent number of amino acids in order to index into the DMS for matches. This can be done by use of the *expected match index*, as defined below:

Definition 1. *The Expected Match Index* i_e is defined as the number used to index into a sorted or approximately sorted data structure to arrive at the region where a match is likely to be found. The match index is constructed for mass tables that represent strings of amino acids a by $i_e = m_j/m_e$, where m_j is a value from a mass list and m_e is the *expected amino acid mass*. We defined the expected amino acid mass in [23] as the weighted mean of all 20 amino acids, i.e, $m_e = \sum_i w_i m(a_i)$, where $\{w_i\}$denotes the relative abundance of each amino acid, and $m(a_i)$ is the mass of an amino acid residue. Using published values for masses and relative abundances of each amino acid [24], we obtain $m_e = 111.17$ Da, with a weighted standard deviation of $\sigma_e = 28.86$ Da.

For each member of the PML, an index is calculated by dividing the member by i_e. These indices are then used to access the corresponding buckets in the hash table. Finally, the mass of each peptide pair in a bucket is computed and compared with the corresponding peak value to determine a match. Because only the disulfide bonded peptide configurations that are indexed have their masses computed, we call this technique *generative indexing*. As discussed earlier, the construction of the mass space requires $O(k^p)$ time, where k is the number of cysteine-containing peptides following proteolytic digestion, and p is the maximum number of peptides in a disulfide bonded peptide structure. Thus the overall time complexity of this step is $O(k^p+|DMS|+|PML|)$. In nature, p greater than 3 are rarely observed, and p greater than 5 has not been reported to our knowledge. Consequently the effective complexity of this step is cubic.

3.1.2 Determining the Confirmed Match

Consider a peptide with intrabonded cysteines. For the general case, the total number of y- and b-ions combined is a constant and depends only on the number of amino acid residues in the peptide, denoted $||p||$. Thus, the construction of the disulfide bonded fragment mass space for this case requires $O(||p||)$ time. We note that the expected match index can be used to improve on this time complexity by only considering those elements that are likely to match an element of the TML. For an interbonded pair of peptides, let p1 denote a peptide with its set of possible y-ions denoted y1 and b-ions denoted b1, and y2 and b2 denotes the y-ions and b-ions for peptide p2. Since p1 and p2 are in a disulfide bond, four types of fragments may occur: y1+y2, y1+b2, b1+y1, and b1+b2. A simple way to compute and display the FMS is to generate four tables based on these four types of fragment combinations. Then, for this

MS/MS mass table the mass of any element T[i, j] equals m(i) + m(j) - 2 Da. If the two peptides consist of $||p1||$ and $||p2||$ amino acid residues, respectively, the total number of elements to compute is $(||p1||+1)(\ ||p2||+1)$. If the ions used to form the mass tables are arranged in order of increasing number of amino acids, the matrices will be sorted. Again, the expected match index can be used to generate only those elements that are likely to match an element of the TML. These elements correspond to a diagonal region in a mass table. This leads to a time complexity to search for a match of $O\ (||p||)$, if $||p1|| \approx ||p2||$. The extension of this analysis to construct the FMS for a n- peptide disulfide bonded structure is now straightforward. The FMS for a n-peptide structure consists of 2^n n-dimensional sorted tables. Given an expected match index value, the region where matches are likely to be found has n-1 dimensions. Thus, the time complexity to determine a match with an element of the TML is $O(2^n\ ||p||^{n-1})$. Based on the previous discussion on the number of fragments that have been observed in the disulfide bonded peptide structure, the effective complexity reduces to cubic.

3.1.3 Aggregating Results to Compute the Overall Disulfide Bond Pattern

The output of the previous step is a collection of confirmed matches between pairs of cysteines. Let the *confirmed match* $CM_{a,b}$ denote a match obtained from a disulfide bonded peptide structure with cysteines C_a and C_b. To convert each $CM_{a,b}$ into a single number that is assigned to the weight of an edge between the pair, we apply the notion of the *match ratio r*, which is defined as the number of matches divided by the size of the tandem mass spectrum, i.e. $r = |CM|/|TML|$. To compute the overall disulfide connectivity, we construct a weighted graph *G* where each vertex represents a cysteine residue in the protein. If there is a match ratio $r_{a,b}$ that is greater than 50%, this number is assigned to the weight of the edge between vertex a and vertex b. Thus each edge represents a Confirmed Match for a disulfide bond between a pair of cysteine residues. Subsequently, the maximal weight matching problem is solved on this graph (using the algorithm by Gabow [25]) to obtain the overall disulfide-bond topology. The complexity of this step is $O(|C|^3)$. This leads to an overall cubic complexity for our method, which we call MS2DB.

4 Experimental Evaluation

The data used in experiments consisted of the primary sequences (obtained from the Swiss-Prot database [24]) and the DTA files obtained from LC/MS/MS analysis using an LCQ ion trap mass spectrometer (Finnigan, San Jose, CA) for nine eukaryotic Glycosyltransferases with varying numbers of cysteines and disulfide bonds. For each protein, all DTA files are used collectively from an LC run. The proposed method was used with the following parameters: bond mass tolerance $bm_t = 3.0$ Da, maximum peak width $p_w = 2$ Da, threshold t = 2% of the maximum intensity, and the limit l = 50 peaks. Further, the MS/MS mass tolerance was set to $fm_t = 1.0$ Da, except when intramolecular bonded cysteines were identified, when a value of 1.5 Da was used. The protease was set to what was used in the actual experiment(s). The number of missed cleavages allowed was set to $m_{max}= 1$. Three different sets of experiments were performed. In the first experiment the gains in efficiency that are achieved by utilizing the generative indexing technique were experimentally studied. In the second experiment, the proposed method was compared with MS2Assign [26], which is a mass spectrometry-based method for determining cross-linkages. In the final experiment, a detailed comparative study was conducted where the disulfide connectivity determination capabilities of the proposed mass spectrometry-based method was compared with three well established methods using the model-and-predict methodology, namely DiANNA [7], DISULFIND [27], and PreCys [28]. The results from each of the methods were analyzed in terms of well established statistical metrics of sensitivity, specificity, accuracy, and Matthew's correlation coefficient.

4.1 Experimental Analysis of the Proposed Approach

To quantify the gains in efficiency achieved by utilizing the generative indexing technique, the fraction of the MS mass space that was actually searched for each of the Glycosyltransferases was determined. For this, the number of mass computations was tracked and divided by the total number of entries in the hash table (i.e. the MS mass space). Fig. 1 (left plot) shows the results obtained. It may be noted that the fraction of the mass space that had to be searched decreased as the number of precursor ions increased, thus underlining the effectiveness of the proposed search strategy. For data obtained after the tandem mass spectrometry step, the efficiency gain was measured by dividing the number of mass computations made by the size of the MS/MS mass space, which is essentially the size of the four tables of b- and y-ion combinations. Fig. 1 (right plot) shows that while a larger fraction of the mass space is accessed by a MS/MS mass peak, a saturation level of about 50% is rapidly achieved. This is because the proposed approach saves mass table entries across searches so that the same element is not recomputed.

In order to quantify the ability of the proposed method to efficiently determine the overall bonding pattern, we first must determine the size of the solution space from which the disulfide bond pattern has to be identified. In Table 1, the first column shows the size of this space if there is no knowledge as to whether any one cysteine is bonded with any other. In other words, the cysteine graph for this protein is fully connected. The second column shows the number of possible patterns that are obtained if all edges with match ratios less than .50 are removed in the cysteine graph.

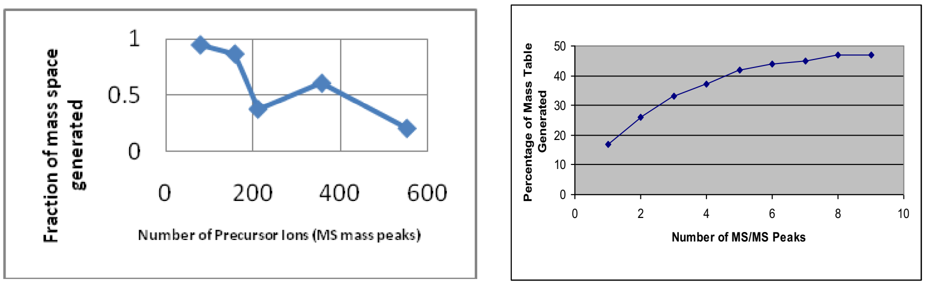


Fig. 1. Experimental analysis of the proposed indexing-based search strategy. The generative indexing approach results in the computation/search of only a fraction of the theoretical mass space.

Table 1. Effectiveness of the proposed approach in reducing the number of disulfide bond patterns that need to be considered for determining the final connectivity

Protein	Number of theoretically possible disulfide bond patterns from fully connected cysteine graph	Number of theoretically possible disulfide bond patterns from cysteine graph with edges for match ratios exceeding 0.50
C2GnT-I	945	67
ST8Sia IV	15	4
FT VII	15	11
Lysozyme	105	61
Lactoglobulin	15	2
FT III	15	10
β1,4-GalT	61	25
Aldolase	124	2
Aspa	124	1

4.2 Comparison with MS2Assign

To compare the proposed method with MS2Assign we identified the DTA files that were used to obtain match ratios for C13 to C59 (true positive identification) and C199 to C413 (false positive identification) of C2GnT-I. The fragment ion m/z portions of the file were then copied to use for the Peak List in MS2Assign. In the true positive identification case, for MS2Assign, the number of matches obtained was 1646 out of 1774 peaks input, giving a match ratio of 0.93. For our method, the corresponding number of matches was 48 out of 50 peaks, giving a match ratio of 0.96. In the false positive identification case, for MS2Assign, the number of matches we obtained was 1791 out of 2169 peaks, giving a match ratio of 0.78, while for our method, the number of matches we obtained was 44 out of 50 peaks, giving a match ratio of 0.72. While preliminary, the results from this study seem to indicate that the accuracy of the proposed approach is indistinguishable from MS2Asssign (the proposed approach performs marginally better in recognizing true positives and scores false positives lower than MS2Assign). However, it should be noted that unlike MS2Assign, the proposed method is fully automated; in MS2Assign the DTA files have to be manually analyzed to identify the mass spectrum and retain the mass values (MS2Assign does not provide such parsing functionality).

4.3 Comparison with Predictive Methods

In this experiment the proposed mass spectrometry-based method was compared with three predictive methods (DiANNA [7], DISULFIND [27], and PreCys [28]) and the results extensively analyzed. The disulfide bonding pattern determined using each method is shown in Table 2. It may be noted that across the entire dataset, using the

Table 2. Dataset and comparison of MS2DB with some prediction servers. The first column displays the name (or abbreviation) of the protein, followed by its Swiss-Prot accession number. Column 3 lists the experimentally determined disulfide bond pattern of each protein. For example for protein C2GnT-I, eight cysteines are engaged in four disulfide bonds, while the remaining three are unbonded. One of these bonds is between the cysteine at amino acid position 59 and the cysteine at amino acid position 413. Columns 4 to 6 show the results for three prediction servers, given the primary sequence of each protein as input. Note that DISULFIND does not support the prediction of proteins with more than 10 cysteines.

Protein (Swiss-Prot ID #)	Number of Cysteines	Disulfide Bond Structure	*DiANNA 1.1*	*DISULFIND*	*PreCys*	MS2DB
C2GnT-I (Q09324)	11	C59-C413 C100-C172 C151-C199 C372-C381 C13, C217, C234 free	C13-C172, C59-C217, C151-C234, C199-C372, C381-C413	Not supported	C59-C381 C100-C372 C151-C172 C199-C413	C59-C413 C100-C172 C151-C199 C372-C381 C13, C217, C234 free
ST8Sia IV (Q92187)	6	C142-C292 C156-C356 C11, C169 free	C11-C156, C142-C292, C169-C356	all free	C142-C356 C156-C292	C142-C292 C156-C356 C11, C169 free
FT VII (Q11130)	6	C68-C76 C211-C214 C318-C321	C68-C321, C76-C211, C214-C318	C76-C318	C68-C76 C211-C214 C318-C321	C68-C76 C211-C214 C318-C321
Lysozyme (P00698)	9	C24-C145 C48-C143 C82-C98 C94-C112 C10 free	C24-C145, C48-C133, C82-C98, C94-C112	C24-C145 C48-C133 C82-C98 C94-C112	C82-C145	C24-C145 C48-C143 C10, C82, C94, C98, C112 free
Lactoglobulin (P02754)	7	C82-C126 C3, C12, C135, C137, C176 free	C12-C137, C82-C176, C126-C135	all free	all free	C82-C126 C3, C12, C135, C137, C176 free
FT III	7	C81-C338 C91-C341, C16, C129, C143 free	C16-C91, C81-C143, C129-C338	none	C81-C91	C81-C338 C91-C341
β1,4-GalT	7	C134-C176 C247-C266 C23, C30, C342 free	C23-C176, C30-C144, C266-C341	none	C134-C247 C176-C266	C134-C176 C247-C266
Aldolase	8	C73, C135, C115, C178, C202, C240, C290, C339 free	C73-C339, C135-C290, C115-C240, C178-C202	none	none	none
Aspa	8	C4, C60, C66, C123, C145, C151, C217, C275 free	C4-C275, C60-C217, C66-C151, C123-C145	none	none	C145-C349
Q_p			0.0	0.0	0.22	0.78

proposed method a Q_p score (representing the fraction of molecules with disulfide bonds correctly identified) of 0.89 was obtained. While DiANNA, DISULFIND, and PreCys are known to perform well on the SP39 dataset, their performance on these nine Glycosyltransferases was significantly inferior.

To further analyze these results, we created connectivity tables for all of the proteins that we studied in a manner similar to what is shown in Table 2. Table 3 is one of the connectivity tables we created. Subsequently the four commonly used metrics of sensitivity, specificity, accuracy, and Matthew's correlation coefficient were calculated.

These four metrics are defined as:

- Sensitivity = TP/(TP+FN)
- Specificity = TN/(TN+FP)
- Accuracy = (TP+TN)/(TP+FP+TN+FN)
- Matthew's correlation coefficient =

$$\frac{TP \times TN - FP \times FN}{\sqrt{(TP+FN)(TP+FP)(TN+FP)(TN+FN)}}$$

In the above formulae the following abbreviations are used: TP (true positive), TN(true negative), FP (false positive), and FN (false negative). In seven out of the nine cases, the metrics for the proposed mass spectrometry-based method were superior those of the predictive methods. However, the proposed method had difficulty with Lysozyme, where two disulfide bonds were observed to occur in a complex topology with one inter-peptide bond sandwiched by cysteines participating in an intra-peptide bond. Currently, MS-based methodologies lack the resolution to disambiguate such patterns. Interestingly however, the predictive methods all performed well for this case. It should also be noted that in practice, researchers consider false negative results to have a more deleterious effect on protein characterization than false positive results. Our results, summarized in Table 4, show that MS2DB generates fewer false negative results than the prediction servers we considered.

Table 3. Connectivity table summarizing validation testing results for two proteins. Below diagonal: β1,4-GalT. Above diagonal: Lactoglobulin. The experimentally determined disulfide bond pattern is shaded in gray. The diagonal is shaded black. Only match ratios greater than 0.5 are included into the table.

	3	12	82	122	135	137	176	Cysteine location
23	■	**TN**	**TN**	**TN**	**TN**	**TN**	**TN**	3
30	**TN**	■	**TN**	**TN**	**TN**	**TN**	**TN**	12
134	**.71 FP**	**TN**	■	**TN**	**TN**	**TN**	**.88 TP**	82
176	**.62 FP**	**TN**	**.92 TP**	■	**TN**	**TN**	**TN**	122
247	**TN**	**TN**	**TN**	**TN**	■	**TN**	**TN**	135
266	**TN**	**TN**	**TN**	**.6 FP**	**.82 TP**	■	**TN**	137
342	**TN**	**TN**	**.64 FP**	**TN**	**TN**	**TN**	■	176
Cysteine location	23	30	134	176	247	266	342	

Table 4. Overall performance results, shown as a collection of sub-tables, one for each protein. The results for the protein C2GnT-I using DiANNA are not reported as proteins with > 10 cysteines are not supported. A zero in the denominator of the performance metric results in it having a value of Undefined.

C2GnT-I	TP	TN	FP	FN	Accuracy	Sensitivity	Specificity	Matthew's Corr. Coeff.
DiANNA	0	46	5	4	0.84	0	0.9	-0.09
DISULFIND	> 10 cys							
PreCys	0	47	4	4	0.85	0	0.92	-0.08
MS2DB	4	45	6	0	0.89	1	0.88	0.59

Lysozyme	TP	TN	FP	FN	Accuracy	Sensitivity	Specificity	Matthew's Corr. Coeff.
DiANNA	3	31	1	1	0.94	0.75	0.97	0.72
DISULFIND	4	32	0	0	1	1	1	1
PreCys	1	32	0	3	0.92	0.25	1	0.48
MS2DB	2	23	9	2	0.69	0.5	0.72	0.15

Aldolase	TP	TN	FP	FN	Accuracy	Sensitivity	Specificity	Matthew's Corr. Coeff.
DiANNA	0	24	4	0	0.86	?	0.86	?
DISULFIND	0	28	0	0	1	?	1	?
PreCys	0	28	0	0	1	?	1	?
MS2DB	0	27	1	0	0.96	?	0.96	?

ASPA	TP	TN	FP	FN	Accuracy	Sensitivity	Specificity	Matthew's Corr. Coeff.
DiANNA	0	24	4	0	0.86	?	0.86	?
DISULFIND	0	28	0	0	1	?	1	?
PreCys	0	28	0	0	1	?	1	?
MS2DB	0	28	0	0	1	?	1	?

ST8Sia IV	TP	TN	FP	FN	Accuracy	Sensitivity	Specificity	Matthew's Corr. Coeff.
DiANNA	0	9	3	3	0.6	0	0.75	-0.25
DISULFIND	0	13	0	2	0.87	0	1	?
PreCys	0	11	2	2	0.73	0	0.85	-0.15
MS2DB	2	13	0	0	1	1	1	1

FucT VII	TP	TN	FP	FN	Accuracy	Sensitivity	Specificity	Matthew's Corr. Coeff.
DiANNA	0	9	3	3	0.6	0	0.75	-0.25
DISULFIND	0	11	1	3	0.73	0	0.92	-0.13
PreCys	3	12	0	0	1	1	1	1
MS2DB	3	12	0	0	1	1	1	1

Lactoglobulin	TP	TN	FP	FN	Accuracy	Sensitivity	Specificity	Matthew's Corr. Coeff.
DiANNA	0	17	3	1	0.81	0	0.85	-0.09
DISULFIND	0	20	0	1	0.95	0	1	?
PreCys	0	20	0	1	0.95	0	1	?
MS2DB	1	20	0	0	1	1	1	1

FT III	TP	TN	FP	FN	Accuracy	Sensitivity	Specificity	Matthew's Corr. Coeff.
DiANNA	1	17	2	1	0.86	0.5	0.89	0.33
DISULFIND	0	19	0	2	0.9	0	1	?
PreCys	0	19	1	1	0.9	0	0.95	-0.05
MS2DB	4	16	1	0	0.95	1	0.94	0.87

b1,4-GalT	TP	TN	FP	FN	Accuracy	Sensitivity	Specificity	Matthew's Corr. Coeff.
DiANNA	1	17	2	1	0.86	0.5	0.89	0.33
DISULFIND	0	19	0	2	0.9	0	1	?
PreCys	0	17	2	2	0.81	0	0.89	-0.11
MS2DB	2	15	4	0	0.81	1	0.79	0.51

5 Conclusions

In this paper we have presented a comparative analysis of disulfide bond determination using computational-predictive and mass spectrometry-based methods. The proposed mass spectrometry-based method seeks to efficiently search the combinatorial space of possible peptide fragments and find high-quality correspondences with measurements from tandem mass spectra. Subsequently, the correspondence scores (match ratios) are used to solve a maximal weight matching problem to obtain a globally optimal disulfide bond assignment. This approach contrasts significantly from the core philosophy of computational predictive methods, where the challenge lies in determining the optimal machine learning algorithm, the features to be used, and selection of the training data set. The experimental results show that in general, the direct measurement philosophy underlying mass spectrometry-based methods can outperform the model-and-predict method. At the same time, specificities of protease-dependent digestion combined with specificities of collision-based fragmentation imply that certain bonding topologies can be more reliably discerned using prediction-based methods. To the best of our knowledge, the comparative investigations presented in this paper (and the underlying questions researched) are unique at the current state-of-the-art. We believe that these results provide important cues for future development of both computational-predictive methods as well as mass spectrometry-based algorithmic techniques.

Acknowledgements

This work is supported in part by funding from the NSF grants IIS-0644418 (CAREER) and CHE-0619163, and the Center for Computing for Life Sciences at San Francisco State University. The authors thank Bruce Macher and Ten-Yang Yen for supplying the data we used and for their input they provided in discussions about the method.

References

[1] Angata, K., Yen, T.Y., El-Battari, A., Macher, B.A., Fukuda, M.: Unique disulfide bond structures found in ST8Sia IV polysialyltransferase are required for its activity. J. Biol. Chem. 18, 15369–15377 (2001)

[2] Fariselli, P., Riccobelli, P., Casadio, R.: Role of evolutionary information in predicting the disulfide-bonding state of cysteine in proteins. Proteins: Structure, Function, and Genetics 36, 340–346 (1999)

[3] Frasconi, P., Passerini, A., Vullo, A.: A Two-Stage SVM Architecture for Predicting the Disulfide Bonding State of Cysteines. In: Proc. of the IEEE Workshop on Neural Networks for Signal Processing, pp. 25–34 (2002)

[4] Martelli, P.L., Fariselli, P., Malaguti, L., et al.: Prediction of the Disulfide Bonding State of Cysteines in Proteins with Hidden Neural Networks. Protein Engineering 15, 951–953 (2002)

[5] Fariselli, P., Casadio, R.: Prediction of disulfide connectivity in proteins. Bioinformatics 17, 957–964 (2001)

[6] Vullo, A., Frasconi, P.: Disulfide connectivity prediction using recursive neural networks and evolutionary information. Bioinformatics 20, 653–659 (2004)

[7] Ferre, F., Clote, P.: DiANNA: A Web Server for Disulfide Connectivity Prediction. Nucleic Acids Research 33, 230–232 (2005)

[8] Cheng, J., Saigo, H., Baldi, P.: Large-scale prediction of disulphide bridges using kernel methods, two-dimensional recursive neural networks, and weighted graph matching. Proteins 62, 617–629 (2006)

[9] Zhao, E., et al.: Cysteine Separation Profiles on Protein Sequences Infer Disulfide Connectivity. Bioinformatics 8, 1415–1420 (2005)

[10] Chen, Y.-C., Hwang, J.-K.: Prediction of Disulfide Connectivity from Protein Sequences. Proteins 61, 507–512 (2005)

[11] Singh, R.: A Review of Algorithmic Techniques for Disulfide-Bond Determination. Briefings in Functional Genomics and Proteomics 1(1) (to appear, 2008)

[12] Fiser, A., Simon, I.: Predicting the Oxidation State of Cysteines by Multiple Sequence Alignment. Bioinformatics 16, 251–256 (2000)

[13] Muskal, S.M., Holbrook, S.R., Kim, S.-H.: Prediction of the Disulfide-bonding state of cysteine in proteins. Protein Engineering 3, 667–672 (1990)

[14] Fariselli, P., Riccobelli, P., Casadio, R.: Role of evolutionary information in predicting the disulfide-bonding state of cysteine in proteins. Proteins: Structure, Function, and Genetics 36, 340–346 (1999)

[15] Frasconi, P., Passerini, A., Vullo, A.: A Two-Stage SVM Architecture for Predicting the Disulfide Bonding State of Cysteines. In: Proc. of the IEEE Workshop on Neural Networks for Signal Processing, pp. 25–34 (2002)

[16] Martelli, P.L., Fariselli, P., Malaguti, L., et al.: Prediction of the Disulfide Bonding State of Cysteines in Proteins with Hidden Neural Networks. Protein Engineering 15, 951–953 (2002)

[17] Fariselli, P., Casadio, R.: Prediction of disulfide connectivity in proteins. Bioinformatics 17, 957–964 (2001)

[18] Vullo, A., Frasconi, P.: Disulfide connectivity prediction using recursive neural networks and evolutionary information. Bioinformatics 20, 653–659 (2004)

[19] Ceroni, A., Passerini, A., Vullo, A., et al.: DISULFIND: A Disulfide Bonding State and Cysteine Connectivity Prediction Server. Nucleic Acids Research 34, 177–181 (2006)

[20] Ferre, F., Clote, P.: DiANNA: A Web Server for Disulfide Connectivity Prediction. Nucleic Acids Research 33, 230–232 (2005)

[21] Cheng, J., Saigo, H., Baldi, P.: Large-scale prediction of disulphide bridges using kernel methods, two-dimensional recursive neural networks, and weighted graph matching. Proteins 62, 617–629 (2006)

[22] Lenffer, J., Lai, P., Mejaber, W.-E., et al.: CysView: Protein Classification Based on Cysteine Pairing Patterns. Nucleic Acids Research 32, 350–354 (2004)

[23] Lee, T., Singh, R., Yen, T.Y., Macher, B.: An Algorithmic Approach to Automated High-Throughput Identification of Disulfide Connectivity in Proteins Using Tandem Mass Spectrometry. In: 6th Annual International Conference on Computational Systems Bioinformatics (CSB 2007) (2007)

[24] Swiss-Prot database web site, `http://expasy.org/sprot/`

[25] Gabow, H.: Implementation of Algorithms for Maximum Matching on Nonbipartite Graphs. Ph.D. thesis, Stanford University (1973)

[26] Schilling, B., Row, R.H., Gibson, B.W., et al.: MS2Assign, Automated Assignment and Nomenclature of Tandem Mass Spectra of Chemically Crosslinked Peptides. Journal of American Society of Mass Spectrometry 14, 834–850 (2003)

[27] Ceroni, A., Passerini, A., Vullo, A., et al.: DISULFIND: A Disulfide Bonding State and Cysteine Connectivity Prediction Server. Nucleic Acids Research 34, 177–181 (2006)

[28] Tsai, C.H., Chen, B.J., Chan, C.H., Liu, H.L., Kao, C.Y.: Improving disulfide connectivity prediction with sequential distance between oxidized cysteines. Bioinformatics 21, 4416–4419 (2005)

Exploration of Evolutionary Relations between Protein Structures

Natalja Kurbatova and Juris Viksna

Institute of Mathematics and Computer Science
University of Latvia
Raina bulvaris 29,
LV-1459 Riga, Latvia
natalja@lu.lv

Abstract. We describe a new method for the exploration of evolutionary relations between protein structures.

The approach is based on the ESSM algorithm for detecting structural mutations, the output of which is then used for construction of fold space graphs. Fold space graphs can be regarded as a convenient tool for visualization and analysis of evolutionary relationships between protein structures, providing more information than traditional phylogenetic approaches.

We have applied the method for analysis of evolutionary relations between CATH protein domains. The experiments allowed us to obtain estimates of the distribution of probabilities for different types of fold mutations, detect several chains of evolutionary related protein domains as well as to explore the most probable β-sheet extension scenarios.

1 Introduction

In this paper a new method for the exploration of evolutionary relations between protein structures is described. The method essentially uses a combination of two separate techniques: prediction of possible structural mutations between protein domains, which is be done by the *ESSM* algorithm, and construction and analysis of fold space graphs.

The method has been applied for exploration of potential evolutionary relationships between CATH [14] protein domains and several facts about the evolutionary relationships between these domains have been established. It should be noted however that this is not the method for making the high certainty predictions about structure evolution, but rather a convenient tool that allows to detect easily potential evolutionary relations, which then should be verified individually.

Generally our approach is based on the assumption that protein structures, similarly to sequences, have evolved by a stepwise process, each step involving a small change in the protein fold. Such a model is unlikely to provide a full picture of structure evolution (a full picture of structure evolution and appropriate model is not known yet). However, there are a number of studies demonstrating that such an approach is useful in the exploration of basic tendencies in evolution of the protein structures and functions [3], [6], [13], [19].

The traditional method to study and represent evolution is visualization of the evolutionary relationship between taxonomic groups by means of phylogenetic trees. A classical phylogenetic tree is a specific type of cladogram where the branch lengths are proportional to the predicted or hypothetical evolutionary time between organisms, proteins (sequences or structures), etc. [2], [11].

Cladograms cannot be considered completely true and accurate descriptions of the evolutionary history of organisms, because they only illustrate the probability that two organisms (sequences or structures) are more closely related to each other than to a third organism. If the underling data is unsufficient for unambiguous explanation of evolution, cladograms still shows just one possible evolution scenario.

In almost all cases cladograms are the best way to represent results of studies in the field of evolution, but to explore evolutionary relations between proteins other graphical methods are sometimes more appropriate.

Here we consider fold space graphs – graphs where vertices represent proteins but edges show possibly evolutionary relationships between them, and we demonstrate that this is a convenient method for the exploration of CATH fold space.

Special types of graphs and the assumption about stepwise process of fold evolution have already been used in the studies of Matsuda *et al.* [13], when the connectivity of the fold space has been explored under the assumption that most of the fold mutations are extensions of a β-sheet with a new β-strand at its end.

The exploration of evolution in the fold space using step-by-step small fold changes have been done by Przytycka *et al.* [16], where the authors do not directly consider fold mutations, they propose 4 "folding motifs" from which a large part of all known β-folds can be constructed, and also by Viksna and Gilbert [19], where an attempt to estimate the frequencies of different types of fold mutations has been made.

Our combined method for exploration of the CATH fold space consists of two stages:

– All-against-all comparison of the CATH domains by the *ESSM* software [9]. This software is based on the SSM tool for structure comparison [7],[8] and permits the discovery of "structural mutations", which correspond to small changes in secondary structure elements (SSEs) between two protein structures;
– Construction of *fold space graphs* on the basis of discovered "structural mutations" along with methods for their visualization and automated analysis.

The basic motivation for the creation of *fold space graphs* (the second stage of our combined method) was potential interest of biologists [3], [6] in an automated method for search of non-trivial (e.g. consisting of at least three elements) chains of structures $F_1,\ldots,F_N$, such that evolutionary relations between structures F_i and F_{i+1} seem feasible, but can't be directly detected between structures F_i and F_{i+k} for $k > 1$.

Our results showed that fold space graphs really allow to find chains of CATH domains according to the put forward conditions. Extracted chains of domains from CATH class 2 (mainly β) gave us a possibility to explore the most probable β-sheet extension scenarios using terms of stepwise process of the fold evolution.

Besides, the analysis of fold space graphs allows to propose the possible evolutionary mechanisms employed in the fold space (such as possible origins inside the group of CATH domains and connections between different CATH homologous superfamilies or between subgroups of one superfamily) and gives magnificent possibility to explore domain/protein clustering in the fold space.

2 Methods

2.1 Fold Mutations

The definition of fold mutations involves β-strand (E), α-helix (H), loop, β-hairpin – two adjacent β-strands that also hold adjacent positions in a β-sheet (S_2), and 3-β-meander – three adjacent β-strands that also hold adjacent positions in a β-sheet (S_3).

The following set of fold mutations (each of them can occur in both directions) has been used in this work:

1. *Insertion (deletion):* loop $\longleftrightarrow E$,
2. *Insertion (deletion):* loop $\longleftrightarrow H$,
3. *Insertion (deletion):* loop $\longleftrightarrow S_2$,
4. *Insertion (deletion):* loop $\longleftrightarrow S_3$,
5. *Substitution:* $E \longleftrightarrow H$,
6. *Substitution:* $S_2 \longleftrightarrow E$,
7. *Substitution:* $S_2 \longleftrightarrow H$,
8. *Substitution:* $S_3 \longleftrightarrow E$,
9. *Substitution:* $S_3 \longleftrightarrow H$.

This set is largely based on the types of fold mutations proposed in [3], [6]. Additionally, it includes insertions/deletions and substitutions of β-hairpins (the existence of such changes was suggested in [19]). The set consists of possible fold mutations that could occur during protein evolution and each of them is confirmed by real biological examples [3], [6], [19]. Most of these fold mutations are presumably the result of accumulated point-mutations (insertions/deletions and substitutions of single amino acids) in the protein sequence.

There are two more types of fold mutations that are not considered here, since their prediction by the *ESSM* tool have been too unreliable [9]: β-*hairpin swaps* – exchange of the order of β-hairpin strands in a β-sheet [3] and *circular permutation of SSEs* – changes in protein connectivity that can be visualized through ligation of the termini and cleavage at another site [4], [15], [18], [21].

2.2 CATH Fold Space

Representative set CATH-95 (latest version 3.1.0) provided by the CATH database and containing proteins with less than 95% sequence similarity was used for experiments. Additionally, protein structures obtained with NMR technology and crystallized structures with resolution greater than 3 Å were removed.

The main reason for using a representative set and not CATH in the whole was to exclude protein structures that for a number of reasons are overrepresented in CATH (a more detailed discussion and motivation for using specifically CATH-95 is given in [19]). Also the number of protein domains in this study had to be limited due to computational restrictions (computational time needed to make all-against-all comparisons).

The obtained representative set for CATH class 1 (CATH1) contains 2502 domains, for class 2 (CATH2) 3314 domains and representative set class 3 (CATH3) contains 6102 domains.

2.3 *ESSM* Algorithm for Recognition of Fold Mutations

ESSM (Evolutionary Secondary Structures Matching) tool has been recently developed to compare protein structures and automatically identify different types of fold mutations [9].

This tool was created to detect a structural changes in proteins and could be used either directly for the comparison of two structures, or for searching for structural changes within a database of known protein structures.

The *ESSM* tool is based on the *SSM* (Secondary Structures Matching) algorithm for protein structure comparison [7], [8] and uses the so-called 3D graphs approach [20]. For the pair of proteins *ESSM* detects not only structural similarity but also possible fold mutations between these proteins (Figure 1).

As a result of pairwise comparison *ESSM* produces two structural similarity scores: tradicional RMSD score and ESSM score – measuring the quality of SSEs matching, as well as the alignment of SSEs and the number of fold mutations of each particular type.

A more detailed description of the *ESSM* algorithm is given in [9].

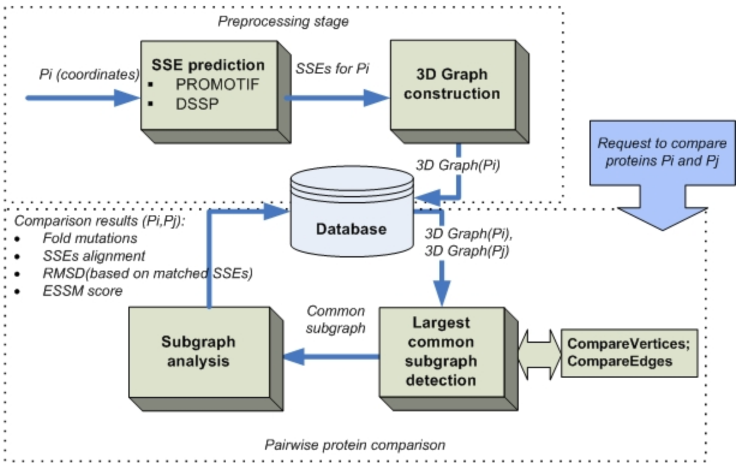

Preprocessing stage
Pi (coordinates)
SSE prediction
PROMOTIF
DSSP
SSEs for Pi
3D Graph construction
3D Graph(Pi)
Request to compare proteins Pi and Pj
Database
Comparison results (Pi,Pj):
Fold mutations
SSEs alignment
RMSD(based on matched SSEs)
ESSM score
3D Graph(Pi), 3D Graph(Pj)
Common subgraph
Subgraph analysis
Largest common subgraph detection
CompareVertices; CompareEdges
Pairwise protein comparison

Fig. 1. Schema of ESSM tool. The *ESSM* tool consists of two parts: **1. Preprocessing.** SSEs are detected using SSE prediction tools (DSSP or Promotif) or obtained from external database (like CATH) than 3D graph construction is performed for every protein/domain. **2. Comparison procedure.** Pairwise comparison based on the detection of the largest common subgraph is performed for given pair of proteins/domains. The results of *ESSM*, SSEs alignment and different scores, are stored in the database.

2.4 Construction of Fold Space Graphs

We have constructed fold space graphs specifically for CATH protein domains, but the same approach could be applied to an arbitrary set of protein structures.

In initial phase all-against-all comparisons of CATH domains using the *ESSM* program have been performed. This starts with construction of 3D graphs for these protein domains (Figure 1 – *ESSM* preprocessing stage). Three decisions have been made to minimize the number of pairwise comparisons (Figure 1 – second *ESSM* stage):

– Sets CATH1, CATH2 and CATH3 were treated separately, because few "short" evolutionary relations was to be expected between these groups;
– Only domains with more than three SSEs were considered: $N_{SSE}(Pi) \geq 4$, where N_{SSE} is the number of secondary structure elements in domain;
– Only CATH domain pairs with a difference in the number of SSEs less than or equal to four were chosen: $N_{SSE}(Pi) - N_{SSE}(Pj) \leq 4$.

After pairwise comparison by using the *ESSM* for each pair of domains the similarity between protein sequences was computed by using the **ssearch** implementation of the Smith-Waterman algorithm [17]. The sequence similarity between two domains Pi and Pj was represented by a normalized score, computed by

$$SW_{norm} = SWscore(Pi,Pj)/max\{SWscore(Pi,Pi),SWscore(Pj,Pj)\}.$$

The score can be interpreted as the percentage similarity between two domains.

For every pair of CATH domains Pi and Pj under examination *ESSM* results (number of fold mutation of each particular type, RMSD score, ESSM score) and sequence comparison result (SW_{norm} score) were obtained.

The graph construction and visualization program is based on a part of the system modeling tool GRADE [5].

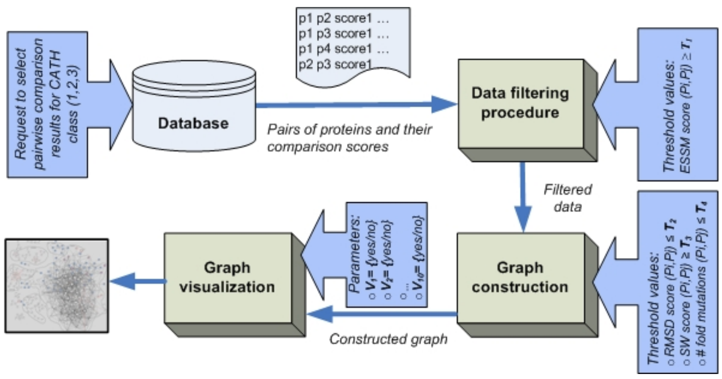
In fold space graphs vertices represent proteins/domains and edges show possibly evolutionary relations between them. PDB codes and CATH classification are used for labeling of domains in graphs.

The strength of our construction method is flexibility in the choice of criterion for evolutionary relationships between proteins, where a combination of different scores (RMSD, sequence similarity and number and types of fold mutations between proteins) is used and user choice defines the thresholds for values of the scores. This means that the construction and visualization processes could be parameterized so that different possible evolutionary relations are emphasized. The following parameters and thresholds were used in this work:

– Parameter for data filtering: ESSM score $\geq T_1$, where in the frame of this work $T_1 = 0.8$.
– Parameters for graph construction: RMSD score $\leq T_2$, where $T_2 = \{2\text{ Å}, 3\text{ Å}, 4\text{ Å}, 5\text{ Å}\}$; SW_{norm} score $\geq T_3$, where $T_3 = \{0, 10, 20, 25, 30\}$; Number of fold mutations $\leq T_4$, where $T_4 = \{1, 2, 3, 4, 5\}$.
– Parameters for graph visualization: $V_n = \{yes/no\}$, where n = 1,...,9. Parameter $V_n = yes$ means that edge (Pi, Pj) will be marked out on the graph if there is fold

mutation of type n (according to the fold mutation number in section 2.1) between Pi and Pj; $V_{10} = \{yes/no\}$, where "yes" means that vertices Pi and Pj will be marked out if labels of CATH classification for Pi and Pj differ, but there is an edge (Pi, Pj) on the graph.

The schema for graph construction is given in Figure 2.



Request to select pairwise comparison results for CATH class (1,2,3)
Database
p1 p2 score1 ...
p1 p3 score1 ...
p1 p4 score1 ...
p2 p3 score1 ...
Pairs of proteins and their comparison scores
Data filtering procedure
Threshold values: ESSM score (Pi,Pj) ≥ T1
Filtered data
Graph construction
Threshold values: RMSD score (Pi,Pj) ≤ T2; SW score (Pi,Pj) ≥ T3; # fold mutations (Pi,Pj) ≤ T4
Constructed graph
Parameters: V1= {yes/no}; V2= {yes/no}; ...; V10= {yes/no}
Graph visualization

Fig. 2. Schema of fold space graph construction. There are two main aspects in the graph creation procedure: data filtering and selection of thresholds for the construction and visualization procedures.

As input data for program file with PDB codes and CATH classification of pairs of domains under examination and their structure and sequence comparison results was used, as well as list of values for thresholds (T_1, T_2, T_3, T_4) and list of values for visualization parameters (V_n, where n=1,...,10).

In the resulting graphs two vertices i and j are connected if scores for corresponding domains (Pi, Pj) are in admissible limits. In such a way graphs represents the protein fold space (respectively for CATH classes 1, 2 and 3), in which possibly evolutionary related proteins are connected.

3 Results

3.1 Distribution of Probabilities

Comparative probabilities of different types of fold mutations were computed for each CATH class (1, 2 and 3) in order to compare results of the *ESSM* with results obtained in previous studies [19] and to check the correspondence between probabilities of fold mutations of particular type and structural features of CATH classes.

For the computation of probabilities for fold mutation types formulas presented in [19] were used: value *m(X,d)/n(d)* gives the probability distributions for type X mutation, where *m(X,d)* is the number of observed mutations of type X between pairs of proteins that have RMSD score less than 3 Å and sequence similarity d, and *n(d)* denotes the total number of protein pairs in a test set with sequence similarity d.

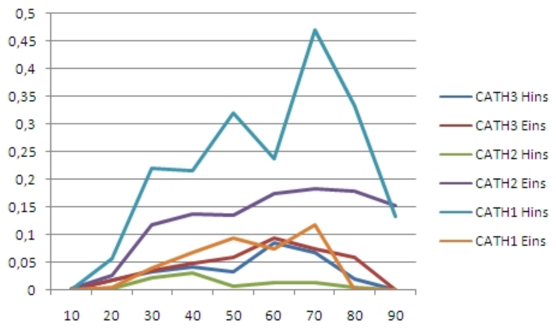


Fig. 3. Distribution of probabilities for insertion/deletion of β-strands (Eins) and α-helices (Hins). The values on the x axis represent normalized scores of sequence similarities. The values on the y axis show computed probabilities.

Figure 3 represents the probability distributions for the most probable fold mutation types [19]: insertion/deletion of single β-strands and α-helices.

For CATH1 the overall distribution of probabilities, if only insertion/deletion of β-strands (Eins) and α-helices (Hins) are considered, is 17% and 83% respectively. For CATH2 this distribution looks different: 92% for β-strands insertion/deletion and only 8% for α-helices indels. Finally for CATH3 distribution is almost bisected: 56% for β-strands and 44% for α-helices insertion/deletion.

By comparison with previous studies of the distribution of probabilities for fold mutation types [19], we were able to estimate the insertion/deletion of α-helices and our results seem to be realistic taking into account the CATH division into classes.

3.2 Evolutionary Relationships between CATH Domains

In this section three examples of fold space graphs (one example for each CATH class) are considered (Figure 4).

Domain clustering. The first example (Figure 4 part I) shows the partitioning of CATH class 2 domains into clusters that could be found in the fold space graphs.

A number of fold space graphs were created for CATH2 using $T_3 = 0$ (sequence similarity was out of consideration) and different thresholds T_2 (RMSD score) and T_4 (number of fold mutations). Clusters that were found in the fold space graph mainly correspond to the CATH classification by homologous superfamilies and in all cases correspond to the CATH topologies. In some cases superfamilies are partitioned into several clusters - each cluster consists of domains with particular number and types of β-sheets (β-hairpin, β-meander, n-stranded β-sheet). Such subclusters disappear with the increase of thresholds T_2 and T_4.

The overall number of superfamilies in the dataset of CATH2 that appear in our fold space graph is 51 (after the data filtering procedure). The number of obtained clusters is 73 where 27 of them were considered as non-trivial (consisting of more than 2 domains) clusters.

The size of obtained clusters varies significantly due to different populations in CATH2 homologous superfamilies. For CATH1 and CATH3 the clustering is also corelated with CATH superfamilies, but often there tend to be several clusters within a single superfamily.

Although at first look this may appear to be hardly surprising, we think that this observed relation between clusters and superfamilies is quite non-trivial fact. Up to some extent this shows that are biological reasons for structure division into superfamilies and they are not just the notion that has been invented for classification convenience.

The second and third examples (Figure 4 parts II and III) demonstrate how exploration of the fold space graphs might allow to propose the possible evolutionary mechanisms employed in the fold space.

Detection of possible evolutinary mechanisms. The second example (Figure 4 part II) represent the part of the fold space graph for CATH class 3 concerning CATH homologous superfamily 3.30.500.10 ”Murine Class I Major Histocompatibility Complex, H2-DB, subunit A, domain 1“. All domains of this superfamily are connected to three particular domains: 1zt1A01, 1kjvA01 and 1k5nA01, where each pair (any domain from the superfamily and one from the three listed domains) could be evolutionary related through two fold mutations: E insertion/deletion and H insertion/deletion.

These results might reflect the evolutionary mechanisms employed in fold space of the 3.30.500.10 superfamily, where structural domains might have evolved from three particular domains.

Detection of non-trivial relationships between CATH domains. The last example (Figure 4 part III) demonstrates how the usage of fold space graphs could help to find non-trivial relationships between CATH domains (connections between different CATH homologous superfamilies). Connections between different CATH homologous superfamilies were highlighted for CATH1 using features of our graph construction and visualization program which allow accentuation of specific vertices of the graph ($V_{10} = yes$). Figure 4 part III shows how domain 1ycsB01 from the homologous superfamily 1.25.40.20 “Cell Cycle,Transcription” is used as a connector between this superfamily and another one - 1.10.220.10 “Protein And Metal Binding Protein”.

Structures of domains from the superfamily 1.10.220.10 which are connected with 1ycsB01 are practically identical to half of this domain structure. At the same time there are very few sequence similarities between them.

In the fold space graph for CATH2 we found that domains in the superfamily 2.60.40.30 “Fibronectin type III” are in many cases connected to immunoglobulin constant domains from the superfamily 2.60.40.10 [10]. Some domains from the superfamily 2.60.40.760 “Allergens” are also connected to immunoglobulin fold [12].

It should be noted however, that generally the observed connections between domains of different superfamilies are somewhat less credible than connections within the same superfamily – they are the one of the first that dissappear with the increase of the threshold T_3 for sequence similarity. Thus, they generally should be used only for guidance, and the observed connections require some additional biological verification.

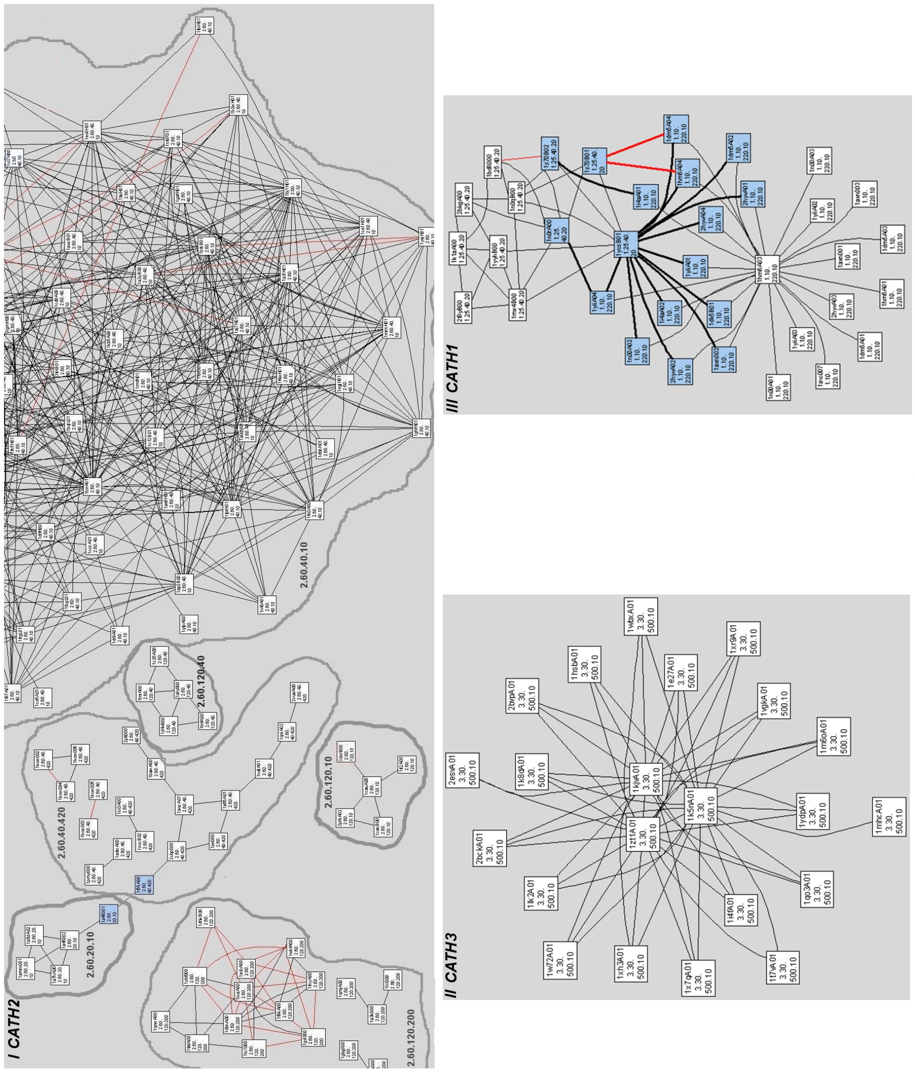


Fig. 4. Fold space graphs for CATH. In all three parts of the figure edges colored with red show β-hairpin insertion/deletion. Vertices colored with blue highlight change-overs between homologous superfamilies. **I** Part of the CATH2 fold space graph showing the partitioning into clusters. Thresholds: $T_2 \leq 4$ Å $T_4 \leq 3$ and $T_3 = 0$. CATH homologous superfamily 2.60.40.10 defines the largest cluster which contains all domains from this superfamily. At the same time domains from the superfamily 2.60.120.200 are partitioned into several subclusters. **II** Part of the CATH3 fold space graph for homologous superfamily 3.30.500.10. Thresholds: $T_2 \leq 3$ Å $T_4 \leq 5$ and $T_3 = 20$. **III** Part of the CATH1 fold space graph for homologous superfamilies 1.25.40.20 and 1.10.220.10. Thresholds: $T_2 \leq 3$ Å $T_4 \leq 5$ and $T_3 = 0$.

3.3 Exploration of β-Sheets

The most challenging usage of fold space graphs is the extraction of evolutionary pathways where every two proteins in a neighborhood are evolutionary related.

Evolutionary pathway might be defined as a chain of structures $F_1,\ldots,F_N$ when $N > 2$, such that an evolutionary relationship between structures F_i and F_{i+1} is feasible. The pathway also defines fold mutations between proteins F_1 and F_N.

CATH class 2 (mainly - beta) has been chosen as a fold space for the exploration of β-sheet extension scenarios, since this CATH class contains domains with rich β-sheet structures.

Different possible evolutionary pathways were found in the largest homologous superfamily of CATH2 – 2.60.40.10 "Immunoglobulins". However, observed fold mutations mainly belong to two types:

- insertion/deletion of β-strand at the C-termini of domains (Figure 5 a $\leftrightarrow$ b: β-strand a; Figure 5 b $\leftrightarrow$ c: β-strands a and a');
- insertion/deletion of β-hairpins at the places of cross-overs between two main β-sheets (Figure 5 a $\leftrightarrow$ b: β-hairpin c' c''; Figure 5 e $\leftrightarrow$ f: β-hairpin f' f'').

The following statistics are obtained for the Immunoglobulins: in 27% of comparison pairs one domain consists of 4-stranded and 7-stranded β-sheets (Figure 5d) and in 50% of pairs there are 4-stranded and 6-stranded β-sheets (Figure 5c).

These results might reflect the process of β-sheet evolution when 2-stranded β-sheet becomes a part of a larger β-sheet (Figure 5 e $\leftrightarrow$ d and f $\leftrightarrow$ e: 2-stranded β-sheets a' b' and a h), at the same time two β-strands are united into the single one (Figure 5 e $\leftrightarrow$ d and f $\leftrightarrow$ e: β-strands b and b').

In most of the cases domains from the Agglutinin homologous superfamily 2.60.120.200 (95% of explored domains) consists of two large (at least 7-stranded) β-sheets (Figure 6b and Figure 6c) and one or two β-hairpins (Figure 6b: β-hairpin c b; Figure 6c: β-hairpins c b and c' c'').

The pair of domains (Figure 6 a $\leftrightarrow$ b) demonstrates the possible formation/destruction scenario of two 7-stranded β-sheets:

- β-meander is extended in N-termini with two β-strands and in C- termini with one β-strand (Figure 6b: k', k''' and a'). These changes together with a small β-strand insertion (Figure 6b: e') on the place of loop between two main β-sheets lead to the formation of a 7-stranded β-sheet (Figure 6b: e' a' k''' e j k k').
- The insertion of β-strand between the 2- and 4-stranded β-sheets (Figure 6b: k'') leads to the formation of 7-stranded β-sheet (Figure 6b: a d k'' f g h i).

The insertion/deletion of β-hairpin is frequently observed fold mutation in the Agglutinin superfamily (Figure 6 b $\leftrightarrow$ c: β-hairpin c' c'').

Scenarios of fold changes concerning extension of β-sheets in others homologous superfamilies of CATH2 are similar to the already described ones for superfamily 2.60.40.10.

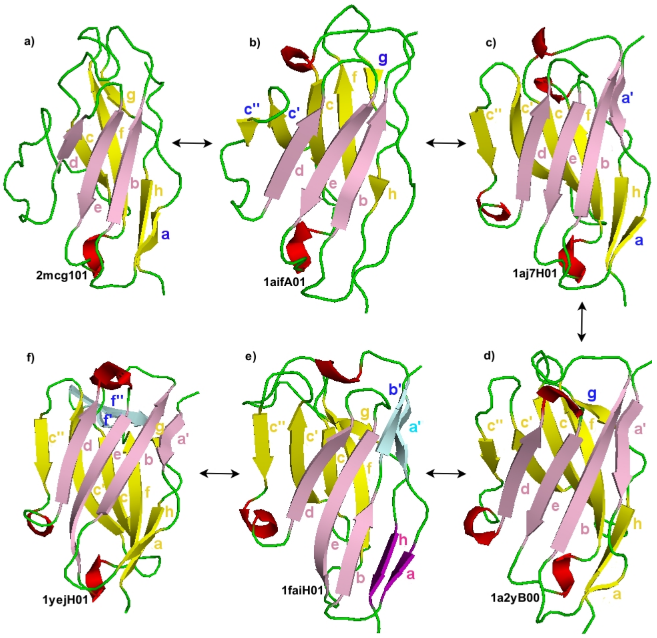


Fig. 5. Representative chain of changes in the CATH2 Immunoglobulin homologous superfamily (2.60.40.10). **a** ↔ **b**: insertion/deletion of β-strand and β-hairpin; **b** ↔ **c** and **c** ↔ **d**: insertion/deletion of β-strands; **e** ↔ **d** and **f** ↔ **e**: two β-strands fusion and formation of 4-stranded β-sheet, 5- and 2-stranded β-sheets unification; **f** ↔ **e**: insertion/deletion of β-hairpin. The most frequently observed folds are **c)** (4-stranded and 6-stranded β-sheets) and **d)** (4-stranded and 7-stranded β-sheets). Ribbon-style representations of domains generated by Pymol [1].

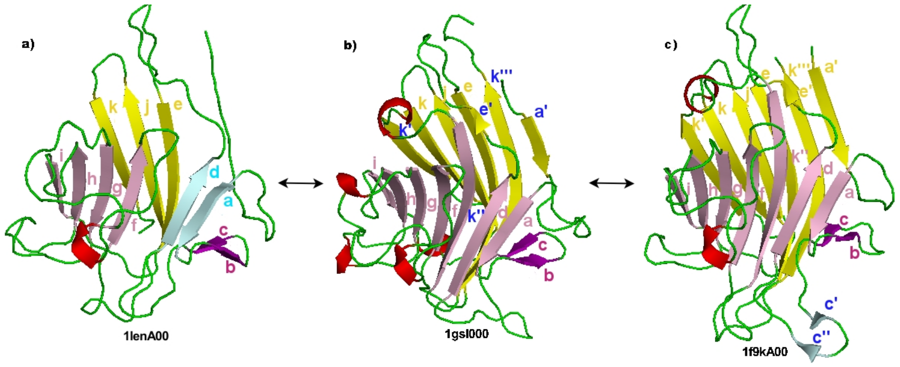


Fig. 6. Representative chain of changes in CATH2 Agglutinin superfamily (2.60.120.200). **a** ↔ **b**: β-meander extension, 2- and 4-stranded β-sheets unification; **b** ↔ **c**: β-hairpin insertion/deletion. Ribbon-style representations of domains generated by Pymol [1].

4 Summary and Conclusions

We have described a combined method based on detection of structural changes by the *ESSM* tool and subsequent construction of fold space graphs for visualization and analysis of evolutionary relationships between protein structures.

The method has been applied for the exploration of CATH fold space separately for classes 1, 2 and 3. Our results showed that such an approach is a convenient way to explore evolutionary relations between protein domains and it could be used either for detection of ”interesting” evolutionary relationships between the structures (although most of the examples that we have identified turned out already been studied and recognized as ”interesting” by biologists, it should be noted that we have detected these examples automatically), or for formulation of more general hypotheses about evolution of protein folds.

The analysis of CATH protein domains that we have performed allowed us to obtain better estimates of the distribution of probabilities for different types of fold mutations, to detect several chains of evolutionary related protein domains, as well as to explore the most probable scenarios of extension of β-sheets.

Acknowledgements

This work has been supported by the European Social Fund (ESF).

References

1. DeLano, W.: The PyMOL Molecular Graphics System. DeLano Scientific Palo Alto, CA, USA (2002), http://www.pymol.org
2. Felsenstein, J.: Inferring phylogenies from protein sequences by parsimony, distance, and likelihood methods. Methods Enzymol 266, 419–426 (2001)
3. Grishin, N.: Fold change in evolution of protein structures. Journal of Structural Biology 134, 167–185 (2001)
4. Jung, J., Lee, B.: Circularly permuted proteins in the protein structure database. Protein Science 10, 1881–1886 (2001)
5. Kikusts, P., Rucevskis, P.: Layout Algorithms of Graph-Like Diagrams for GRADE Windows Graphic Editors. In: Brandenburg, F.J. (ed.) GD 1995. LNCS, vol. 1027, pp. 361–364. Springer, Heidelberg (1996)
6. Kinch, L., Grishin, N.: Evolution of protein structures and functions. Current Opinion in Structural Biology 12, 400–408 (2002)
7. Krissinel, E., Henrick, K.: Secondary-structure matching (SSM), a new tool for fast protein structure alignment in three dimensions. Acta Crystallographica D60, 2256–2268 (2004)
8. Krissinel, E., Henrick, K.: Common subgraph isomorphism detection by backtracking search. Software – practice and experience 34, 591–607 (2004)
9. Kurbatova, N., Mancinska, L., Viksna, J.: Protein structure comparison based on fold evolution. In: Lecture Notes in Informatics - GCB 2007 Proceedings, pp. 78–89 (2007)
10. Leahy, D., Hendrickson, W., Aukhil, I., Erickson, H.: Structure of a Fibronectin Type Ill Domain from Tenascin Phased by MAD Analysis of the Selenomethionyl Protein. Science 258, 987–991 (1992)

11. Maddison, W., Maddison, D.: Interactive analysis of phylogeny and character evolution using computer program MacClade. Folia Primatol (Basel) 53(1-4), 190–202 (1989)
12. Marino, S., Castiglione Morelli, M., Fraternali, F., Tamborini, E., Musco, G., Vrtala, S., Dolecek, C., Arosio, P., Valenta, R., Pastore, A.: An immunoglobulin-like fold in a major plant allergen: the solution structure of Phl p 2 from timothy grass pollen. Structure 7-8, 943–952 (1999)
13. Matsuda, K., Nashioka, T., Kinoshita, K., Kawabata, T., Go, N.: Finding evolutionary relations beyond superfamilies: fold-based superfamilies. Protein Science 12, 2239–2251 (2003)
14. Orengo, C., Michie, A., Jones, D., Swindelis, M., Thornton, J.: CATH – a hierarchic classification of protein domain structures. Structure 5, 1093–1108 (1997)
15. Peisajovic, S., Rockah, L., Tawfik, D.: Evolution of new protein topologies through multistep gene rearrangements. Nature Genetics 32, 168–174 (2006)
16. Przytycka, T., Srinivasan, S., Rose, G.: Recursive domains in proteins. Protein Science 11, 409–417 (2002)
17. Smith, T., Waterman, M.: Identification of common molecular subsequences. Journal of Molecular Biology 147, 195–197 (1981)
18. Uliel, S., Fliess, A., Amir, A., Unger, R.: A simple algorithm for detecting circular permutations in proteins. Bioinformatics 15, 930–936 (1999)
19. Viksna, J., Gilbert, D.: Assessment of the probabilities for evolutionary structural changes in protein folds. Bioinformatics 23, 832–841 (2007)
20. Wang, X., Shapiro, B., Rigoutsos, I., Zhang, K.: Finding patterns in three-dimensional graphs: algorithms and application to scientific data mining. IEEE Transactions on Knowledge and Data Engineering 14, 731–749 (2002)
21. Weiner, J., Thomas, G., Bornberg-Bauer, E.: Rapid motif-based prediction of circular permutations in multi-domain proteins. Bioinformatics 21, 932–937 (2005)

Two Local Search Methods for Protein Folding Simulation in the HP and the MJ Lattice Models*

L. Kapsokalivas[1], X. Gan[1], A. Albrecht[2], and K. Steinhöfel[1]

[1] King's College London
Department of Computer Science
London WC2R 2LS, UK
[2] University of Hertfordshire
Science & Technology Research Institute
Hatfield AL10 9AB, UK

Abstract. We present experimental results on benchmark problems for two local search procedures that utilise the pull-move set: (i) simulated annealing with logarithmic cooling schedule and (ii) guided local search that traverses the energy landscape with greedy steps towards (potential) local minima followed by upwards steps to a certain level of the objective function. The latter method returns optimum values on established 2D and 3D HP benchmark problems faster than logarithmic simulated annealing (LSA), however, it performs worse on five benchmarks designed for the Miyazawa-Jernigan energy function, where LSA reaches optimum solutions on all five benchmarks. Moreover, the number of function evaluations executed by LSA is significantly smaller than the corresponding number for Monte Carlo simulations with kink-jump moves.

1 Introduction

Proteins regulate almost all of the cellular functions in an organism. Their functionality is determined by the 3D structure, also referred to as tertiary structure [1]. According to Anfinsen's thermodynamic hypothesis, proteins fold into states of minimum energy and, moreover, the tertiary structure can be predicted from the linear sequence of amino acids [2]. The contradiction, though, between the actual, very short folding time and the huge number of possible conformations that have to be searched in order to find the lowest energy structure, motivated C. Levinthal [3] to suggest that the native protein state might have a higher energy value than the theoretical free-energy minimum, if the latter is not kinetically accessible. This is referred to as the Levinthal's paradox and implies that the protein might be able to avoid kinetic traps of local minima or quickly escape them and therefore to find a greedy path to the native protein state [4].

Protein folding simulation has attracted many researchers from the combinatorial optimization community [5]. In a simplified version, protein folding simulation has been shown to be NP-complete [6,7] and, consequently, a variety of

* Research partially supported by EPSRC Grant No. EP/D062012/1.

search-based methods has been proposed to tackle the problem. In this context it is interesting to note that, to the best of our knowledge, little progress has been made in incorporating research on chaperone-mediated protein folding [8,9] into simulation methods. From a computational point of view, chaperonin complexes seem to act like an *oracle* in complexity theory [10], where the information provided by the oracle can be used to speed-up computations or to analyse the inherent complexity of a problem subject to the oracle complexity. The utilisation of mechanisms that determine the functionality of such *chaperonin folding machines* [8] in local search procedures, for example in associated neighbourhood relations, will certainly open up new avenues for more efficient protein folding simulations.

In the present paper, we employ a stochastic local search method that is based on the genetic local search algorithm devised in [11], and we further extend the approach presented in [14] to a new, more realistic objective function. The protein folding simulations are executed in 2D and 3D lattice structures with two types of the objective function, namely the HP-model function [15] and the Miyazawa-Jernigan energy function [16]. The neighbourhood relation is determined by the pull-move set [17,18,19]. The three components, i.e. the set of conformations in lattice structures, the objective functions and the neighbourhood relation, define an energy landscape. As described in [11], we employ information about approximations of the maximum value Γ of the minimum escape height from local minima in a landscape induced by a particular protein sequence. In our approach, the search procedure traverses the energy landscape with greedy steps towards (potential) local minima followed by escapes up to the level of $Z + \Gamma$, where Z denotes the value of the objective function of the preceding local minimum. The procedure starts with a population of randomly selected initial conformations, and after a certain number of steps a part of the best solutions found so far plus some additional conformations are selected for the re-start with a new population. Due to the random neighbourhood steps, it then creates a diversity of conformations.

2 HP and MJ Energy Functions

2.1 The HP Model

In the HP model proposed by Dill et al. [15], a protein is a connected chain of beads which can either be hydrophobic (H) or polar (P). The whole chain is embedded into either a rectangular 2D grid or a 3D cubic lattice. The potential free energy of an HP sequence is computed by a pairwise energy function, which gives value -1 to all non-covalent H-H bonds and 0 to the rest, namely H-P and P-P. Hence, minimizing the free energy of a conformation is equivalent to maximizing non-covalent HH bonds, resulting in native conformations of compact shape with a hydrophobic core surrounded by P acids. A major shortcoming of the HP model is the high degeneracy, meaning that one sequence can have many different native structures. In other words, it is difficult to prevent a sequence from folding into a different structure than the one it was designed to fold to.

There are HP sequences of unique optimal foldings, but this is not sufficient to classify them as protein-like sequences. Another problem inherent to the HP model is the limited range of different possible energy values, which yields large plateau regions in the energy landscape. Due to these facts, the performance of local search algorithms is usually degraded as a consequence of trapping them into a random walk on these plateaus. Finally, two-letter alphabets, such as H and P, exhibit several weaknesses that are highlighted in a variety of papers. In [22], for example, the author provides an explanation, from a thermodynamic point of view, why the two-letter alphabet is a poor choice for structure design. In [23], a statistical method is invoked to show that adopting a two-letter alphabet discards useful structural information. Finally, Li et al. demonstrate in [24] that the Miyazawa-Jernigan energy function [16] can improve the designability of sequences and reduce their degeneracy, thus justifying our choice to study a more realistic energy function.

2.2 The Miyazawa-Jernigan Energy Function

The MJ energy function is a pairwise interaction matrix, extracted from the distribution of contacts as they occur in real proteins, and for which the structure has been resolved into PDB format. In the original paper [16], where this energy function is introduced, there are two different interaction matrices. The first matrix, often referred to as MJa in the literature, stands for the actual energy value of each bond, while the second matrix, MJb, stands for the pairwise contributions to the total free energy related to the fact that two amino acids are forced to expel a solvent molecule and form a contact. In our study, we use the second matrix. This choice is motivated by the fact that benchmarks with provably optimal energy value in the cubic lattice exist for this matrix, thus helping to better understand the potential of our local search methods. For a comparative study of the two MJ matrices in the context of the induced energy landscapes we refer the reader to [25].

The total free energy under the MJ pairwise interactions is given by the equation 1, where $\{r_i\}$ is the set of bead coordinates that define a conformation, $\{s_i\}$ represents an amino acid sequence and e_{ij} are the entries of MJb matrix. The contact function $D(r_i - r_j)$ is 1 if beads i and j form a contact (that is not a covalent linkage) and is 0 otherwise.

$$E(\{r_i\}, \{s_i\}) = \sum_{i>j}^{N} e_{ij} D(r_i - r_j). \tag{1}$$

Regarding the MJ function, many methods are based on Monte Carlo sampling algorithms that aim at examining the validity of protein folding theories, like the properties of the sequence landscapes [25]. The MJ matrix has also been employed in protein threading and fold recognition [26]. On the other hand, a variety of structure prediction algorithms has been tested in the HP model, including a number of stochastic local search and evolutionary algorithms.

3 Related Algorithmic Methods

The first genetic algorithm applied to protein folding simulation, presented in [20], used pivot moves for the mutation step and a single point crossover operator. Several extensions followed that mainly focused on different recombination strategies, such as the multipoint crossover [27] or optimized strategies of selecting individuals for crossover [28]. Some genetic algorithms coped with the problem of non-self-avoiding conformations resulting from recombination, either with backtracking techniques [29] or by introducing penalty terms in the objective function [30]. There are also evolutionary techniques employing methods for a constraint search of the conformational space as in [31] and the evolutionary Monte Carlo approach in [32], where secondary structure constraints were introduced. Finally, an evolutionary approach hybridized with tabu search was presented in [33].

Among the stochastic algorithms for HP model, there are some examples with better performance than genetic algorithms. This includes PERM [34], SISPER [35], the ant colony optimization algorithm ACO-HPPFP-3 [36], the replica exchange Monte Carlo method [37] and GTabu search [17], where pull-moves were first introduced. For a more comprehensive discussion of local search methods, we refer the reader to [38].

4 Population-Based Local Search

Our local search algorithm resembles the genetic local search algorithm described in [11]. It starts from a random initial population of protein conformations and creates future generations of off-springs by repeatedly applying mutation and elitism. The mutation step consists of a quasi-determinist local search with continuous improvements of the objective function for all individuals in the population, meaning that downward steps may have a random component, i.e. the neighbours with improved values of the objective function might be chosen randomly. The choice of upward steps, until escaping from a local minimum, is random as well. The elitism is applied after a certain number of mutation steps, where the whole process is restarted with the best individual seen in the previous step. We should emphasize that no recombination operator is employed, since no suitable encoding of protein conformations was found that allows exchanging any part between two conformations and results in a conformation at least close to a self-avoiding conformation. Usually, only a few pairs at each generation can produce a feasible conformation after recombination and, in fact, checking the feasibility of a conformation and then recombining off-springs until a feasible conformation is achieved has a significant (negative) impact on the performance. Alternatively, we examined the possibility of fixing infeasible conformations, but this would require a number of pull-moves that is difficult to predict and would consequently spoil building blocks of good solutions, whereas recombination aims at preserving them in the population.

Apart from this specific problem with crossover, additional negative examples can be found in the literature: in [12], the authors examine the performance of a genetic algorithm with and without crossover for solving the Travelling Salesman Problem. Similar to our case, the feasibility of solutions obtained by crossover is not trivial. The authors observed that without crossover the performance was better and although they cannot argue on the speed of convergence, they provide a proof of convergence for their non-crossover genetic algorithm. In another example [13], the efficiency of crossover is examined for the Subset Sum Problem. The authors stress the fact that candidate solutions can have similar fitness and yet be quite different, which is true for the case of our protein conformations as well. In this setting, they show that crossover can lead to a sharp divergence if the mutation rate approaches a critical value.

We are now going to present the details of the proposed local search algorithm. The algorithm requires the specification of the following parameters:

1. The population size: $popSize$.
2. The number of mutation steps before the whole local search is restarted: $KSteps$.
3. The number of solutions selected to be put in the restart population: $MBestSolutions$.
4. The maximum escape from any local minimum: Γ.
5. The number of trials performed, including transitions to conformations of the same energy, before reporting a potential local minimum: $Ltrials$.

Algorithm 1. The population based local search

```
1: Starting from straight line conformations, initialize a population
   of popSize structures, by performing 5000 random pull-moves on each individual, $S_i$
2: for all $S_i$ in the population do
3:    Energy[i] = CalcFitness($S_i$)
4:    Mode[i] = Downwards
5:    Neighbors[i] = NULL
6:    plateauTransitions[i] = 0
7: end for
8: repeat
9:    RecordBestConformations($MBestSolutions$)
10:   SelectMutations($Ltrials$) /*For each individual randomly choose a neighbor*/
11:   ExecuteMutations($Gamma$) /*Accept the neighbor according to being in either
      downwards or upwards steps mode*/
12:   if $KSteps$ are completed since last restart then
13:      ElitismRestart(MBestSolutions) /*Restart population with $\frac{popSize}{MBestSolutions}$
         copies of the individuals gathered by RecordBestConformations*/
14:      Discard the best individuals gathered in previous $KSteps$.
15:   end if
16: until Minimum Energy Conformation is found.
```

We further specify the sub-procedures SelectMutations and ExecuteMutations. The treatment of plateaus and variants of RecordBestConformations with respect to performance is also discussed.

The procedure RecordBestConformations is responsible for storing the best overall solution as well as for gathering solutions for the re-start with a new population, including the best overall conformation found so far. Re-starting with only the best conformation proved to provide the best performance (in the HP model) among the alternatives tested for gathering solutions. Alternative strategies include accumulating the best individuals every *Xsteps*, so that half of the population is re-started with the best individual and the other half with individuals already accumulated. Also, instead of picking the best individual every *Xsteps*, we tried to accumulate *popSize*/2 local minima that were found during the previous *KSteps*, so as to re-start the population with half of the individuals being copies of the best overall conformation and the other half being the local minima we have accumulated.

Algorithm 2. PerfomMutations($Ltrials$)

```
1: for all $S_i$ in the population do
2:   if there was a transition from another conformation to $S_i$ then
3:     if plateauTransitions[i]+VisitedNeighbors($S_i$) $\geq$ Ltrials then
4:       Change Mode /*Downwards $\rightarrow$ Upwards , Upwards $\rightarrow$ Downwards*/
5:       if Mode was Downwards then
6:         Report Si as a local minimum and free all its neighbors so that they can
           be revisited.
7:       end if
8:     end if
9:     Compute the neighborhood, Neighbors[$S_i$] and mark all of them as free.
10:  else
11:    if All neighbors are visited or VisitedNeighbors($S_i$) == LTrials then
12:      Report local minimum/maximum
13:      Change Mode and mark current Neighbors[$S_i$] as free to be revisited.
14:    else
15:      Pick randomly a neighbor $X_i$ from Neighbors[$S_i$].
16:    end if
17:  end if
18: end for
```

In order to deal with the problem of plateaus, we only accept plateau transitions up to some extent before we switch the direction of steps. When hitting a plateau for the first time, the algorithm includes any successive transitions to conformations of the same energy as being on a potential local minimum/maximum. Then the mode switches when the plateau transitions added to the number of discarded transitions exceed *LTrials* (see line 3 of SelectMutations). In case, though, the sequence of accepting conformations with the same energy is interrupted by a rejection of a conformation, then the plateau transitions are neglected (see line 11 of SelectMutations) for the decision of switching mode, unless the algorithm performs another plateau transition. The assumption we make here is that a future plateau transition is likely to return to the same

Algorithm 3. ExecuteMutations($Ltrials$)

```
1: for all $S_i$ in the population do
2:   if Mode[i] == DOWN then
3:     if CalcFitness($X_i$) < Energy[$S_i$] then
4:       Accept the transition to $X_i$ and update Energy[$S_i$] = CalcFitness($X_i$).
5:       Reset plateauTransitions[i] to zero.
6:     else if CalcFitness($X_i$) == Energy[$S_i$] then
7:       Accept the plateau transition to $X_i$.
8:       plateauTransitions[i]++
9:     else
10:      Discard transition to $X_i$.
11:    end if
12:  end if
13:  if Mode[i] == UP then
14:    if CalcFitness($X_i$) > Energy[$S_i$] then
15:      Accept the transition to $X_i$ and update Energy[$S_i$] = CalcFitness($X_i$).
16:      Reset plateauTransitions[i] to zero.
17:      if localMinEnergy[i]+Gamma is reached then
18:        Change mode to Downwards.
19:      end if
20:    else if CalcFitness($X_i$) == Energy[$S_i$] then
21:      Accept the transition to $X_i$.
22:      plateauTransitions[i]++
23:    else
24:      Discard transition to $X_i$.
25:    end if
26:  end if
27: end for
```

plateau; if it does not then the particular conformation was rather isolated from the current plateau and thus we should not base our decision on current values of plateau transitions but on the neighbourhood of this particular conformation. The size of the plateau is unknown anyway, so when reaching a boundary of the current plateau, we decide that this is an actual boundary if we go back to the same plateau.

For the estimation of Γ we a employ the simulated annealing algorithm with logarithmic cooling schedule, where the maximum increase of the objective function is monitored in-between two successive improvements of the best value obtained so far. This approach usually overestimates Γ significantly. Therefore, we are searching for a suitable constant c such that $D = G_{\text{monit}}/c$ comes close to Γ, where G_{monit} is the maximum of the monitored increases of the objective function in-between two successive total improvements of the objective function. Using the same procedure we have previously conjectured that $\Gamma \leq n^{1-\frac{1}{d}}/2$ for HP sequences of length n and dimensionality $d = 2, 3$ [14], and now we make a suitable choice of D and c for the MJ sequences as well.

5 Results on Benchmark Problems

By using the population-based guided random walk, we were able to obtain optimum conformations for the popular HP benchmarks in 2D and 3D [17,19,20], improving on the runtime required by the simulated annealing (SA) algorithm with logarithmic cooling schedule [14], especially in the 3D case. Interestingly, we obtain a different picture for the five protein sequences provided in [21] with the Miyazawa-Jernigan (MJ) energy function. Our SA algorithm with logarithmic cooling schedule introduced in [14] returned optimum solutions for all five MJ sequences with significantly reduced transition numbers compared to [21]. However, for approximately the same time effort, the population-based local search returned optimum solution only on two of the MJ sequences within five trials, with an average of 13% above the optimum value on the remaining three benchmarks.

5.1 The Benchmarks

The sequences used for the HP model are the 2D sequences $S36_{-14}$, $S48_{-23}$, $S60_{-36}$ and $S64_{-42}$ taken from [20] and the $S85_{-53}$ and $S100_{-48}$ from [17]. In the 3D case, we test the 10 sequences of 48 amino acids from [19]. The MJ sequences are taken from [21], where the authors used a well-known procedure devised by Shankovich [39] for the design fast folding sequences with respect to their landscape properties. The sequences were designed to adopt two distinct topologies. For brevity, we provide a table for the MJ sequences only, since the HP benchmarks are well established.

Table 1. MJ sequences

No.	Sequence	E_{native}
MJ1	FRTRPLNHDFYNYKIWEPFKPADFPKAWDRMLDHVWDSMASWGHQHCS	-25.85
MJ2	CDLPPFTYRHHGNDFWKNYEMIKHWDLWRDMFRAFWSDPVKASPHQAS	-25.92
MJ3	FRTPWVSHQFYAYKLMEHFKWGDFCRNMDKWIDSLPDRWNPAPHDHAS	-26.09
MJ4	KDKIHFRMNYGYPAWDAQSVKDLTCPRDWHFPHMRDPSHNWELAFFWS	-25.87
MJ5	ENDVTMDMDPSPCLFRIHNLPRAHSFDRFGWHQFDKYHYKWKWAWAPS	-26.15

5.2 HP Model Simulations

The simulations were performed for the method described in Section 4 only, since the results for our LSA algorithms are published already in [38] (2D case) and [14] (3D case).

On all HP benchmark problems, the population-based local search from Section 4 has been executed until a conformation with minimum energy was found. The results are presented in terms of transitions between conformations, but the actual number of function evaluations is the product of number of generations (fourth column in Table 2) and population size, which is set to 20 for all 2D and

3D benchmarks. The fifth column in Table 2 indicates the number of feasible neighbourhood transitions.

Table 3 shows the results for the 3D case. All results are the average of 10 independent experiments per benchmark with the same parameters, while the run-time is reported in minutes for a Dual-Core AMD Opteron$^{\text{TM}}$ Processor 2212 machine.

When compared to the results obtained for the LSA procedure as presented in [14,38], one can see that the population-based local search converges faster to optimum conformations on the sequences under consideration, with a particular speed-up on 3D benchmarks.

Table 2. 2D HP results

Seq.	KSteps	$\Gamma = \sqrt{n}/2$	Generations	Transitions	Run-time	Energy
S36	1000	3	4,996	28,018	0.41	-14
S48	1000	3	203,909	989,984	13.81	-23
S60	1000	4	13,753	45,328	0.81	-35
S64	2000	4	1,926,834	4,325,433	73.37	-42
S85	2000	5	902,778	5,193,968	51.97	-53
S100	2000	5	8,792,302	119,487,437	3,879.77	-48

Table 3. 3D HP results

Seq.	KSteps	$\Gamma = n^{2/3}/2$	Generations	Transitions	Run-time	Energy
S1	1000	7	371,404	1,564,737	75.71	-32
S2	1000	7	3,728,709	8,381,287	409.63	-34
S3	1000	7	959,939	2,062,400	107.86	-34
S4	1000	7	601,332	1,543,557	83.94	-33
S5	1000	7	639,200	2,450,313	126.61	-32
S6	1000	7	531,346	1,853,860	85.08	-32
S7	2000	7	45,868,511	139,606,926	7,198.13	-32
S8	1000	7	759,870	2,474,248	130.74	-31
S9	2000	7	1,197,295	7,336,260	389.92	-34
S10	1000	7	2,122,927	7,290,896	415.06	-33

5.3 MJ Model Simulations

We performed the landscape analysis devised in [11] and briefly described at the end of Section 4 on the five MJ benchmark problems listed in Section 5.1. We applied LSA for different estimations of the maximum value over all minimum escape heights from local minima, where the number of transitions is set to 20,000. Then, we are looking for the point where the minimum values of the objective function obtained for each estimation changes from better to worse. The corresponding (closest) estimation is taken as the approximation of Γ, in this case for the MJ objective function with the associated landscape. The results are shown in Table 4.

Table 4. Energy reached after $20,000$ steps

D	2	4	5	7
MJ1	-11.07	-13.34	-13.67	-10.80
MJ2	-8.84	-12.22	-11.27	-7.52
MJ3	-7.55	-10.86	-9.27	-7.33
MJ4	-5.53	-9.18	-7.21	-8.12
MJ5	-7.06	-12.81	-9.15	-6.06

As one can see from Table 4, the best results for the given number of transitions were obtained for $D = 4$. Thus, an appropriate choice for an approximation D of Γ seems to be a value from the interval $[4, 5)$.

LSA for MJ benchmarks

We executed the LSA procedure devised in [14,38] for $D = 4.5$ and obtained optimum conformations for all five benchmark problems presented in Table 1.

Table 5 shows a comparison of logarithmic simulated annealing (LSA) and Monte Carlo simulations with respect to the number of successful transitions and the actual number of enrgey function evaluations. For Monte Carlo simulations, the neighbourhood relation is given by kink-jump moves as described in [40] and utilised in [21].

Table 5. SA with pull-moves vs MC with kink-jump moves [40]

Seq.	LSA-Transitions	LSA-evaluations	MC-Evaluations	E_{native}
MJ1	730,380	3,738,152	8,128,305	-25.85
MJ2	777,727	2,923,552	2,630,268	-25.92
MJ3	639,943	2,030,358	1,513,561	-26.09
MJ4	1,352,904	6,191,319	29,512,092	-25.87
MJ5	548,991	2,042,505	19,952,623	-26.15

On MJ1, MJ4 and MJ5, the number of function evaluations executed by LSA is significantly smaller than the corresponding number for MC simulations. On MJ2 and MJ3, LSA needs more evaluations to reach optimum solutions, but the difference is not as significant as for the other three benchmarks.

Population-based local search for MJ benchmarks

Finally, we report the outcome of the experiments with the population-based local search. The parameter settings are $Ltrials = 100$, $KSteps = 1,500$, $D = 4.5$ as for LSA, and 1.5×10^6 for the number of generations.

Table 6 shows the results from five independent runs with a run-time comparable to the LSA computations. Stable results equal to optimum conformations were obtained only for MJ3. On MJ5, an optimum conformation was found in one out of the five runs. For the remaining three cases, the average difference to optimum values is about 13%.

Table 6. The values in bold are the best obtained

MJ1	MJ2	MJ3	MJ4	MJ5
-20.98	-21.16	-22.56	-20.63	-19.08
-21.22	-22.62	**-26.09**	-20.07	-21.26
-21.90	**-25.10**	**-26.09**	-20.03	-22.32
-21.00	-25.09	**-26.09**	**-20.84**	**-26.15**
-21.23	-23.71	**-26.09**	-20.91	-19.99

6 Concluding Remarks

The population-based local search proposed in this paper has been compared to our previous work where a simulated annealing algorithm with logarithmic cooling schedule (LSA) was applied to protein folding simulation in the HP model [14]. In our current computational experiments we observed that the population-based local search with LSA pre-processing converges faster to optimum conformations of HP benchmarks than our LSA procedure. On the other hand, it performs worse on MJ benchmarks, where LSA finds optimum conformations on all five benchmarks. Moreover, on three of the MJ benchmarks the LSA procedure is much faster than MC simulations and has a comparable speed on the remaining two benchmarks. Thus, future research will concentrate on fine-tuning of population-based local search, in particular, with respect to the selection of subsequent populations for the re-start of guided local search.

References

1. Campbell, P.N., Smith, A.D., Peters, T.J.: Biochemistry Illustrated: Biochemistry and Molecular Biology in the Post-genomic Era. 5th edn., Churchill Livingstone, Edinburgh (2005)
2. Anfinsen, C.B.: Principles that govern the folding of protein chains. Science 181, 223–230 (1973)
3. Levinthal, C.: Are there pathways for protein folding? J. de Chimie Physique et de Physico-Chimie Biologique 65, 44–45 (1968)
4. Govindarajan, S., Goldstein, R.A.: On the thermodynamic hypothesis of protein folding. Proc. Natl. Acad. Sci. USA 95, 5545–5549 (1998)
5. Greenberg, H.J., Hart, W.E., Lancia, G.: Opportunities for combinatorial optimization in computational biology. INFORMS J. Comput. 16, 211–231 (2004)
6. Paterson, M., Przytycka, T.: On the complexity of string folding. Discrete Appl. Math. 71, 217–230 (1996)
7. Berger, B., Leighton, T.: Protein folding in the hydrophobic-hydrophilic (HP) model is NP-complete. J. Comput. Biol. 5, 27–40 (1998)
8. Saibil, H.R., Ranson, N.A.: The chaperonin folding machine. Trends Biochem. Sci. 27, 627–632 (2002)
9. Spiess, C., Meyer, A.S., Reissmann, S., Frydman, J.: Mechanism of the eukaryotic chaperonin: protein folding in the chamber of secrets. Trends Cell Biol. 14, 598–604 (2004)

10. Homer, S., Selman, A.L.: Computability and Complexity Theory. Springer, New York (2001)
11. Zahrani, M.S., Loomes, M.J., Malcolm, J.A., Ullah, A.Z.M.D., Steinhöfel, K., Albrecht, A.A.: Genetic local search for multicast routing with pre-processing by logarithmic simulated annealing. Comput. Oper. Res. 35, 2049–2070 (2008)
12. Xiaoming, D., Runmin, Z., Rong, S., Rui, F., Shao, H.: Convergence properties of non-crossover genetic algorithm. In: Proc. IEEE 4^{th} World Congress on Intelligent Control and Automation., vol. 3, pp. 1822–1826 (2002)
13. Rogers, A., Prügel-Bennett, A., Jennings, N.R.: Phase transitions and symmetry breaking in genetic algorithms with crossover. Theor. Comput. Sci. 358, 121–141 (2006)
14. Steinhöfel, K., Skaliotis, A., Albrecht, A.A.: Stochastic Protein Folding Simulation in the d-dimensional HP-Model. In: Hochreiter, S., Wagner, R. (eds.) BIRD 2007. LNCS (LNBI), vol. 4414, pp. 381–394. Springer, Heidelberg (2007)
15. Dill, K.A., Bromberg, S., Yue, K., Fiebig, K.M., Yee, D.P., Thomas, P.D., Chan, H.S.: Principles of protein folding - A perspective from simple exact models. Protein Sci. 4, 561–602 (1995)
16. Miyazawa, S., Jernigan, R.L.: Estimation of effective interresidue contact energies from protein crystal structures: quasi-chemical approximation. Macromolecules 18, 534–552 (1985)
17. Lesh, N., Mitzenmacher, M., Whitesides, S.: A complete and effective move set for simplified protein folding. In: Proceedings of the 7th Annual International Conference on Computational Biology, pp. 188–195. ACM Press, New York (2003)
18. Milostan, M., Lukasiak, P., Dill, K.A., Blazewicz, A.: A tabu search strategy for finding low energy structures of proteins in HP-model. In: Proceedings of the 7th Annual International Conference on Computational Biology, pp. 205–206. ACM Press, New York (2003)
19. Blazewicz, J., Lukasiak, P., Milostan, M.: Application of tabu search strategy for finding low energy structure of protein. Artif. Intell. Med. 35, 135–145 (2005)
20. Unger, R., Moult, J.: Genetic algorithms for protein folding simulations. J. Mol. Biol. 231, 75–81 (1993)
21. Faisca, P., Plaxco, K.: Cooperativity and the origins of rapid, single-exponential kinetics in protein folding. Protein Sci. 15, 1608–1618 (2006)
22. Shakhnovich, E.I.: Protein design: a perspective from simple tractable models. Fold. Design 3, R45–R58 (1998)
23. Fan, K., Wang, W.: What is the minimum number of letters required to fold a protein? J. Mol. Biol. 328, 921–926 (2003)
24. Li, H., Tang, C., Wingreen, N.S.: Designability of protein structures: a lattice–model study using the Miyazawa-Jernigan matrix. PROTEINS: Structure, Function, and Genetics 49, 403–412 (2002)
25. Shell, M.S., Debenedetti, P.G., Panagiotopoulos, A.Z.: Computational characterization of the sequence landscape in simple protein alphabets. PROTEINS: Structure, Function, and Bioinformatics 62, 232–243 (2006)
26. Mirny, L.A., Shakhnovich, E.I.: Protein structure prediction by threading. why it works and why it does not. J. Mol. Biol. 283, 507–526 (1998)
27. Khimasia, M., Coveney, P.: Protein structure prediction as a hard optimization problem: The genetic algorithm approach. Mol. Sim. 19, 205–226 (1997)
28. Konig, R., Dandekar, T.: Improving genetic algorithms for protein folding simulations by systematic crossover. Biosystems 50, 17–25 (1999)

29. Cotta, C.: Protein structure prediction using evolutionary algorithms hybridized with backtracking. In: Mira, J., Álvarez, J.R. (eds.) IWANN 2003. LNCS, vol. 2687, pp. 321–328. Springer, Heidelberg (2003)
30. Patton, A.L., Punch, W.F., Goodman, E.D.: A standard GA approach to native protein conformation prediction. In: Proceedings of the 6^{th} International Conference on Genetic Algorithms, pp. 574–581. ACM Press, New York (1995)
31. Hoque, M.T., Chetty, M., Dooley, L.S.: A new guided genetic algorithm for 2D hydrophobic-hydrophilic model to predict protein folding. In: IEEE Congress on Evolutionary Computation (CEC 2005), Edinburgh, pp. 259–266 (2005) ISBN 0-7803-9364-3
32. Liang, F., Wong, W.H.: Evolutionary Monte Carlo for protein folding simulations. J. Chem. Phys. 115, 3374–3380 (2001)
33. Liang, F., Wong, W.H.: Protein folding simulations of the hydrophobic-hydrophilic model by combining tabu search with genetic algorithms. J. Chem. Phys. 119, 4592–4596 (2003)
34. Hsu, H.P., Mehra, V., Nadler, W., Grassberger, P.: Growth-based optimization algorithm for lattice heteropolymers. Phys. Rev. E 68, 21113 (2003)
35. Zhang, J.L., Liu, J.S.: A new sequential importance sampling method and its application to the two-dimensional hydrophobic-hydrophilic model. J. Chem. Phys. 117, 3492–3498 (2002)
36. Shmygelska, A., Hoos, H.: An ant colony optimisation algorithm for the 2D and 3D hydrophobic polar protein folding problem. BMC Bioinformatics 6 (2005)
37. Thachuk, C., Shmygelska, A., Hoos, H.: A replica exchange Monte Carlo algorithm for protein folding in the HP model. BMC Bioinformatics 8 (2007)
38. Steinhöfel, K., Skaliotis, A., Albrecht, A.A.: Relating Time Complexity of Protein Folding Simulation to Approximations of Folding Time. Comp. Phys. Comm. 176, 465–470 (2007)
39. Shakhnovich, E.I., Gutin, A.M.: Engineering of stable and fast-folding sequences of model proteins. Proc. Natl. Acad. Sci. USA 90, 7195–7199 (1993)
40. Landau, D.P., Binder, K.A.: A guide to Monte Carlo simulations in statistical physics, pp. 122–123. Cambridge University Press, Cambridge (2000)
41. Wyman, J., Allen, D.W.: The problem of the heme interactions in hemoglobin and the basis of the Bohr effect. J. Polym. Sci. 7, 499–518 (1951)

A Robust Class of Stable Proteins in the 2D HPC Model

Alireza Hadj Khodabakhshi, Ján Maňuch, Arash Rafiey, and Arvind Gupta

School of Computing Science
8888 University Drive, Simon Fraser University
Burnaby, BC, V5A 1S6, Canada
alireza@cs.sfu.ca, jmanuch@sfu.ca, arashr@cs.sfu.ca, arvind@mitacs.ca

Abstract. The inverse protein folding problem is that of designing an amino acid sequence which has a prescribed native protein fold. This problem arises in drug design where a particular structure is necessary to ensure proper protein-protein interactions. The input to the inverse protein folding problem is a shape and the goal is to design a protein sequence with a unique native fold that closely approximates the input shape. Gupta *et al.* [10] introduced a design in the 2D HP model of Dill that can be used to approximate any given (2D) shape. They conjectured that the protein sequences of their design are stable but only proved the stability for an infinite class of very basic structures. In [11], we have introduced a refinement of the HP model, in which the cysteine and non-cysteine hydrophobic monomers are distinguished and SS-bridges which two cysteines can form are taken into account in the energy function. This model was called the 2D HPC model. In [11], the snake structures in the HPC model were introduced and it was conjectured that they are stable. In this paper, we show that this conjecture is true for a subclass of snake structures. This subclass is robust enough to approximate any given 2D shape, although more coarsely than the general constructible structures proposed in [10]. In the proof we use a semi-automated tool 2DHPSolver developed in [11].

1 Introduction

It has long been known that protein interactions depend on their native three-dimensional fold and understanding the processes and determining these folds is a long standing problem in molecular biology. Naturally occurring proteins fold so as to minimize total free energy. However, it is not known how a protein can choose the minimum energy fold amongst all possible folds [9].

Many forces act on the protein which contribute to changes in free energy including hydrogen bonding, van der Waals interactions, intrinsic propensities, ion pairing, disulfide bridges and hydrophobic interactions. Of these, the most significant is hydrophobic interaction [8]. This led Dill to introduce the *Hydrophobic-Polar model* [7]. Here the 20 amino acids from which proteins are formed are replaced by two types of monomers: hydrophobic (H or '1') or polar (P or '0') depending on their affinity to water. To simplify the problem, the protein is laid out on vertices of a lattice with each monomer occupying exactly one vertex and neighboring monomers occupy neighboring vertices. The free energy is minimized when the maximum number of hydrophobic monomers are adjacent in the lattice. Therefore, the "native" folds are those with the maximum number of such HH contacts. Even though the HP model is the simplest model of the protein folding process, computationally it is an NP-hard problem for both the two-dimensional [5] and the three-dimensional [2] square lattices.

In many applications such as drug design, we are interested in the complement problem to protein folding: *inverse protein folding* or *protein design*. The *inverse protein folding problem* involves starting with a prescribed target fold or structure and designing an amino acid sequence whose native fold is the target. The success of a protein design method is evaluated based on three criteria [6,22]. First the protein sequence should fold to the target conformation. This means that the native fold of the sequence should be the target conformation. This criteria is often called *positive design*. Second, the target conformation should be the only native fold of the sequence. This criteria is referred to as *negative design*. Third, there should be a large gap in the energy of the native (target) fold of the sequence and the energy of any other fold of the sequence. The computational complexity of the inverse protein folding problem (IPF) is still unknown but it has been conjectured that IPF is in fact intractable [12]. Several heuristic based algorithms have been described that attempt to solve IPF problem [15,16,20,22] but they do not guarantee that the designed sequences satisfy the positive and negative design criteria. These heuristic methods can be separated into two categories. The methods in the first category use observations about the properties of proteins to justify algorithms that design sequences [15,22]. The second category of heuristic methods are those in which an alternative formulation of IPF is considered [6,16,20,21]. This alternative formulation attempts to capture the positive and negative design issues by defining a heuristic sequence design (HSD) problem. In [12] and [3] the computational complexity of two HSD problems the *canonical* and the *grand canonical* models, introduced in [20] and [21] respectively, were studied. In the canonical model the number of hydrophobic monomers that can be used in a protein sequence is limited by fixing the ratio between hydrophobic and hydrophilic amino acids. The intuition behind this model is the fact that for any target conformation, the conformational energy can be minimized simply by using the sequence of all hydrophobic monomers, but this sequence is unlikely to achieve its lowest energy with the given target conformation. In the grand canonical model, the number of hydrophobic monomers is limited by adjusting the contact energy (energy function) instead. More specifically, the contact energy gives an energy of -2 to hydrophobic-hydrophobic contacts, an energy 1 for every solvent accessible site on a hydrophobic amino acid, and 0 for all other interactions. Because hydrophobic monomers are penalized for their exposure to solvent, this contact potential implicitly limits the number of hydrophobics in the sequence.

It has been shown that the protein sequence design problem can be solved in polynomial time in the grand canonical model for both 2D and 3D square lattices, cf. [12], and in polynomial time for 2D lattices while the problem is NP-hard for 3D square lattice in the canonical model, cf. [3,4]. Note however that the designed sequences under these two models do not guarantee that the generated sequence satisfies the two criteria (positive and negative design) of the inverse protein folding problem.

In Gupta *et al.* [10], the IPF problem was studied from a different perspective. Instead of designing a sequence directly for the target fold and relaxing conditions the sequence has to satisfied, they introduced a design method in 2D square lattice under the HP model that can approximate any target conformation and it was shown that approximated structures are native for designed proteins (positive design). It was conjectured that the protein sequences of their designed structures are also stable but only proved for an infinite class of very basic structures (arbitrary long "I" and "L" shapes), as well as computationally tested for over 48,000 structures (including all with up to 9 tiles). Design of stable proteins of arbitrary lengths in the HP model was also studied by Aichholzer *et al.* [1] (for 2D square lattice) and by Li *et al.* [17] (for 2D triangular lattice), motivated by a popular paper of Brian Hayes [13].

In natural proteins, sulfide bridges between two cysteine monomers play an important role in improving stability of the protein structure [14]. In our previous work [11] we extended the HP model by adding the third type of monomers, cysteines, and incorporating sulfide bridges between two cysteines into energy model. This model is called the HPC (hydrophobic-polar-cysteine) model. The cysteine monomers in the HPC model act as hydrophobic, but in addition two neighboring cysteines can form a sulfide-sulfide bridge to further reduce the energy of the fold. Therefore, between many folds of the same protein with the same number of hydrophobic bonds the one with the maximum number of sulfide bridges is the most stable fold. This added level of stability can help in proving formally that the designed proteins are indeed stable.

In [11] we introduced a class of structures called the *snake structures*. The class of snake structures is a subset of the class *linear structures* introduced by Gupta *et al.* [10]. The linear structures are formed by a sequence of "plus" shape tiles, cf. Figure 1(a), connected by overlapping two pairs of polar monomers (each coming from a different tile). The structures are linear which means that every tile except the first and the last is attached to exactly two other tiles. In the snake structures every second tile is a bending tile. The first, last and the bending tiles in a snake structure contain cysteine monomers while the rest of the tiles contain hydrophobic monomers. In [11] we conjectured that the protein of snake structures are stable and we proved it under an additional assumption that non-cysteine hydrophobic monomers act as cysteine ones, i.e., they tend to form their own bridges to reduce the energy. This model was called the strong HPC mode. Even though this model is artificial, we used it to demonstrate that our techniques can be used to prove stability of snake structures in the "proper" HPC model.

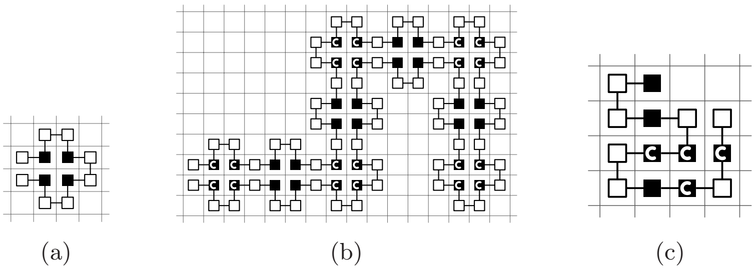


Fig. 1. (a) The basic building tile for constructible structures: black squares represent hydrophobic and white polar monomers. The lines between boxes represent the peptide bonds between consecutive monomers in the protein string. (b) An example of snake structure. The bending tiles use cysteines (black squares marked with C). (c) Example of energy calculation of a fold in HPC model. There are 5 contacts between hydrophobic monomers, thus the contact energy is -5. There are three potential sulfide bridges sharing a common vertex, hence only one can be used in the maximum matching. Thus the sulfide bridge energy is -2 and the total energy is -7.

In this paper we consider a subclass of snake structures which is robust enough to approximate any given shape and the same time restricted enough to be proved stable using our techniques. We call this subclass *wave structures*. The wave structures are instances of the snake structures that do not contain any occurrence of the four forbidden motifs in Figure 2. We believe this a first robust design formally provable that it is stable.

This paper is organized as follows. We start by the definition of the HPC model and introducing the wave structures in Section 2. In Section 3 we explain our proof techniques and used them to prove the protein of any wave structure is stable.

2 Definitions

In this section we define the HPC model introduced in [11] as extension of the HP model of Dill [7] and introduce wave structures.

2.1 Hydrophobic-Polar-Cysteine (HPC) Model

Proteins are chains of 20 types of amino acids. In the HPC model, we consider only 3 types of amino acids: polar, cysteine and non-cysteine hydrophobic. We can represent a protein chain as a string $p = p_1 p_2 \ldots p_{|p|}$ in $\{0, 1, 2\}^*$, where "0" represents a polar monomer, "1" a hydrophobic non-cysteine monomer and "2" a cysteine monomer.

The proteins are folded onto the regular lattice. A *fold* of a protein p is embedding of a path of length n into lattice, i.e., vertices of the path are mapped into distinct lattice vertices and two consecutive vertices of the path are mapped

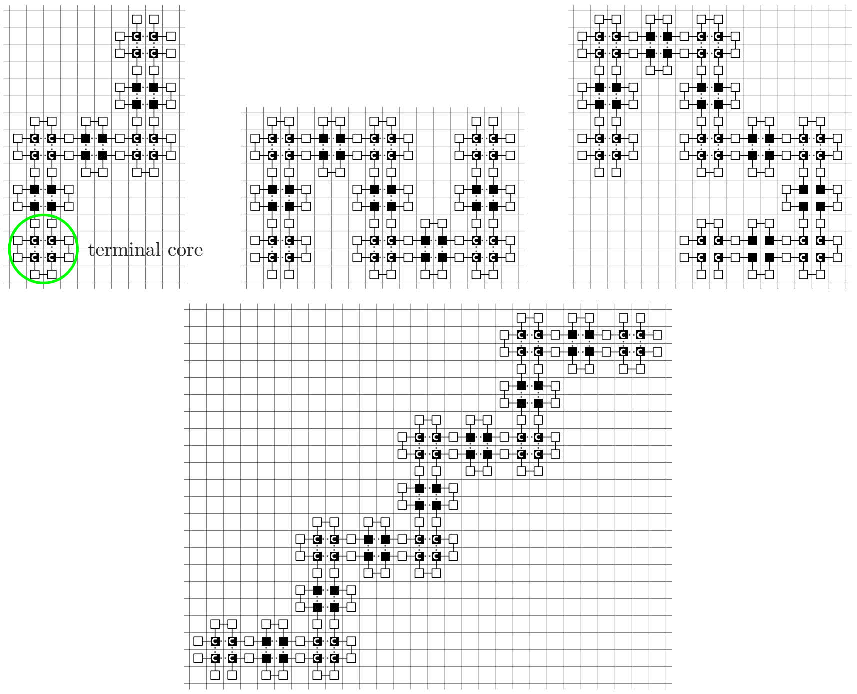


Fig. 2. Forbidden motifs in wave structures

to lattice vertices connected by an edge (a peptide bond). In this paper we use the 2D square lattice.

A protein will fold into a fold with the minimum free energy, also called a *native fold*. In the HP model only hydrophobic interactions between two adjacent hydrophobic monomers which are not consecutive in the protein sequence (*contacts*) are considered in the energy model, with each contact contributing with -1 to the total energy. In addition, in the HPC model, two adjacent non-consecutive cysteines can form a sulfide bridge contributing with -2 to the total energy. However, each cysteine can be involved in at most one sulfide bridge. More formally, any two adjacent non-consecutive hydrophobic monomers (cysteine or non-cysteine) form a contact and the contact energy is equal to -1 times the number of contacts; and any two adjacent non-consecutive cysteines form a *potential* sulfide bridge and the sulfide-bridge energy is equal to -2 times the number of matches in the maximum matching in the graph of potential sulfide bridges. The total energy is equal to the sum of the contact and sulfide bridge energies. For example, the energy of the fold in Figure 1(c) is $(-5)+(-2) = -7$. (Note that the results in the paper are independent on the exact value of the energy of sulfide bridge, as long as it is negative, and therefore we did not research on determination of the correct value for this energy.)

There might be several native folds for a given protein. A protein with a unique native fold is called *stable* protein.

2.2 Wave Structures

In Gupta *et al.* [10], a wide class of 2D structures, called *constructible structures*, was introduced. They are formed by a sequence of "plus" shape tiles, cf. Figure 1(a), connected by overlapping two pairs of polar monomers (each coming from different tile). It was conjectured that these structures are stable and proved for two very simple subclasses of the linear structures, namely for L_0 and L_1 structures. The L_0 and L_1 structures consist of an arbitrary large sequence of tiles in the shape of a straight line and the letter L, respectively. Note that although L_1 structures are still quite simple, the proof of their stability involves analysis of a large number of cases. In our previous work [11], we introduced a subclass of constructible structures, snake structures, and refine it for the HPC model with nice combinatorial properties, e.g., in the proteins of such structures any two consecutive hydrophobic monomers are of the same type if there two polar monomers between them and are of different type if there is one polar monomer between them. This significantly reduces the case analysis and we conjectured that the snake structures are stable.

The snake structures are *linear* structures which means that every tile t_i except the first t_1 and the last t_n is attached to exactly two other tiles t_{i-1} and t_{i+1} (and the first and the last ones are attached to only one tile, t_2 and t_{n-1}, respectively). In addition, in a snake structure the sequence of tiles has to change direction ("bend") in every odd tile. The hydrophobic monomers of these "bending" tiles are set to be cysteines, and all other hydrophobic monomers are non-cysteines, cf. Figure 1(b). Although, the snake structures are more restricted, the proof of their stability under the HPC model required the analysis of huge number of cases. However, in [11] we were able to prove that they are stable under the artificial strong HPC model. This model assumes that the non-cysteine hydrophobic monomers form SS-bridges of their own to reduce the energy of the conformation. Notice that cysteine and none-cysteine monomers cannot form SS-bridges. Although, the strong HPC model is not a proper biological model, the proof of the stability of the snake structures under the strong HPC model raised the hope for finding the structures that can be proved to be stable under the proper HPC model.

In this paper, we introduce a subclass of the snake structures called the *wave structures* and formally prove that they are stable under the proper HPC model. Although, the wave structures is only a subclass of the snake structures they can still approximate any given shape in 2D square lattice. The wave structures are instances of the snake structures that do not contain occurrence of the four forbidden motifs in Figure 2. The wave structures can be constructed using a set of four super-tiles and their flipped versions (cf. Figure 3).

The super-tiles are simple instances of the constructible structures. The *starting* super-tile has one receptor and consists of two basic tiles (Figure 3(a)), the *terminating* super-tile has one ligand and consists of 5 basic tiles (Figure 3(b)),

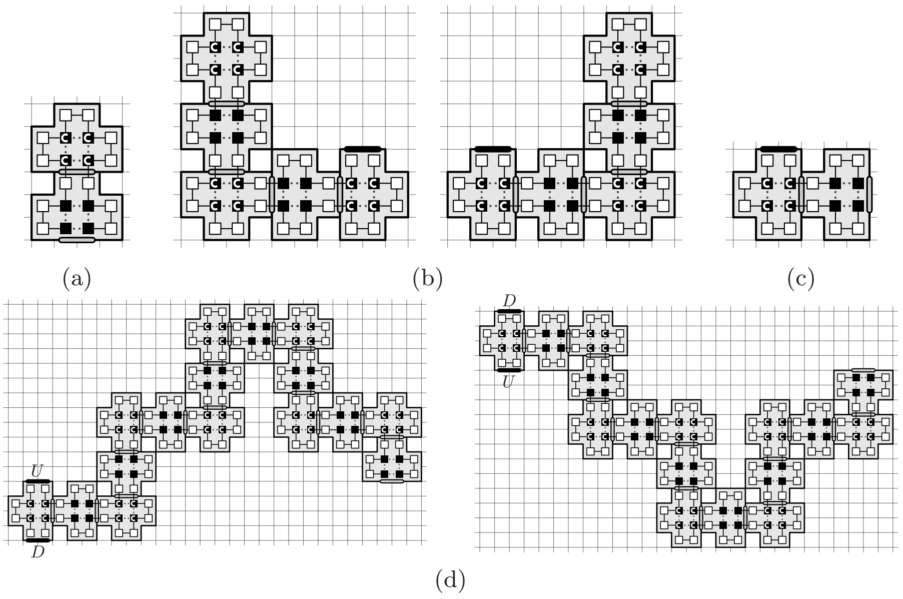


Fig. 3. Super-tiles used to construct wave structures: (a) starting super-tile; (b) unflipped and flipped versions of terminating super-tile; (c) bending super-tile; and (d) flipped and non-flipped versions of regular tile

the *bending* super-tile has one ligand and one receptor and consists of two tiles (Figure 3(c)), and the *regular* super-tile has two ligands and one receptor and consists of 16 basic tiles (Figure 3(d)). The receptor of one super-tile can connect to the ligand of another one however, the regular super-tile must only connect through one of its ligands. A wave structure is a partial tiling of the two-dimensional grid obtained by the following procedure.

1. Place the starting super-tile into the grid and place a regular super-tile into the grid so that its U ligand is attached to the receptor of the starting super-tile.
2. Let the last placed super-tile be a (flipped) regular super-tile R; either place a (flipped) regular super-tile so that its U ligand is attached to the receptor of R and continue with step 4 or place a bending super-tile such that its ligand is attached to receptor of R and continue with step 3.
3. Let the last placed super-tile be a bending super-tile B and let R be a regular super-tile attached to B. If R is a flipped super-tile then attach a new non-flipped regular super-tile to B otherwise, attach a new flipped super-tile to B. The new super-tile can be attached either with U or D ligand depending on intended direction of the bend.
4. Continue with step 2 or end the structure by attaching a (flipped) terminating super-tile to the last placed (flipped) regular super-tile.

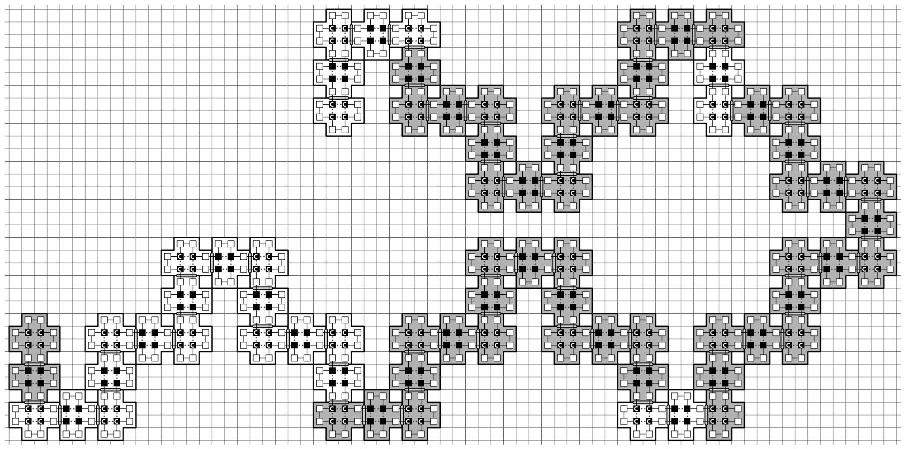


Fig. 4. An example of a wave structure. It consists of 8 super-tiles. The borders between super-tiles are marked by the change of underlying color of the core tiles.

In the above procedure the super-tiles are placed into the grid such that they do not overlap. An example of a wave structure is depicted in Figure 4.

As observed in [11] for snake structures, approximately 40% of all monomers in wave structures are hydrophobic and half of those are cysteines. Thus approximately 20% of all monomers are cysteines. Although, the most of naturally occurring proteins have much smaller frequency of cysteines, there are some with the same or even higher ratios: 1EZG (antifreeze protein from the beetle [18]) with 19.5% ratio of cysteines and the protein isolated from the chorion of the domesticated silkmoth [19] with 30% ratio.

Note that the wave structures can still approximate any given shape, although more coarsely than the linear/snake structures. The idea of approximating a given shape with a linear structure is to draw a non-intersecting curve consisting of horizontal and vertical line segments. Each line segment is a linear chain of basic tiles depicted in Figure 1(a). At first glance, the wave structures seem more restricted than linear structures, as the line segments they use are very short and have the same size (3 tiles long). However, one can simulate arbitrary long line segments with wave structures forming a zig-zag pattern, cf. Figure 5.

We prove that the proteins for the wave structures are stable in the HPC model. Our techniques to achieve this include (i) the case analysis (also used in Gupta *et al.* [10]) and (ii) the induction on diagonals. Furthermore, to increase the power of the case analysis technique, we used a program called "2DHP-Solver" for semi-automatic proving of hypothesis about the folds of proteins of the designed structures developed in [11]. Note that 2DHPSolver can be used for all three models: HP, HPC and strong HPC by setting the appropriate parameters.

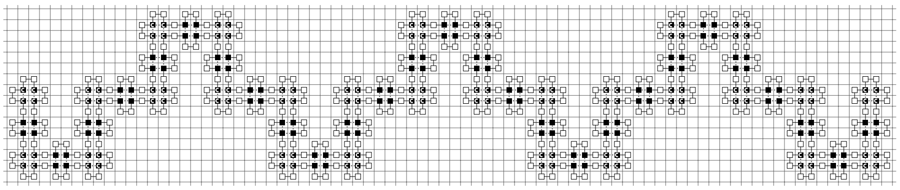


Fig. 5. Simulation of a straight line segment with a wave structure

3 Stability of the Wave Structures

In this section we prove that the protein of any wave structure is stable. In the proof we will use a concept of saturated structures and 2DHPSolver tool developed in [11]. Let us briefly introduce them.

3.1 Saturated Folds

The proteins used by Gupta *et al.* [10] in the HP model and the wave proteins in HPC have a special property. The energy of their native folds is the smallest possible with respect to the numbers of hydrophobic cysteine and non-cysteine monomers contained in the proteins. We call such folds *saturated*. In saturated folds all parts of energy function produce minimum possible values. This means: (i) every hydrophobic monomer (cysteine or non-cysteine) has two contacts with other monomers; (ii) there is a sulfide bridge matching containing all or all but one cysteine monomers. Obviously, a saturated fold of a protein must be native, and furthermore, if there is a saturated fold of a protein, then all native folds of this protein must be saturated.

3.2 2DHPSolver: A Semi-automatic Prover

2DHPSolver is a tool for proving the uniqueness of a protein design in 2D square lattice under the HP, HPC or strong HPC models developed in [11]. 2DHP-Solver is not specifically designed to analyze the wave structures or even the constructible structures. It can be used to prove the stability of any 2D HP design based on the induction on the boundaries. It starts with an initial configuration (initial field) which is given as the input to the program. In each iteration, one of the fields is replaced by all possible extensions at one point in the field specified by user. Note that in displayed fields red 1 represents a cysteine monomer, blue 1 a non-cysteine monomer and finally, uncolored 1 is hydrophobic monomer, but it is not known whether it is cysteine or not.

These extensions are one of the following type:

- extending a path (of consecutive monomers in the protein string);
- extending a 1-path (of a chain of hydrophobic monomers connected with contacts);
- coloring an uncolored H monomer.

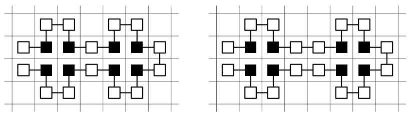


Fig. 6. Configurations with correctly aligned cores

There are 6 ways to extend a path, 3 ways to extend a one-path and 2 ways to color an uncolored H monomer. For each of these possibilities, 2DHPSolver creates a new field which is then checked to see if it violates the rules of the design. Those which do not violate the design rules will replace the original field.

However, this approach will result in producing too many fields, which makes it hard for the user to keep track of. Therefore, 2DHPSolver contains utilities to assist in automatically finding an extending sequence for a field which leads to either no valid configurations, in which case the field is automatically removed, or to only one valid configuration, in which case the field is replaced by the new more completed configuration. This process is referred to as a *self-extension*. The time required for searching for such extending sequence depends on the depth of the search, which can be specified by user through two parameters "depth" and "max-extensions". Since the configurations space is infinite it is not possible to perform an exhaustive search by setting these parameters to high values. Instead, one should set parameters to moderate values and use intuition in choosing the next extension point when 2DHPSolver is unable to automatically find self-extending sequences. Note that these parameters can be changed at any time during the use of the program by the user.

2DHPSolver is developed using C++ and its source code is freely available to all users under the GNU Public Licence (GLP). For more information on 2DHPSolver and to obtain a copy of the source codes please visit `http://www.sfu.ca/~ahadjkho/2dhpsolver/`.

3.3 Proof

Let S be a wave structure, p its protein and let F be an arbitrary native (i.e., saturated) fold of p.

Define a path in F as a sequence of vertices such that no vertex appears twice and any pair of consecutive vertices in the path are connected by peptide bonds. A cycle is a path whose start and end vertices are connected by a peptide bond. For $i \in \{0, 1, 2\}$, an i-vertex in the fold F is a lattice vertex (square) containing a monomer i. For instance, a square containing a cysteine monomer in F is called a 2-vertex. An H-vertex is a vertex which is either 1-vertex or 2-vertex. Define a 1-path in F to be a sequence of H-vertices such that each H-vertex appears once and any pair of consecutive ones form an HH contact. A 1-cycle in F is a 1-path whose first and last vertices form an HH contact. A 1-cycle of length 4 is called a core in F.

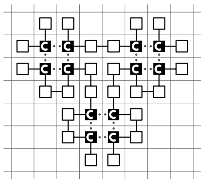


Fig. 7. Configuration with misaligned cores

A core c is called *monochromatic* if all its H-vertices are either cysteines or non-cysteines. Let c_1 and c_2 be two cores in F. We say, c_1 and c_2 are adjacent if there is a path of length 2 or 3 between an H-vertex of c_1 and an H-vertex of c_2. We say c_1 and c_2 are correctly aligned if they are adjacent in one of the forms in Figure 6.

In what follows we prove that every H-vertex in F belongs to a monochromatic core and the cores are correctly aligned.

Lemma 1. *Every H-vertex in F belongs to a monochromatic core and either all the cores are correctly aligned or there are three cores in F that are not correctly aligned while all other cores are correctly aligned and these three cores form the configuration depicted in Figure 7.*

Proof. For any integer i, let SW_i be the set of lattice vertices $\{[x, y]; x + y = i\}$. Let m be the maximum number such that SW_i, $i < m$ does not contain any H-vertex, i.e., SW_m is a boundary of diagonal rectangle enclosing all H-vertices.

We start by proving the following claim.

Claim. Every H-vertex in F belongs to a monochromatic core.

Proof. We prove the claim by induction on SW_k, i.e., we prove that for every k and every H-vertex v on SW_k, v is in a monochromatic core. For the base case, consider smallest k such that for $i \geq k$, there is no H-vertex on SW_i. Then the claim is trivially true. For induction step, it is enough to show that for every k, if for every H-vertex v on SW_i, $i > k$, v is in a monochromatic core, then for every H-vertex w on SW_k, w is on a monochromatic core c.

Fix k and by induction hypothesis, assume that for every H-vertex v lying on SW_i, where $i > k$, v belongs to a monochromatic core. Consider an H-vertex w on SW_k. We show that if w is not on a monochromatic core then we see a subsequence in F which is not in p or an unpaired cysteine monomer. This is done by enumerative case analysis of all possible extensions of this configuration and showing that each branch will end in a configuration that has a subsequence not in p or has an unpaired cysteine monomer.

This process requires the analysis of many configurations which is very hard and time consuming to do manually. Therefore, we used 2DHPSolver to assist in analyzing the resulting configurations. The program generated proof of this step of the induction can be found on our website at
`http://www.sfu.ca/~ahadjkho/2dhpsolver/core-monochromatic-proof`.

Finally we showed the following claim using the 2DHPsolver tool.

Claim. Let c_1 and c_2 be two adjacent monochromatic cores in F. Then either c_1 and c_2 are aligned correctly or there is a third core c_3 such that c_1, c_2 and c_3 form the configuration in Figure 7.

The program generated proof of this claim can be found on our website at `http://www.sfu.ca/~ahadjkho/2dhpsolver/core-alignment-proof`.

The main result follows from the previous lemma and the proof of the main result in [11].

Theorem 1. *Every* H*-vertex in F belongs to a monochromatic core and all the cores are correctly aligned. Hence, $F = S$, i.e., all wave structures are stable.*

4 Conclusions

In this paper we introduce a robust subclass of constructible structures introduced by Gupta *et al.* [10], refine these structures for the HPC model [11] and prove that these structures are stable. Since this subclass is robust enough to approximate any given shape, although more coarsely than the class of all constructible structures, the result in this paper partially verifies the conjecture in [10].

Furthermore, our result shows that use of cysteines in the design of proteins might help to improve their stability. To further verify this, in the future, we would like to extend our results to 3D lattice models and test them using existing protein folding software.

References

1. Aichholzer, O., Bremner, D., Demaine, E.D., Meijer, H., Sacristán, V., Soss, M.: Long proteins with unique optimal foldings in the H-P model. Computational Geometry: Theory and Applications 25(1-2), 139–159 (2003)
2. Berger, B., Leighton, T.: Protein folding in the hydrophobic-hydrophilic (HP) model is NP-complete. J. Comp. Biol. 5(1), 27–40 (1998)
3. Berman, P., DasGupta, B., Mubayi, D., Sloan, R., Turán, G., Zhang, Y.: The protein sequence design problem in canonical model on 2D and 3D lattices. In: Sahinalp, S.C., Muthukrishnan, S.M., Dogrusoz, U. (eds.) CPM 2004. LNCS, vol. 3109, pp. 244–253. Springer, Heidelberg (2004)
4. Berman, P., DasGupta, B., Mubayi, D., Sloan, R., Turán, G., Zhang, Y.: The inverse protein folding problem on 2D and 3D lattices. Discr. Appl. Math. 155, 719–732 (2007)
5. Crescenzi, P., Goldman, D., Papadimitriou, C., Piccolboni, A., Yannakakis, M.: On the complexity of protein folding. In: Proc. of STOC 1998, pp. 597–603 (1998)
6. Deutsch, J.M., Kurosky, T.: New algorithm for protein design. Physical Review Letters 76, 323–326 (1996)
7. Dill, K.A.: Theory for the folding and stability of globular proteins. Biochemistry 24(6), 1501–1509 (1985)

8. Dill, K.A.: Dominant forces in protein folding. Biochemistry 29(31), 7133–7155 (1990)
9. Dill, K.A., Bromberg, S., Yue, K., Fiebig, K.M., Yee, D.P., Thomas, P.D., Chan, H.S.: Principles of protein folding: A perspective from simple exact models. Protein Science 4, 561–602 (1995)
10. Gupta, A., Maňuch, J., Stacho, L.: Structure-approximating inverse protein folding problem in the 2D HP model. Journal of Computational Biology 12(10), 1328–1345 (2005)
11. Hadj Khodabakhshi, A., Maňuch, J., Rafiey, A., Gupta, A.: Structure-approximating design of stable proteins in 2D HP model fortified by cysteine monomers. In: Proc. of APBC 2008, pp. 49–58 (2008)
12. Hart, W.E.: On the computational complexity of sequence design problems. In: Proc. of the First Annual International Conference on Computational Molecular Biology, pp. 128–136 (1997)
13. Hayes, B.: Prototeins. American Scientist 86, 216–221 (1998)
14. Jaenicke, R.: Protein stability and molecular adaptation to extreme conditions. Eur. J. Biochem. 202, 715–728 (1991)
15. Kamtekar, S., Schiffer, J.M., Xiong, H., Babik, J.M., Hecht, M.H.: Protein design by binary patterning of polar and nonpolar amino acids. Science 262, 1680–1685 (1993)
16. Kurosky, T., Deutsch, J.M.: Design of copolymeric materials. Physics A: Mathematical and General 28, 387–393 (1995)
17. Li, Z., Zhang, X., Chen, L.: Unique optimal foldings of proteins on a triangular lattice. Appl. Bioinformatics 4(2), 105–116 (2005)
18. Liou, Y.C., Tocilj, A., Davies, P.L., Jia, Z.: Mimicry of ice structure by surface hydroxyls and water of a beta-helix antifreeze protein. Nature 406, 322–324 (2000)
19. Rodakis, G.C., Kafatos, F.C.: Origin of evolutionary novelty in proteins: How a high-cysteine chorion protein has evolved. Proc. Natl. Acad. Sci. USA 79, 3551–3555 (1982)
20. Shakhnovich, E.I., Gutin, A.M.: Engineering of stable and fast-folding sequences of model proteins. Proc. Natl. Acad. Sci. 90, 7195–7199 (1993)
21. Sun, S., Brem, R., Chan, H.S., Dill, K.A.: Designing amino acid sequences to fold with good hydrophobic cores. Protein Engineering 80, 1205–1213 (1995)
22. Yue, K., Dill, K.A.: Inverse protein folding problem: Designing polymer sequences. Proc. Natl. Acad. Sci. USA 89, 4163–4167 (1992)

A Novel Adaptive Multiple Imputation Algorithm

Veselka Boeva[1] and Elena Tsiporkova[2]

[1] Computer Systems and Technologies Department
Technical University of Sofia, branch Plovdiv
Tsanko Dyustabanov 25, 4000 Plovdiv, Bulgaria
vboeva@tu-plovdiv.bg
[2] Innovation Center - East Flanders
House of Economy, Seminariestraat 2
B-9000 Ghent, Belgium
elena.tsiporkova@innovatiecentrum.be

Abstract. The accurate estimation of missing values is important for efficient use of DNA microarray data since most of the analysis and clustering algorithms require a complete data matrix. Several imputation algorithms have already been proposed in the biological literature. Most of these approaches identify, in one or another way, a fixed number of neighbouring genes for the estimation of each missing value. This increases the possibility of involving in the evaluation process gene expression profiles, which are rather distant from the profile of the target gene. The latter may significantly affect the performance of the applied imputation algorithm. We propose in this article a novel adaptive multiple imputation algorithm, which uses a varying number of neighbouring genes for the estimation of each missing value. The algorithm generates for each missing value a list of multiple candidate estimation values and then selects the most suitable one, according to some well-defined criteria, in order to replace the missing entry. The similarity between the expression profiles can be estimated either with the Euclidean metric or with the Dynamic Time Warping (DTW) distance measure. In this way, the proposed algorithm can be applied for the imputation of missing values for both non-time series and time series data.

1 Introduction

DNA microarray is high-throughput technology, which is widely used nowadays to study biological processes through measuring thousands of gene expression levels simultaneously under different conditions. Unfortunately, microarray gene expression datasets, like other experimental data, often contain multiple missing values due to various reasons. However, most of gene expression data analysis techniques and methods require complete expression data. Therefore the development of robust imputation algorithms for accurate estimation of missing values in gene expression data is of crucial importance for the optimal use of such data.

Recently, several methods for missing value imputation of microarray data have been proposed in the biological literature, including row average [8], singular value decomposition (SVD) [5] and weighted K-Nearest Neighbours (KNNimpute) [15] methods. The row average estimation of missing values ignores the observed correlation structure of the genes and therefore it has limited accuracy. In general, the KNN-based methods have been proven to be rather efficient. The KNNimpute algorithm [15] uses Euclidean distance to measure the similarity between two gene profiles and selects for each gene with missing value a preliminary fixed number, the same for all missing values, of candidate genes for estimation. Kim *et al.* [7] proposed a sequential variant of KNN imputation (SKNN), which estimates the missing values sequentially from the gene having the least number of missing values, and then uses the imputed values for further imputation. Oba *et al.* [10] considered a missing value estimation method based on Bayesian Principal Component Analysis. More recently, local least square techniques were applied in [6] to estimate missing values.

All the above imputation methods are quite similar in always selecting a fixed number of nearest neighbouring genes in order to estimate all the missing values in a target gene. The latter fact may lead to a somewhat random choice of candidate gene profiles for imputation. For instance, it may happen that the expression profiles of some candidate genes are rather distant from the target profile since these have only been included in the estimation list in order to reach the required fixed number of genes. In [19], Zhipeng *et al.* proposed an Iterated Local Least Squares imputation (ILLSimpute) method, which does not fix the number of nearest neighbouring genes, but considers instead all genes that are within a distance threshold to a target gene as coherent to it. The distance threshold in the ILLSimpute is set as δ times the average distance to the target gene and the ratio δ is learned using the dataset itself with pseudo missing values. We presented in [16,17] an imputation method, called Dynamic Time Warping imputation (DTWimpute), which also identifies for each gene profile with missing values a varying number of candidate expression profiles that exhibit at least minimum relative (preliminary defined) similarity in terms of some distance measure to the gene profile that best matches the profile of the target gene.

The usage of multiple imputation candidates, or so-called multiple imputation [8,11], is also known in the imputation literature. It has been applied for a first time to microrray data to evaluate the efficiency of the developed in [7] SKNN imputation algorithm. The implemented there imputation algorithm creates multiple plausible estimations for each missing observation. Subsequently, the mean of these estimations is used to fill in the missing value. Further, Nguyen *et al.* [9] proposed a multiple imputation method via ordinary least squares. Multiple estimation of each missing value is obtained by repeatedly regressing the target gene on each of the selected candidate genes. The final estimate of the missing value is again calculated as the mean (or weighted) average of the separate estimations.

An adequate choice of a distance measure used to select the candidate estimation profiles ensures good imputation accuracy and it is usually data dependent. Thus the KNNimpute algorithm, based on the Euclidean distance, was found to work well on non-time series data [15], while the added value of the Dynamic Time Warping (DTW) based techniques for missing value estimation in time series data has been demonstrated in [16,17].

We suggest in this work a novel Adaptive Multiple Imputation (AMimpute) algorithm for missing value estimation of gene expression data, which provides for a choice between two distance measures (Euclidean and DTW) to evaluate the similarity between two expression profiles and in this way, it can be applied to both non-time series and time series data. Additionally, the algorithm is adaptive since it uses a varying number of neighbouring genes for the estimation of each missing value in a gene. This is implemented by generating for each missing value a list of multiple candidate estimation values and then selecting the most suitable one according to some well-defined criteria, in order to replace the missing entry. Each candidate value for a given missing entry is derived from the set of nearest neighbouring gene profiles of the expression profile containing the missing value in question. These nearest neighbouring genes are selected according to preliminary defined degree of similarity to the estimated profile.

All the procedures used in this paper are implemented in Perl and C++ and the programs are available upon request.

2 Methods

2.1 Adaptive Multiple Imputation Algorithm

As already emphasized in the introduction, one of the important features of the imputation method proposed herein is the ability to use in the estimation process of each missing value a varying number of neighbouring gene profiles. The latter is implemented by generating for each missing value a list of candidate imputation values and then selecting the most suitable one, according to some well-defined criteria, in order to fill in the missing value. Each candidate value for a given missing value is derived from the set of nearest neighbouring gene profiles of the expression profile containing the missing value in question. These nearest neighbouring genes are selected according to preliminary defined degree of similarity to the estimated profile. This is realized by constructing an estimation list for each gene profile with missing values, which consists of genes with expression profiles all at a maximum R-relative (R is preliminary defined) distance from the best matching expression profile to the target profile (the one with missing values). These expression profiles are consequently used for missing value estimation. The estimation list construction algorithm has already been applied in [16,17].

The gene expression data generated from a set of microarray experiments are usually presented as a large matrix, with expression levels of genes ordered in rows and the experimental conditions in columns. Assume that a matrix G

of $m \times n$ $(m \gg n)$ contains the expression values of m genes measured in n experiments:

$$G = \begin{bmatrix} g_1 \\ \vdots \\ g_m \end{bmatrix} = \begin{bmatrix} g_{11} & \cdots & g_{1n} \\ \vdots & & \vdots \\ g_{m1} & \cdots & g_{mn} \end{bmatrix},$$

where the row $g_i = [g_{i1}, \ldots, g_{in}]$ represents the expression profile of the i-th gene.

Consider a set of M radiuses (relative degrees of similarity) $\{R_p \mid p = 1, \ldots, M\}$, where $R_p \in [1, 100]$, and an expression profile $g_i = [g_{i1}, \ldots, g_{in}]$ with some missing values. The proposed imputation algorithm uses this preliminary defined list of radiuses and generates a gene estimation list for each missing value in g_i and each radius R_p, consisting of genes with expression profiles at a maximum R_p-relative radius distance from the gene profile g_i. The obtained set of M estimation lists are further used to derive a list of M candidate imputation values, which on their turn are used to select the most suitable value to fill in the missing value. In the implemented C++ version of our AMimpute algorithm this most suitable value is the closest to the average of the two non-missing neighbours of the missing value. Of course, other more elaborated selection criteria may be applied.

Note also that two types of gene estimation lists are considered in the present implementation: 1) containing only complete (no missing values) gene profiles 2) complete and non-complete profiles, excluding only those gene profiles that have missing value at the same position with the gene profile to be imputed. Thus each missing value at position t in g_i will be imputed according to the following algorithm.

1. Construct an initial gene estimation list E_{it} either as:

   $$E_{it} = \{\text{all genes}\} \setminus \{\text{genes with missing value in experiment } t\}$$

   for expression matrices with high missing rate and large number of experiments or as:

   $$E_{it} = \{\text{all genes}\} \setminus \{\text{genes with missing values}\}$$

   otherwise, i.e. for expression matrices with low missing rate or small number of experiments.
2. Choose a distance measure to evaluate the similarity between the expression profiles, the DTW distance[1] for time series data or the Euclidean metric in all other cases.
3. Build a set of M gene estimation lists $\{E_{it_p} \mid p = 1, \ldots, M\}$, one for each radius R_p as follows:
   - Let $K = \#E_{it}$.
   - For $j = 1, \ldots, K$ calculate the distance d_{ik_j} (Euclidean or DTW distance) between gene i (the one with missing values) and gene $k_j \in E_{it}$.

[1] A more detailed description of the DTW algorithm can be found in the Appendix.

- Reorder E_{it} in such way that the genes are sorted in increasing order of their distances to gene i.
- Construct a new gene estimation list E_{it_p} by removing from the gene estimation list E_{it} all genes k_j, for which:

$$\frac{d_{ik_j} - d_{ik_1}}{d_{ik_j}} 100 \geq R_p.$$

  The final estimation list E_{it_p} contains only genes at a maximum R_p-relative distance (Euclidean or DTW) from gene k_1, which is the best matching gene to gene i with missing values.

4. Create a list of M candidate imputation values $\{g_{it_p} \mid p = 1, \ldots, M\}$, one for each estimation list E_{it_p}, for the missing value in experiment t. Calculate each imputation value as follows:

$$g_{it_p} = \sum_{j=1}^{K_p} w_{k_j} g_{k_j t},$$

   where w_{k_j} is the weight assigned to gene $k_j \in E_{it_p}$, expressing the degree of similarity between the expression profile of this gene to the expression profile of gene i, i.e. w_{k_j} is obtained by the formula:

$$w_{k_j} = \left(1 - \frac{d_{ik_j}}{\sum\limits_{j=1}^{K_p} d_{ik_j}} \right) / (K_p - 1),$$

   where $K_p = \#E_{it_p}$.
5. Select the value among $\{g_{it_p} \mid p = 1, \ldots, M\}$, which is the closest to the average of the two non-missing neighbours of the missing value $(g_{it-k} + g_{it+l})/2$. That value will be used to impute the missing entry at position t in g_i.

3 Results and Discussion

We have compared the efficiency of our AMimpute method with two other imputation methods: KNNimpute and DTWimpute. The latter have been specially selected for their various features (*e.g. fixed* versus *varying* number of gene profiles used for imputation; *complete* versus *all* profiles, except for the ones with missing value at the missing entry position, considered in the imputation process), which make them very suitable as benchmarking baseline for AMimpute.

The KNNimpute method, developed by Troyanskaya *et al.* [15], is widely used in microarray data analysis. It selects a preliminary fixed number (K-nearest) of neighbouring genes from the microarray matrix excluding any gene profiles that have missing value at the same position with the gene to be imputed. The

Euclidean distance is used to estimate the similarity of neighbouring genes. The missing value is finally imputed with the weighted average of the corresponding column of the K-nearest genes. Note that the Euclidean distance is known to be sensitive to outliers, which can often be present in microarray data. Therefore, Troyanskaya *et al.* [15] proposed to partially overcome this shortcoming by always applying a log-transformation on the microarray data.

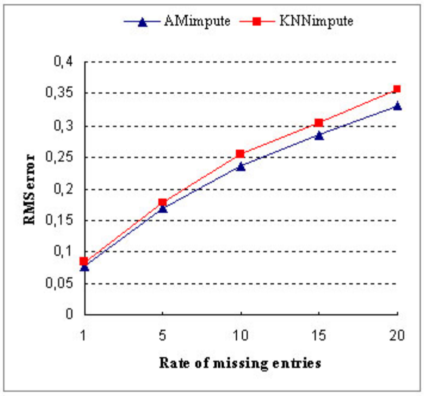
Most of the existing missing value imputation approaches are not particularly suitable for time series expression profiles. The DTWimpute algorithm has specially been designed for estimation of missing values in gene expression time series data [16,17]. The algorithm utilizes the DTW (Dynamic Time Warping) distance in order to measure the similarity between time expression profiles and subsequently, for each gene profile with missing values a varying number (based on the degree of similarity) of complete expression profiles (no missing values) is identified for imputation. The DTW distance is known to produce a more intuitive similarity measure for time series data since it allows for similar shapes to match even if they are out of phase in the time axis.

Four microarray datasets with different missing rates have been used in the evaluation process. The first three datasets are from Rustici *et al.* [12], used for examining the global cell-cycle control of gene expression in fission yeast *Schizosaccharomyces pombe* and are time series data. The whole study includes 8 independent time-course experiments synchronized respectively by elutriation, cdc25 block-release and a combination of both methods. We have included in our test corpus the elutriation datasets, i.e. the following 3 different time series data sets have been used: *elutriation1*, *elutriation2* and *elutriation3*. These data sets contain expression levels of 5038 to 5120 genes measured in 20 time points. The fourth dataset is non-time series and is a study of response to environmental changes in yeast [3]. It contains 6152 genes monitored in 173 experiments. Note that the above datasets have been used in some previous studies, *e.g.*, the time series datasets have been included in the test corpus of DTWimpute [16,17], while the non-time series set has been used in the studies of KNNimpute [15] and SKNNimpute methods [7].

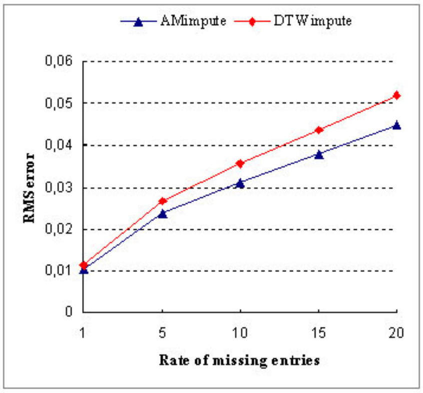
A special test data corpus has been created as follows. Initially, all rows containing missing values have been removed from each of the four original gene expression datasets. These transformed original datasets are further referred to as the *complete* (no missing values) data matrices. Consequently, five new test sets have been derived from each complete data matrix by deleting randomly 1%, 5%, 10%, 15% and 20% respectively of the original data.

The accuracy of estimation has been calculated as the Root Mean Squared (RMS) difference between the imputed matrix and the original matrix, divided by the average data value in the complete dataset:

$$\mathrm{RMS} = \frac{\sqrt{\sum_{ij} \left(R_{ij} - I_{ij}\right)^2 / nm}}{\sum_{ij} \left|R_{ij}\right| / nm},$$



(a)



(b)

Fig. 1. (a) Comparison of the AMimpute and KNNimpute imputation performance on non-time series data; (b) Comparison of the AMimpute and DTWimpute imputation performance on time series data

where R_{ij} is the value in the complete matrix, I_{ij} is the value in the imputed matrix, m is the number of genes and n is the number of experiments.

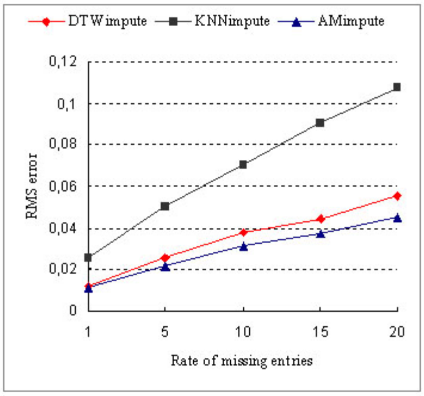
The RMS error rate of the AMimpute algorithm has been benchmarked against the KNNimpute algorithm on the non-time series data and against the DTWimpute on the time series datasets. Fig. 1a depicts the AMimpute and KNNimpute RMS curves for the non-time series test data using the Euclidean distance to measure the similarity between the expression profiles. Clearly, the AM imputation method attains lower RMS figures for the whole range of missing rates. The RMS values of the AMimpute have been obtained for a list of 25 radiuses R (all integer values from 1 to 25). The gene estimation lists have been composed of gene profiles that have a value at the missing value position and are at a maximum R-relative radius Euclidean distance from the target profile. The RMS values of the KNNimpute have been generated for $K = 10$, which produced the lowest RMS error in almost every dataset tested in [15].

Fig. 1b compares the imputation performance of the AMimpute and DTWimpute algorithms on the time series data using DTW distance to measure the similarity between the expression profiles. Analogously to Fig. 1a, the AMimpute RMS curves have been generated using 25 candidate imputation values (for integer radius R from 1 to 25) for the estimation of each missing value. The estimation lists have been constructed of gene profiles that have a value at the missing value position and are at a maximum R-relative radius DTW distance from the target profile. The RMS values of the DTWimpute have been obtained for radius 1, which guarantees, as demonstrated in [17], overall low RMS error rates. AMimpute is clearly superior over the whole range of tested missing rates.

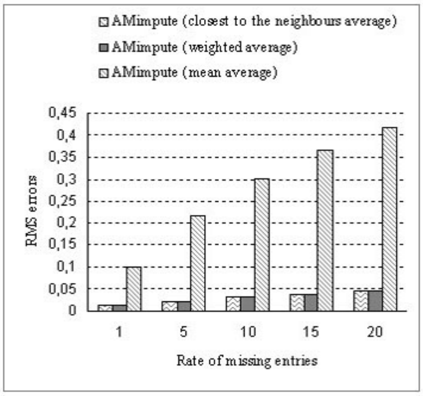
Recall that the DTWimpute algorithm selects neighbours for the imputation of a missing value only among genes with complete expression profiles. However, the pool of complete expression profiles may be drastically reduced as a result of

Table 1. The RMS error rate of AMimpute algorithm using two different initial estimation lists. The DTW distance was used for the time series data and the Euclidean distance for the non-time series.

RMS	**time series**		**non-time series**	
mis. rate	complete profiles	complete & non-complete profiles	complete profiles	complete & non-complete profiles
1%	0.0106	0.0102	0.083	0.076
5%	0.0249	0.0236	0.493	0.169
10%	0.0334	0.0311	0.675	0.235
15%	0.0408	0.0378	0.816	0.284
20%	0.0495	0.0448	0.947	0.332



(a) AMimpute, DTWimpute and KNNimpute RMS curves.



(b) AMimpute RMS figures for three different final imputation values.

Fig. 2. The different imputation performances are evaluated on the *elutriation1* time series dataset employing the DTW distance

increased missing rate. In this situation, the imputation method will evidently select less similar neihghbours for imputation. The AMimpute allows in the gene estimation list any gene profiles that have a value at the missing value position. Thus the imputation accuracy for data with high level of non-complete gene profiles and large number of experiments, which is the case for our non-time series test data (over 80% of rows have at least one missing value), can considerably be improved. Table 1 illustrates the impact on the RMS error rate of the choice of a particular pool (complete/non-complete) of expression profiles for estimation.

As already mentioned in the introduction, the choice of a distance measure to be used in the selection process of the candidate estimation genes is of a crucial importance for the estimation performance of the applied imputation algorithm. For instance, we have generated the RMS curves of the three tested

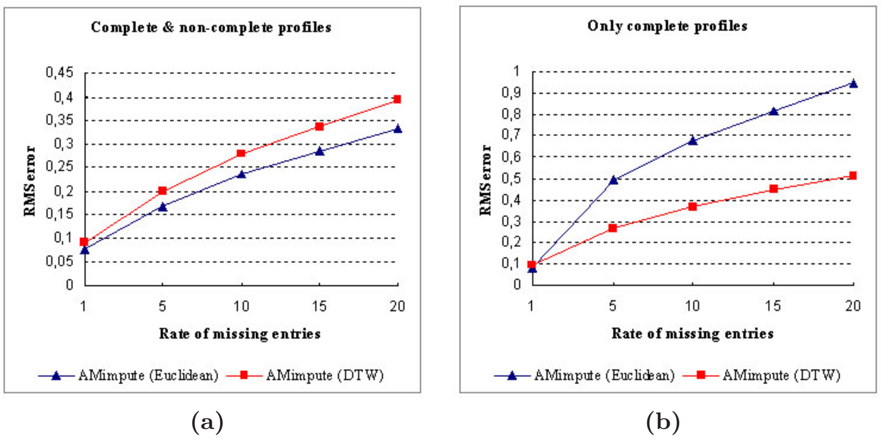
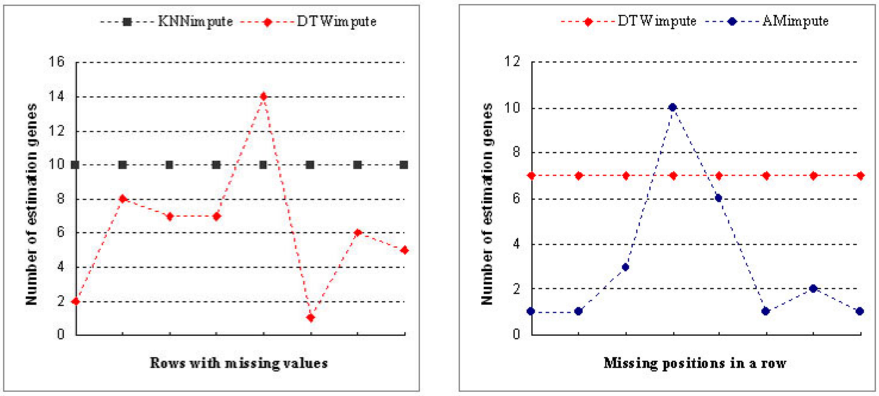


Fig. 3. AMimpute imputation performance on non-time series data applying two different distances (Euclidean versus DTW distance): (a) Complete & non-complete profiles; (b) Only complete profiles

algorithms, AMimpute, DTWimpute and KNNimpute, for the *elutriation1* time series datasets. These are depicted in Fig. 2a. Recall that KNNimpute uses the Euclidean distance to measure the similarity between the expression profiles, whereas the DTW distance is implemented in the DTWimpute and AMimpute algorithms. Consequently, as it can be seen in Fig. 2a, the difference in RMS performance between KNNimpute and DTWimpute is significantly higher than the one between DTWimpute and AMimpute, respectively.

The imputation performance of AMimpute has further been tested on the non-time series test set using two different distance measures, Euclidean versus DTW, (see Fig. 3). The RMS values reported in Fig. 3a have been obtained for gene estimation lists composed of gene profiles that have a value at the missing value position, while only complete expression profiles have been used for the RMS results shown in Fig. 3b. In the first case (Fig. 3a), the AM imputation method using the Euclidean distance as a measure of similarity attains, as expected, lower RMS figures for the whole range of missing rates. However, in the case of complete expression profiles (Fig. 3b), the RMS results obtained for the DTW distance are strangely superior over the whole range of tested missing rates. One possible explanation of this phenomenon may be the better alignment, which the DTW algorithm usually produces when the complete expression profiles are involved.

The KNNimpute uses a fixed number of neighbouring genes for the estimation of each missing observation in the whole microarray matrix, while DTWimpute uses in general a varying number of profiles for the estimation of the different missing values, except for the ones belonging to the same expression profile. In the latter case, i.e. for missing values situated in the same expression profile, the DTWimpute algorithm identifies a common set of expression profiles to be used in the imputation process of each of these missing entries. The AM imputation method goes further than that by using a varying number of neighbouring gene



(a) KNNimpute versus DTWimpute. **(b)** DTWimpute versus AMimpute.

Fig. 4. Comparison of the number of estimation genes used in the imputation process

profiles in the estimation process of each missing value in the target gene. These essential differences between the three tested algorithms are illustrated in Fig. 4. A random set of expression profiles with missing values have been considered and the number of genes used for the estimation of each missing value in this set has been recorded for KNNimpute and DTWimpute as shown in Fig. 4a. The same procedure has been performed for the number of profiles used by the DTWimpute and AMimpute algorithms to estimate the missing values belonging to a selected single profile. Fig. 4b compares the number of estimation genes used by the DTWimpute and AMimpute for each missing value in the profile. For instance, the DTWimpute uses an estimation list of 7 neighbouring genes for the imputation of each missing observation in the studied profile, while the number of genes in the estimation lists of AMimpute vary from 1 to 10 (see Fig. 4b).

Nguyen *et al.* [9] proposed an imputation method, which generates multiple estimates for each missing observation via ordinary least squares regression. The final imputation value is obtained as the weighted average (or mean) of these multiple estimations. The latter prompts us to study how the estimation performance of our AM imputation algorithm depends on the way the final imputation value is derived. The AMimpute imputation performance has been evaluated on the *elutriation1* time series dataset comparing three different possible calculations for the final imputation value: 1) the closest to the average of the two non-missing neighbours of the missing value, 2) the weighted mean of the multiple candidate imputation values, and 3) the mean of the multiple candidate imputation values (see Fig. 2b). In the case of weighted mean, the weights are based on the radiuses (relative degrees of similarity) used to select the nearest neighbouring genes, i.e.

$$w_p = \frac{R_M - R_p + 1}{\sum\limits_{j=1}^{M} R_j},$$

where w_p $(p = 1, \ldots, M)$ is the weight assigned to the estimate obtained for radius R_p and R_M is the highest radius used in the concrete estimation process. As it can be seen in Fig 2b, the RMS values obtained for the first two approaches are very comparable over the whole range of the tested missing rates. However, the RMS results are significantly worse when the final estimate is calculated as the mean of the single estimates.

As was shown in Table 1, involving non-complete profiles in the estimation process significantly improves the imputation accuracy of the AM imputation algorithm on data with high level of non-complete gene profiles and large number of experiments. This improvement however, increases the computational complexity of the algorithm since an initial gene estimation list is constructed individually for each missing value, while in case of complete expression profiles such a list is common for all estimations. For instance, the C++ implementation of AMimpute employing the Euclidean distance takes on average 4.5 min for performing missing data estimation on a matrix of a size 755×173 with 20% missing entities on a standard PC (with Pentium 4 processor) in the former case (non-complete profiles) and about 1 sec in the latter one (complete profiles). The computational complexity of AMimpute when Euclidean distance is used is comparable with this of the KNNimpute and it is approximately $O(m^2n)$ for a matrix with m genes and n experiments. It is to expect that KNNimpute is slightly faster than AMimpute since it uses a fixed number of neighbours for the imputation of each missing value. Thus in case of 20% missing values, the KNN execution time is about 3 min for the above matrix and $K = 10$. However, the computational performance of the KNNimpute is affected by the number of neighbouring genes used and it may increase a few times for a higher values of K. In case the DTW distance is used to measure the similarity between the expression profiles, the complexity of AMimpute is in the range of $O(m^2n^2)$. In fact, when complete expression profiles are involved in the estimation process its execution time will be comparable with this of the DTWimpute. AMimpute takes on average 9 sec for performing a missing data estimation using only complete expression profiles on a time series data matrix of a size 3500×20 with 20% missing entries, which is almost equal to the DTWimpute execution time of 8 sec for the same matrix.

In our opinion, the AMimpute performance is very satisfactory, considering that in most cases the missing value imputation needs to be performed once, usually off-line, and then the complete data matrix is available for further processing and analysis.

4 Conclusion

We have developed a novel imputation algorithm (AMimpute) for estimation of missing values in gene expression data, which offers more robust and more accurate missing value estimation by using a varying number of neighbouring genes in the estimation process of each missing value. The proposed algorithm has been tested and evaluated on both non-time series and time series data.

The efficiency of the developed adaptive multiple imputation method has been compared with two other imputation methods: KNNimpute and DTWimpute by applying them to microarray datasets with different missing rates. It has been demonstrated that the performance of the AMimpute method is better than the conventional KNN imputation approach for non-time series data and than the DTWimpute algorithm designed specially for time series data.

References

1. Aach, J., Church, G.M.: Aligning gene expression time series with time warping algorithms. Bioinformatics 17, 495–508 (2001)
2. Criel, J., Tsiporkova, E.: Gene Time Expression Warper: A tool for alignment, template matching and visualization of gene expression time series. Bioinformatics 22(2), 251–252 (2006)
3. Gasch, A.P., Spellman, P.T., Kao, C.M., Carmel-Harel, O., Eisen, M.B., Storz, G., Botstein, D., Brown, P.O.: Genomic expression programs in the response of yeast cells to environmental changes. Molecular Biology of the Cell 11, 4241–4257 (2000)
4. Hermans, F., Tsiporkova, E.: Merging microarray cell synchronization experiments through curve alignment. Bioinformatics 23, e64–e70 (2007)
5. Hastie, T., Tibshirani, R., Sherlock, G., Eisen, M., Brown, P., Botsein, D.: Imputing missing data for gene expression arrays. Technical report, Division of Biostatistics, Standford University (1999)
6. Kim, H., Golub, G.H., Park, H.: Missing value estimation for DNA microarray gene expression data: Local least squares imputation. Bioinformatics 21, 187–198 (2005)
7. Kim, K., Kim, B.J., Yi, G.S.: Reuse of imputed data in microarray analysis increases imputation efficiency. BMC Bioinformatics 5, 160 (2004)
8. Little, R., Rubin, D.: Statistical analysis with missing data. Wiley, New York (1987)
9. Nguyen, D., Wang, N., Carroll, R.: Evaluation of missing value estimation for microarray data. Journal of Data Science 2, 347–370 (2004)
10. Oba, S., Sato, M., Takemasa, I., Monden, M., Matsubara, K., Ishii, S.: A Bayesian missing value estimation method for gene expression profile data. Bioinformatics 19, 2088–2096 (2003)
11. Rubin, D.B.: Multiple imputation for nonresponse in surveys. John Wiley & Sons, Inc., New York (1987)
12. Rustici, G., Mata, J., Kivinen, K., Lio, P., Penkett, C.J., Burns, G., Hayles, J., Brazma, A., Nurse, P., Bähler, J.: Periodic gene expression program of the fission yeast cell cycle. Nat. Genet.
13. Sakoe, H., Chiba, S.: Dynamic programming algorithm optimization for spoken word recognition. IEEE Trans. on Acoust., Speech, and Signal Proc. ASSP 26, 43–49 (1978)
14. Sankoff, D., Kruskal, J.: Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison. AddisonWesley, Reading Mass. (1983)
15. Troyanskaya, O., Cantor, M., Sherlock, G., Brown, P., Hastie, T., Tibshirani, R., Botstein, D., Altman, R.B.: Missing value estimation methods for DNA microarrays. Bioinformatics 17(6), 520–525 (2001)
16. Tsiporkova, E., Boeva, V.: Dynamic time warping techniques for missing value estimation in gene expression time series. In: Proc. of the 15th Dutch-Belgium Conference on Machine Learning, pp. 97–104 (2006)

17. Tsiporkova, E., Boeva, V.: Two-pass imputation algorithm for missing value estimation in gene expression time series. Journal of Bioinformatics and Computational Biology 5(5), 1005–1022 (2007)
18. http://www.tu-plovdiv.bg/Container/bi/DTWimpute
19. Zhipeng, C., Maysam, H., Guohui, L.: Iterated local least squares microarray missing value imputation. Journal of Bioinformatics and Computational Biology 4(5), 935–957 (2006)

5 Appendix

5.1 Dynamic Time Warping Algorithm

The Dynamic Time Warping (DTW) alignment algorithm was developed originally for speech recognition [13] and it aims at aligning two sequences of feature vectors by warping the time axis iteratively until an optimal match (according to a suitable metrics) between the two sequences is found.

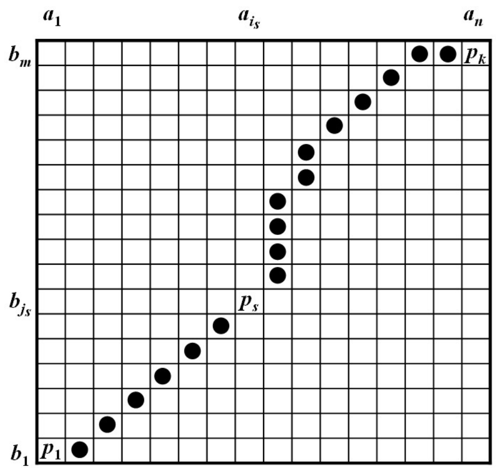


Fig. 5. The DTW grid with the optimal warping path through it

Let us consider two sequences of feature vectors:

$$A = [a_1, a_2, \ldots, a_n]$$
$$B = [b_1, b_2, \ldots, b_m].$$

The two sequences can be arranged on the sides of a grid, with one on the top and the other on the left hand side, see Fig. 5. Both sequences start on the bottom left of the grid. Inside each cell a distance measure can be placed, comparing the corresponding elements of the two sequences. To find the best match or alignment between these two sequences one needs to find a path through the grid

$$P = p_1, \ldots, p_s, \ldots, p_k,$$

where $p_s = (i_s, j_s)$ and P, referred to as the warping function, minimizes the total distance between A and B (Fig. 5). Thus the procedure for finding the best alignment between A and B involves finding all possible routes through the grid and for each one compute the overall distance, which is defined as the sum of the distances between the individual elements on the warping path. Consequently, the final DTW distance between A and B is the minimum overall distance over all possible warping paths:

$$\mathrm{dtw}(A, B) = \frac{1}{n+m} \min_P \left(\sum_{s=1}^{k} d(i_s, j_s) \right).$$

It is apparent that for any pair of considerably long sequences the number of possible paths through the grid will be very large. However, the power of the DTW algorithm resides in the fact that instead of finding all possible routes through the grid, the DTW algorithm makes use of dynamic programming and works by keeping track of the cost of the best path at each point in the grid. Due to its flexibility, DTW has been widely used in many scientific disciplines including several computational biology studies [1,2,4]. A detail explanation of DTW algorithm can be found in [4,13,14].

Topological Metrics in Blast Data Mining: Plasmid and Nitrogen-Fixing Proteins Case Studies

Pietro Lió[1], Matteo Brilli[2,3], and Renato Fani[2]

[1] Computer Laboratory, University of Cambridge, Cambridge CB3 0FD, United Kingdom
pl219@cam.ac.uk

[2] Department of Animal Biology and Genetics, University of Florence, via Romana 17-19, 50125 Firenze, Italy
renato.fani@unifi.it

[3] Biometrie et Biologie Evolutive, UMR CNRS 5558, Universite Lyon 1, 43, bvd du 11 novembre 69622 Villeurbanne Cedex, France
brilli@biomserv.univ-lyon1.fr

Abstract. Over the past years, a number of metrics have been introduced to characterize the topology of complex networks. We use these methodologies to analyze networks obtained through Blast data mining. The algorithm we present consists of the following steps: 1- encode results of Blast searches as a distance matrix of e-values; 2- perform entropy-controlled clustering analysis to identify the communities; 3- statistical analysis of the resulting network, 4- gene ontology and data mining in sequence databases to infer the function of the identified clusters. We report on the analysis of two data sets; the first is formed by over 3300 plasmid encoded proteins and the second comprises over 4200 sequences related to nitrogen fixation proteins. In the first case we observed strong selective pressures for horizontal transfer and maintenance of genes encoding proteins for resistance to antibiotics, plasmid stability and conjugal transfer. Nitrogen fixation proteins can be divided on the basis of our results into three different groups: proteins with no paralogs in any of the genomes considered, proteins with paralogs belonging to different metabolic processes (O–paralogs) and proteins with paralogs in other and the same metabolic processes (I/O–paralogs).

1 Introduction

The determination of the statistical relationships among biological sequences is one of the most important tasks in bioinformatics. Most of the work in the past has focused on phylogenetic methods that describe the non independence of the sequences because of the common ancestors (the nodes of the phylogenetic tree). An alternative meaningful approach is to consider the networks of relationships found in a data mining approach performed using Blast-family [1] or PatternHunter algorithms [2]. This approach is justified by the lateral gene transfer events in prokaryotes, between nuclear and mitochondrial genomes, and between host and parasites or symbionts. Models of the relationships between sequences are defined on a connectivity network, where nodes are proteins and edges represent evolutionary connections. Current research in complex network analysis generally have four different focuses: (i) Characterization of the geometry/structural properties of large networks from the real world [3, 4]; (ii) developing of new models (growing graphs) [4]; (iii) dynamical processes on networks, such as epidemics [5]; (iv) adaptive networks, i.e. changes in topologies due to node activity [6]. Here we will focus on the first point. When analyzing complex networks, the class of topology needs to be compared with regular lattices, small worlds or random networks by making use of several measures that describe the network's topology. One such quantity is the connectivity. In order to visualize the influence of connectivity on network topology, let us introduce the *connectivity* distribution $P(k)$, which is proportional to the number of sites with connectivity k. The probability distribution of a regular lattice is extremely peaked around the lattice's *coordination number* (the number of vertices directly connected in the lattice), being a δ-function; in a random graph it follows a Poisson distribution, centered around the mean connectivity $\langle k \rangle$. Both of those choices are characterized by a well defined value of the mean connectivity $\langle k \rangle$, and small variance $\langle k^2 \rangle - \langle k \rangle^2$. As shown by Watts and Strogatz [7], the simple rewiring of a small fraction of links in an otherwise regular lattice results in a sudden lowering of the diameter of the graph, without affecting the average connectivity or the degree of clustering. This *small world* effect manifests itself in a dramatic shortage of the distance between any two nodes, almost without affecting the local perception of the network of contacts. A meaningful measure is the degree of *clustering* of the network, which can be defined as the number of neighbors of a given node which share an edge. In a regular lattice the clustering coefficient is high, while in a random graph it is vanishingly low. Finally, one is interested in the average length of the minimum path that connects any two vertices, called the *diameter* of the graph. This is large in a regular lattice, and low in a random graph. The study of biological networks has shown major differences from regular and random graphs because the degree distribution often follows a power-law (i.e. $P(k) \simeq k^{-\gamma}$, with $\gamma \simeq 2$). This distribution is characterized by a relatively large number of highly connected nodes (called *hubs*). Moreover, such distributions have a diverging second moment $\langle k^2 \rangle$ for $\gamma \leq 3$ and a diverging average connectivity $\langle k \rangle$ for $\gamma \leq 2$. An important class of biological networks show scale-free connectivity properties [8]. A simple model to generate networks with this property is based on the *preferential attachment* and can be exemplified in the following way. We start with a ring of K nodes, and we add the other $N - K$ nodes by choosing, for each of them, K connected nodes $j_n, n = 1, 2, \ldots, K$, with probability $k(j_n)/\sum_{l=1}^{N} k(l)$ (preferential attachment). The new node is added to the *inputs* of the chosen nodes. We also choose another node at random and add it to the list of input nodes of the new node. This process simulates the growing of a sequence network in which a new node (a paralogous gene) is duplicated from another one. Networks have been used to study metabolic pathways [9], signal transduction [10], transcriptional regulation [11] and other cellular processes; all of them showed *scale-free* properties [8].

Inference on gene functions in the analysis of novel genomes generally relies on local alignment tools, such as Blast [1]. These programs allow searching for homologs in sequence databases and assigning functions on the basis of sequence similarity to known proteins (*functional transfer*). Local alignment tools have been implemented in several automated strategies for the necessary post–processing of the outputs [12]. The output of a similarity search may be encoded as an adjacency matrix where each element s_{ij} is a measure of the similarity shared by proteins i and j and can be used as input for a clustering algorithm to infer communities of related sequences. This strategy has been used to construct the Clusters of Orthologous Groups (COG, [13]) and OrthoMCL databases [14]. Advanced clustering techniques have been implemented to better identify clusters of related proteins in large scale similarity searches; tribeMCL [15] and orthoMCL [14] both implement a relatively recent procedure, the Markov Clustering algorithm (MCL, [16]) to partition proteins into families. The main advantage of the MCL over other clustering methods is that there is no need to specify the number of clusters.

We describe a framework combining complex network theory, MCL clustering and standard genomics tools to make inferences on protein function and gene family classification; we tested it on two case studies. The first one comprises over 3300 plasmid encoded proteins from three enterobacterial genera. Plasmids are shared resources of a large community of bacteria; from this viewpoint, bacteria are part of a super-organism and studying the ”omics” of plasmids corresponds to investigating the system biology of this super-organism. Despite the huge amount of plasmid sequences in databases there are few papers reported results from large scale comparative analysis e.g. [17] and none of them considered all sequences coming from entire taxonomic groups. The network approach is particularly suited in this case because plasmids are easily transferred between different bacteria [18] and the process is evolutionarily important.

Our second case study comprises over 4300 proteins related to enzymes of nitrogen fixation. This process is the biological reduction of atmospheric dinitrogen to ammonium and it is performed by some prokaryotes. Nitrogen is a fundamental constituent of many biological molecules, first of all proteins and nucleic acids, thus the survival of organisms unable to fix dinitrogen (i.e. eukaryotes, most prokaryotes) depends on the environmental availability of reduced nitrogenous compounds. Several nitrogen fixation proteins are part of paralogous families [19] and are related to enzymes involved in the biosynthesis of photosynthetic pigments (i.e. Bch proteins).

2 Methods

We devise an effective way to (i) establish the relationships between sequences by means of a similarity matrix recording local alignment scores, (ii) reduce its dimension by a clustering algorithm and (iii) perform statistical analysis on the network centrality measures such as shortest path length, diameter, betweenness, and assortativeness. In particular the dimensionality reduction is obtained through the use of an entropy controlled clustering method. Our approach consists of four basic steps (Figure 1):

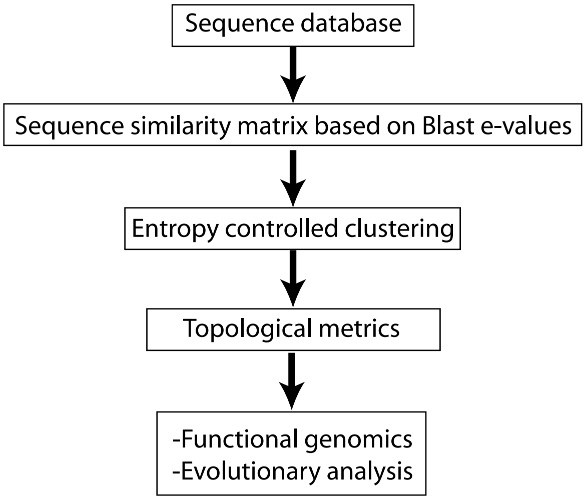

Sequence database
Sequence similarity matrix based on Blast e-values
Entropy controlled clustering
Topological metrics
-Functional genomics
-Evolutionary analysis

Fig. 1. Summary of the strategy adopted in this work

1. We build a similarity matrix based on alignment scores from Blast. The s_{ij} element of the matrix is the normalized score for the local alignment between sequence i and j. Raw blast scores are not suitable because they are the sum of substitution and gap scores taken from a given substitution matrix (e.g. Blosum, Dayhoff) and they are consequently dependent on the alignment length; this causes the risk of over-scoring long alignments; we normalize the matrix row by row on the basis of the diagonal element s_{max} (the score of a self aligned protein); in this way each element becomes $s' = \frac{s_{ij}}{s_{max}}$.

   After this transformation, the matrix encodes a network of normalized global sequence relationships: nodes are proteins with edges between homologous proteins. Edge values can also be weighted with distance matrices obtained by comparing housekeeping proteins, allowing to take into account the evolutionary relationships between organisms; this might help the identification of groups of proteins with a degree of homology not expected by the phylogenetic distance of the source organisms i.e. horizontally transferred proteins.
2. The following step is the clustering of the network: we use a clustering algorithm that does not need specifying the number of clusters in advance. Although there is now a wide range of clustering algorithms, only some of them are able to handle a network with the complete and weighted graph properties. One of these is the MCL [16] that we describe more in detail. The input is a stochastic matrix where each element is the probability of a

transition between adjacent nodes. The matrix is processed by iterative rounds of *expansion* (raising matrix $\mathbf{M}$ to the k^{th} power) and *inflation* (it takes the r^{th} power m_{ij}^r of every element using Hadamard exponentiation, where r is the granularity parameter of the clustering). Expansions correspond to computing the probabilities of random walks of higher lengths through the graph, while inflation promotes and demotes the probabilities of paths in the graph. We incorporated the strategy of Gfeller et al., [23] which allows detecting unstable nodes and comparing results obtained with different granularity (inflation) parameters. In this algorithm, the original matrix is modified by adding a user-defined amount of noise to its values. The noise is homogeneously distributed between $-\sigma w_{ij}$ and σw_{ij} where w_{ij} is the edge weight and σ the user-defined noise parameter in the range $[0, 1]$. The noise is added randomly to edges and the MCL clustering is performed on many noisy realizations of the matrix. Finally the program outputs a matrix storing P_{ij} values, corresponding to the fraction of times nodes i and j have been grouped together. Unstable nodes are identified as those having edges with less than a threshold value that can be specified by the user (i.e. the θ). All edges with a value below θ are then pruned and cluster structure is recorded. This matrix is used to calculate the *clustering entropy* measure developed by [23]: $S = -1/L \sum_{ij}[P_{ij} log_2 P_{ij} + (1 - P_{ij}) log_2 (1 - P_{ij})]$, where the sum is over all edges, normalized by the total number of edges, L. The clustering entropy allows identifying the best clustering after several repetitions with different parameters.

3. The program computes several topological metrics on the clusters, such as node degree distribution, assortativity, shortest path lengths (ASPL), diameter, betweenness and centrality. Definitions of these estimators can be found in [3,8] and references therein.
4. The final step is a data mining procedure in functional databases to characterize the function(s) of the proteins of each cluster.

3 Results and Discussion

3.1 Plasmids Network

The choice of the plasmid dataset relies on two main reasons: they represent good study models for network analysis because they can be transferred between different microrganisms and they often have biomedical importance since they harbor resistance genes and virulence associated factors. The plasmid dataset is composed by 3393 proteins coded for by 39 plasmids from the enterobacteria *Escherichia*, *Shigella* and *Salmonella*. The graph resulting from an all-against-all comparison was clustered with the MCL ($r = 2.2$). The original network showed a power-law distribution of the degree $\gamma = 1.32$ (see Table 1 for comparison with coefficients for other biological networks). We focused our attention to the functions of the proteins belonging to highly populated clusters because they might reflect the selective pressures acting in the environment. Clusters with the highest average degree mainly consist of transposases, IS-related sequences

Table 1. The γ parameter inferred for several types of networks. PI: Protein Interaction.

Network Type	**Estimated γ**	**Reference**
Yeast paralogs PI divergence (n=274)	1.38	[25]
Yeast PI (data from [26])	2.80	[27]
Yeast PI (core data from [28])	2.2	[29]
C. elegans PI (data from [30])	1.8	[31]
Metabolic network *E. nidulans*	2.2	[29]
Protein domains family size (*T. maritima*)	3	[32]
Protein domains family size (*C. elegans*)	1.9	[32]
Plasmid (γ–proteobacteria; n=3393)	1.32	This work
Nitrogen fixation proteins (n=4299)	2.3	This work

and transposons; they are more than 1400 proteins and represent the hubs of the network, being widely distributed and are often present in several copies for each plasmid.The biological significance of this finding is still unclear, if we consider that plasmids coming from other species generally encode few transposases (data not shown).

The analysis of the other clusters (showing a lower average degree) revealed that *Escherichia*, *Salmonella* and *Shigella* share genes coding for (i) the anti–restriction protein KlcA, involved in the broad-host range of IncP plasmids [33]; (ii) the RNA chaperone FinO, related to repression of sex pilus formation; and (iii) the CcdB protein, involved in plasmid stability by killing bacteria that have lost the plasmid [35].

It is worth noticing that recent works suggest that *Shigella* spp. are *Escherichia coli* strains [36]. However, our data are in disagreement with this finding because they revealed a higher affinity between *E. coli* and *Salmonella*. Indeed, several proteins are shared by only *E. coli* and *Salmonella*, such as the already known determinants for antibiotic resistance (TetA and TetR), involved in tetracycline resistance [37], beta-lactamases [38], proteins conferring resistance to amino glycosides (e.g.: AadA) and sulphonamides (e.g.: DHPT synthase); this defence repertoire is almost entirely missing in *Shigella* plasmids. Moreover, sex pilus–related proteins (i.e. Tra and Trb) revealed the same intriguing occurrence pattern: 21 out of 22 different clusters of Tra proteins included those coming from *E. coli* and *Salmonella* spp. Only the TraDI (DNA transport) and TraX (pilin acetylation) clusters include the three genera. The proteins TraP, TrbA and TrbJ are harbored only by *E. coli* plasmids. Similarly, there are 5 different Trb clusters, but *Shigella* has only a TrbH homologous. These data fully agree with recent horizontal transfers of plasmid genes between enteroinvasive *E. coli* and *Salmonella*. On the other side, *Shigella* spp. have a specific set of pathogenesis associated genes, including some of the proteins of the type III secretion system (TTS), which is formed by (i) a TTS apparatus spanning the bacterial envelope, (ii) translocator proteins forming a pore in the host cell membrane (the translocon) and (iii) specific transcription factors (among the three categories: Mxi, Spa, Ipa, Ipg and Osp proteins).

3.2 Nitrogen Fixation Network

The ability to fix nitrogen relies on the activity of a set of nitrogen fixation (Nif) proteins, which have been particularly studied in the γ–proteobacterium *Klebsiella pneumoniae*. In this organism 21 *nif* genes have been identified; we used their aminoacid sequence to retrieve homologous proteins related to those of nitrogen fixation using the Blastp program on 55 completely sequenced genomes comprising all diazotrophs species and a representative number of organisms from the other available taxonomic groups. All hits with a Blast e-value below 0.0001 were retrieved and a non-redundant dataset of over 4200 proteins was then constructed and used as input for our algorithm. Data obtained are shown in Figure 2 and can be summarized as follows:

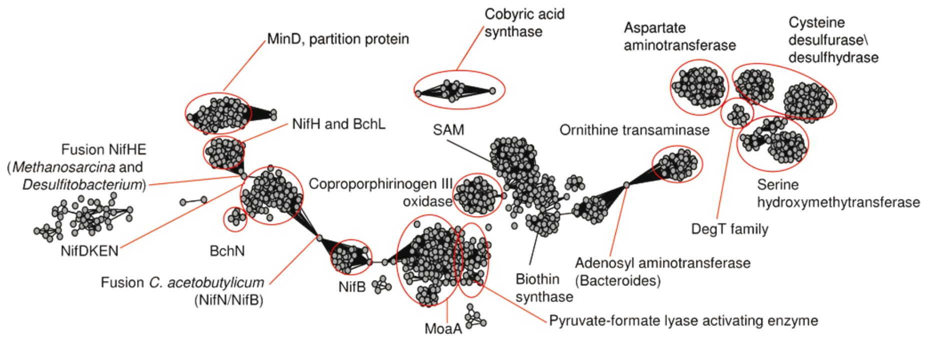

Cobyric acid synthase
MinD, partition protein
Aspartate aminotransferase
Cysteine desulfurase\ desulfhydrase
NifH and BchL
SAM
Ornithine transaminase
Fusion NifHE (*Methanosarcina* and *Desulfitobacterium*)
Coproporphirinogen III oxidase
Serine hydroxymethytransferase
DegT family
NifDKEN
BchN
Adenosyl aminotransferase (Bacteroides)
Fusion *C. acetobutylicum* (NifN/NifB)
NifB
Biothin synthase
MoaA
Pyruvate-formate lyase activating enzyme

Fig. 2. Network of Nif proteins

1. Four Nif proteins, NifW (NifO) (present in 12 genomes), NifT (FixU) (in 13 genomes), NifQ (in 11 genomes) do not have any paralog. Interestingly, these sequences are also missing from about half the diazotroph genomes we have analyzed (30).
2. Eight Nif proteins (NifA, F, H, J, L, M, S, U) are related to proteins involved in other metabolic pathways (Out– or O–paralogs). NifS is related to a number of proteins for amino acid and carbon metabolisms. NifJ, a multidomain Pyruvate:ferredoxin (flavodoxin) oxidoreductase, is part of a large multigene family whose representatives belong to distinct metabolic processes. However it is possible that NifJ is required for nitrogen fixation only in some diazotrophs (e.g. in *Erwinia carotovora*), because orthologs are not easily identifiable in several species (data not shown and [39]). Several proteins involved in Fe–Mo nitrogenase's cofactor biosynthesis have paralogs in other similar processes, suggesting an ancestral interconnection between them.
3. Eight Nif proteins share a significant degree of sequence similarity with other proteins involved in nitrogen fixation and other metabolic routes (I/O–paralogs). This group can be further split into two different clusters, the first one including NifD, K, E, N, and the second NifB, X, Y, V. As expected

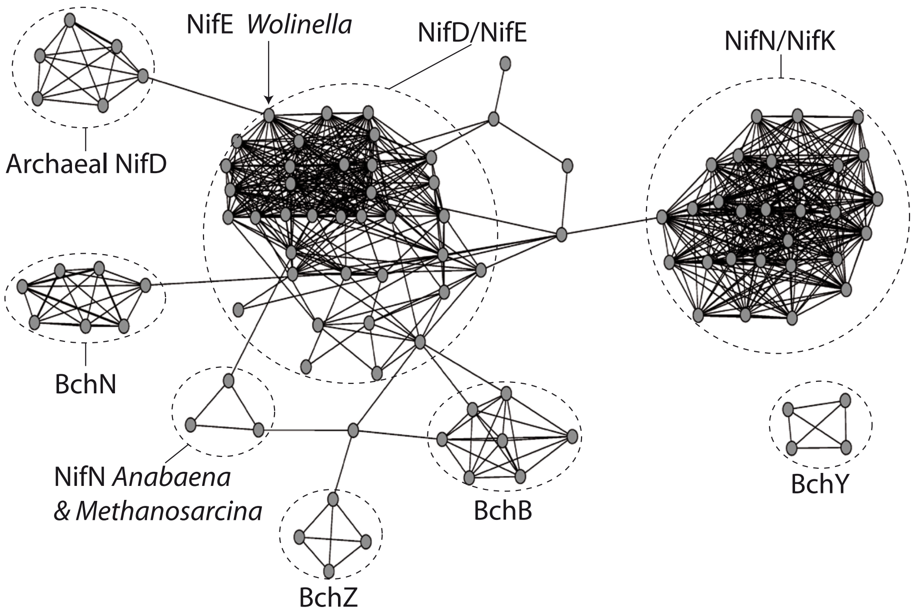
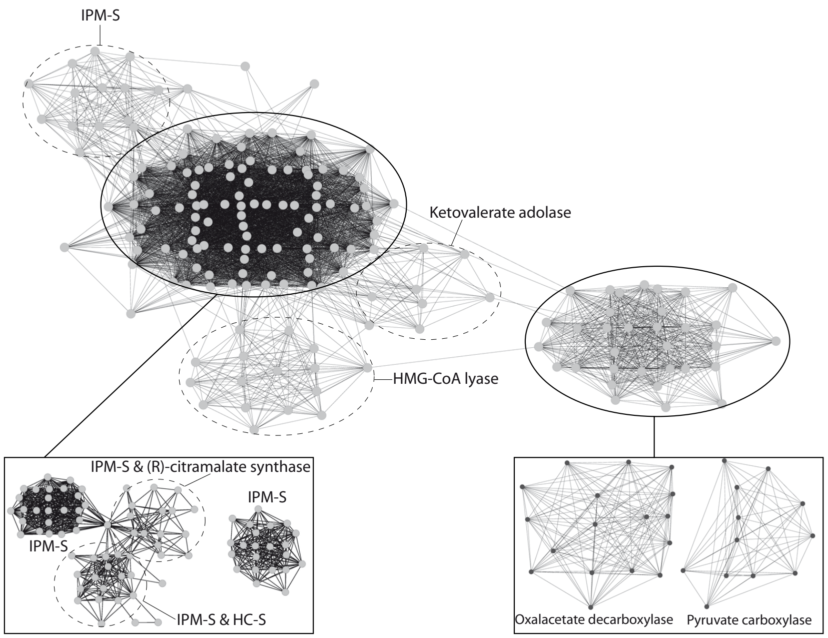
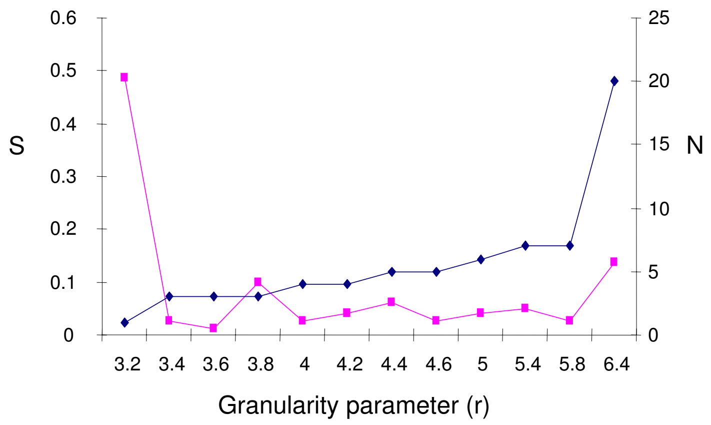


Fig. 3. Clusters of NifD,K,E,N showing the evolutionary connection among nitrogen fixation and photosynthesis

NifD, K, E, N showed sequence similarity with Bch proteins (Figure 3) that are involved in the biosynthesis of bacteriochlorophyll, supporting the idea of a common origin for these processes [41]. Similarly, NifV proteins cluster with metabolic proteins (Figure 4). NifB, X, Y are related through a common domain of about 90 amino acid; moreover, NifB has an additional SAM domain, found in proteins that catalyze diverse reactions, including methylation, isomerization, sulphur insertion, ring formation, anaerobic oxidation and protein radical formation. This is confirmed by a bibliographical data mining: SAM domains are among the most abundant protein-protein interaction motifs because they are remarkably versatile in their binding properties. Some identical SAM domains can interact with each other to form homodimers or polymers. In other cases, SAM domains can bind to other related SAM domains, to non-SAM domains, and even to RNA. Such versatility earns them functional roles in myriad biological processes, from signal transduction to transcriptional and translational regulation [40]. NifV is related to several widespread proteins: in Table 2 we list the functions associated to the 9 largest clusters, taken from Figure 4. Table 3 shows some of the statistics for the original and a clustered network. Particularly informative are the third (skewness) and fourth (kurtosis) moments. Skewness quantifies the degree of asymmetry of a distribution: when positive the distribution has an elongated tail on the right of the mean. Kurtosis indicates how much peaked is the distribution; the higher the kurtosis, the more peaked



(a) Clustered network of NifV related proteins.



(b) Entropy and number of clusters in function of the granularity parameter.

Fig. 4. a) Clustered network of NifV related proteins. Clusters obtained with r=3.6 and, inbox r=6.4. NifV enzymes are Homocitrate synthases (HC-S); they are evolutionarily related to Isopropylmalate synthases (IPM-S), (R)-citramalate synthases (CimA in Table 2), HMG-CoA lyases, Ketovalerate aldolases, Oxalacetate and Pyruvate carboxylases. The IPM-Synthases moreover belong to different groups; b) Clustering entropy (squares, S) and number of clusters in function of the granularity parameter (diamonds, N).

Table 2. Summary of the clustering procedure on NifV-related sequences. Granularity parameter (r) was 6.4, total number of clusters was 20. Cluster identifiers refer to the 5 clusters obtained at r=3.6, corresponding to the entropy minimum in the chart of Figure 4b. The procedure started from a full connected graph of 180 proteins coming from 55 different organisms comprising all diazotrophs. Abbreviations used: Id. is the cluster name; Nr Org. is the number of different organisms present in a given cluster; Nr Nodes is the total number of proteins in a given cluster; Avg. Nr nodes and Var are the average number of proteins of a given cluster coming from the same organism and the corresponding variance, respectively; Functions indicates the main function assigned to the proteins of a given cluster; we also report the percentage of proteins with that assignment.

Id.	**Nr Org.**	**Nr Nodes**	**Avg. Nr nodes**	**Var**	**Functions**
A	15	15	1.00	0.00	100% IPMS
B1	25	25	1.00	0.00	100% IPMS
B2	8	15	1.87	0.69	87% IPMS + 13% CimA
B3	17	20	1.18	0.28	55% HCS + 35% IPMS
B4	22	22	1.00	0.00	100% IPMS
C	14	16	1.14	0.29	75% HMG CoA lyase
D	4	8	2.00	4.00	100% 4-H-2-K aldolase
E1	16	17	1.06	0.06	100% Oxalacetate decarboxylase
E2	11	11	1.00	0.00	100% Pyruvate carboxylase

Table 3. Summary statistics calculated for NifV sub-network before and after clustering. Var. (corr.) indicates the Bessel correction of the variance. Skew. and Kurt. are the skewness and kurtosis of the distribution. See text for details.

Original (diameter=3)

	Mean	**Var (corr.)**	**Var.**	**Skew.**	**Kurt.**
Average Distances:	1.302	0.033	5.672	1.259	4.068
Betweenness Centrality:	25.977	344.529	59258.96	0.749	0.0885
Clustering Coefficients:	0.830	0.005	0.897	0.978	0.011

After MCL clustering with σ=0.15; r=2.0 (diameter=2)

Average Distances:	1.276	0.044	7.645	0.985	-0.181
Betweenness Centrality:	23.722	3691.506	634939	2.181	2.790
Clustering Coefficients:	0.963	0.009	1.517	-2.181	2.790

the distribution. The table shows that the "average distance" distribution of the original network is concentrated around the mean (high kurtosis) and has a long right tail (positive skewness). The clustered network has similar average distances but the distribution is less asymmetric and less peaked around the mean. Betweenness is quite normal in the original network, while it is highly peaked around the mean with a long right tail in the clustered example. Clustering coefficients of the original network are close to a normal distribution of identical mean and variance while the distribution of the clustered network is highly peaked around the mean but with a long left tail (negative skewness).

4 Conclusions

In this work we described a procedure that uses Blast e-values or scores to build a similarity matrix. Then a clustering algorithm and statistical analysis on networks extract the relevant information. We showed two case studies: plasmid-encoded genes and Nif proteins. Considering the rapid spreading of plasmids in bacterial populations and their dispensability, the analysis of plasmids suggests that the most represented functions might be considered a sort of fingerprint of the selective pressures acting in a given environment. The hubs were transposases; the other functions of the hub proteins are indeed indicative of the selective forces acting on these three pathogens. The presence of proteins involved in antibiotic resistance, virulence / pathogenesis or plasmid replication, partition, stabilization and transfer, suggests strong selective pressures acting on these microorganisms for a content devoted to host's protection, plasmid maintenance and spreading, in agreement with their life–style (most of them are continuously attacked by a set of different drugs because they are agents of several diseases). The network obtained have power-law distribution of degree of connectivity (Table 1), suggesting that plasmids might evolve through a preferential attachment, i.e. a greater probability of horizontal transfer or duplication rate of the most represented genes in the network.

In the second case study, Nif proteins, we found large clusters to be partitioned into several sub-clusters by progressively increasing the granularity parameter of the MCL. The sub-clusters at the minimum entropy (Figure 4b) are quite homogeneous with respect to the assigned biological function of nodes (Table 2).

In conclusion, our strategy aims at a deep characterization of sequence similarity networks through the calculation of several quantities related to network topology. These characterizations might also be used for comparisons between different datasets or organisms.

Finally we want to comment on the work in progress. Note that the clustering may show that some graphs are occurring more often than others and that there might be correlations between the presence of different clusters. These subgraphs, which are building blocks of a network, may be defined network motifs in analogy with sequence motifs, which are sequences occurring more often than random sequences. Network motifs may be identified by computing the statistical significance for each subgraph, i.e. the p-value.

References

1. Altschul, S.F., Madden, T.L., Schaffer, A.A., Zhalg, J., Zhalg, Z., Miller, W., Lipman, D.J.: Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucl. Acids Res. 25, 3389–3402 (1997)

2. Li, M., Ma, B., Kisman, D., Tromp, J.: Patternhunter II: highly sensitive and fast homology search. J. Bioinform. Comput. Biol. 2(3), 417–439 (2004)
3. Boccaletti, S., Latora, V., Moreno, Y., Chavez, M., Hwang, D.: Complex Networks: Structure and Dynamics. Physics Reports 424, 175–308 (2006)
4. Dorogovtsev, S.N., Mendes, J.F., Samukhin, A.N.: Structure of growing networks with preferential linking. Phys Rev. Lett. 85(21), 4633–4636 (2000)
5. Colizza, V., Barrat, A., Barthelemy, M., Vespignani, A.: The role of the airline transportation network in the prediction and predictability of global epidemics. Proc. Natl. Acad. Sci. U S A 103(7), 2015–2020 (2006)
6. Gross, T., D'Lima, C.J., Blasius, B.: Epidemic dynamics on an adaptive network. Phys Rev. Lett. 96(20), 208701 (2006)
7. Watts, D.J., Strogatz, S.H.: Collective dynamics of 'small-world' networks. Nature 393, 440–442 (1998)
8. Barabasi, A.L., Oltvai, Z.N.: Network biology: understanding the cell's functional organization. Nat. Rev. Genet. 5, 101–113 (2004)
9. Forster, J., Famili, I., Fu, P., Palsson, B.O., Nielsen, J.: Genome-scale reconstruction of the *Saccharomyces cerevisiae* metabolic network. Genome Res. 13, 244–253 (2003)
10. Monge, R.A., Roman, E., Nombela, C., Pla, J.: The MAP kinase signal transduction network in *Candida albicans*. Microbiology 152, 905–912 (2006)
11. Herrgard, M.J., Covert, M.W., Palsson, B.O.: Reconstruction of microbial transcriptional regulatory networks. Curr. Opin. Biotechnol. 15, 70–77 (2004)
12. Jones, C.E., Baumann, U., Brown, A.L.: Automated methods of predicting the function of biological sequences using GO and BLAST. BMC Bioinformatics 15, 272 (2005)
13. Tatusov, R.L., Natale, D.A., Garkavtsev, I.V., Tatusova, T.A., Shankavaram, U.T., Rao, B.S., Kiryutin, B., Galperin, M.Y., Fedorova, N.D., Koonin, E.V.: The COG database: new developments in phylogenetic classification of proteins from complete genomes. Nucleic Acids Res. 29, 22–28 (2001)
14. Li, L., Stoeckert Jr., C.J., Roos, S.D.: OrthoMCL: Identification of Ortholog Groups for Eukaryotic Genomes. Genome Res. (13), 2178–2189 (2003)
15. Enright, A.J., Van Dongen, S., Ouzounis, C.A.: An efficient algorithm for large-scale detection of protein families. Nucleic Acids Res. 30, 1575–1584 (2002)
16. van Dongen, S.: Graph clustering by flow simulation. (2000) PhD thesis `http://igitur-archive.library.uu.nl/dissertations/1895620/inhoud.htm`, `http://micans.org/mcl/`
17. Johnson, T.J., Siek, K.E., Johnson, S.J., Nolan, L.K.: DNA sequence and comparative genomics of pAPEC-O2-R, an avian pathogenic *Escherichia coli* transmissible R plasmid. Antimicrob Agents Chemother 49, 4681–4688 (2005)
18. Thomas, C.M., Nielsen, K.M.: Mechanisms of, and barriers to, horizontal gene transfer between bacteria. Nat. Rev. Microbiol. 3, 711–721 (2005)
19. Kondrashov, F.A., Kondrashov, A.S.: Role of selection in fixation of gene duplications. J. Theor. Biol. 21, 141–151 (2006)
20. Guimera, R., Sales-Pardo, M., Amaral, L.A.: Modularity from fluctuations in random graphs and complex networks. Phys Rev. E Stat. Nonlin. Soft Matter Phys 70, 025101 (2004)
21. Newman, M.E., Girvan, M.: Finding and evaluating community structure in networks. Phys Rev. E Stat. Nonlin. Soft Matter Phys 69, 026113 (2004)
22. Fortunato, S., Barthelemy, M.: Resolution limit in community detection. Proc. Natl. Acad. Sci. U S A 104, 36–41 (2007)

23. Gfeller, D., Chappelier, J.C., De Los Rios, P.: Finding instabilities in the community structure of complex networks. Phys Rev. E Stat. Nonlin. Soft Matter Phys 75, 056135 (2005)
24. Tetko, I.V., Facius, A., Ruepp, A., Mewes, H.W.: Super paramagnetic clustering of protein sequences. BMC Bioinformatics 6, 82 (2005)
25. Zhang, Z., Luo, Z.W., Kishino, H., Kearsey, M.J.: Divergence Pattern of Duplicate Genes in Protein-Protein Interactions Follows the Power Law. Mol. Biol. Evol. 22, 501–505 (2005)
26. Uetz, P., Giot, L., Cagney, G., Mansfield, T.A., Judson, R.S., Knight, J.R., Lockshon, D., Narayan, V., Srinivasan, M., Pochart, P., Qureshi-Emili, A., Li, Y., Godwin, B., Conover, D., Kalbfleisch, T., Vijayadamodar, G., Yang, M., Johnston, M., Fields, S., Rothberg, J.M.: A comprehensive analysis of protein-protein interactions in Saccharomyces cerevisiae. Nature 403, 623–627 (2000)
27. Wagner, A.: The yeast protein interaction network evolves rapidly and contains few redundant duplicate genes. Mol. Biol. Evol. 18, 1283–1292 (2001)
28. Ito, T., Chiba, T., Ozawa, R., Yoshida, M., Hattori, M., Sakaki, Y.: A comprehensive two-hybrid analysis to explore the yeast protein interactome. PNAS 98, 4569–4574 (2001)
29. Goh, K.I., Oh, E., Jeong, H., Kahng, B., Kim, D.: Classification of scale-free networks. Proc. Natl. Acad. Sci. U S A 99, 12583–12588 (2002)
30. Li, S., Armstrong, C.M., Bertin, N., Ge, H., Milstein, S., Boxem., M., Vidalain, P.-O., Han, J.-D.J., Chesneau, A., Hao, T., Goldberg, D.S., Li, N., Martinez, M., Rual, J.-F., Lamesch, P., Xu, L., Tewari, M., Wong, S.L., Zhang, L.V., Berritz, G.F., Jacotot, L., Vaglio, P., Reboul, J., Hirozane-Kishikawa, T., Li, Q., Gabel, H.W., Elewa, A., Baumgartner, B., Rose, D.J., Yu, H., Bosak, S., Sequerra, R., Fraser, A., Mange, S.E., Saxton, W.M., Strome, S., van den Heuvel, S., Piano, F., Vandenhaute, J., Sardet, C., Gerstein, M., Doucette-Stamm, L., Gunsalus, K.C., Harper, J.W., Cusick, M.E., Roth, F.P., Hill, D.E., Vidal, M.: A map of the interactome network of the metazoan C. elegans. Science 303, 540–543 (2004)
31. Hughes, A.L., Friedman, R.: Gene Duplication and the Properties of Biological Networks. J. Mol. Evol. 61, 758–764 (2005)
32. Koonin, E.V., Wolf, Y.I., Karev, G.P.: The structure of the protein universe and genome evolution. Nature 420, 218–223 (2002)
33. Larsen, M.H., Figurski, D.H.: Structure, expression, and regulation of the kilC operon of promiscuous IncP alpha plasmids. J. Bacteriol. 176, 5022–5032 (1994)
34. Arthur, D.C., Ghetu, A.F., Gubbins, M.J., Edwards, R.A., Frost, L.S., Glover, J.N.: FinO is an RNA chaperone that facilitates sense-antisense RNA interactions. EMBO J. 22, 6346–6355 (2003)
35. Aguirre-Ramirez, M., Ramirez-Santos, J., Van Melderen, L., Gomez-Eichelmann, M.C.: Expression of the F plasmid ccd toxin-antitoxin system in *Escherichia coli* cells under nutritional stress. Can J. Microbiol. 52, 24–30 (2006)
36. Escobar-Paramo, P., Giudicelli, C., Parsot, C., Denamur, E.: The evolutionary history of *Shigella* and enteroinvasive *Escherichia coli* revised. J. Mol. Evol. 57, 140–148 (2003)
37. Hartman, A.B., Essiet, I.I., Isenbarger, D.W., Lindler, L.E.: Epidemiology of tetracycline resistance determinants in Shigella spp. and enteroinvasive Escherichia coli: characterization and dissemination of tet(A)-1. J. Clin. Microbiol. 41, 1023–1032 (2003)
38. Call, D.R., Kang, M.S., Daniels, J., Besser, T.E.: Assessing genetic diversity in plasmids from Escherichia coli and Salmonella enterica using a mixed-plasmid microarray. J. Appl. Microbiol. 100, 15–28 (2006)

39. Sperotto, R.A., Gross, J., Vedoy, C., Passaglia, L.M., Schrank, I.S.: The electron transfer flavoprotein fixABCX gene products from *Azospirillum brasilense* show a NifA-dependent promoter regulation. Curr. Microbiol. 49, 267–273 (2004)
40. Qiao, F., Bowie, J.U.: The many faces of SAM. Sci STKE. 286, re7 (2005)
41. Burke, D.H., Hearst, J.E., Sidow, A.: Early evolution of photosynthesis: clues from nitrogenase and chlorophyll iron proteins. Proc. Natl. Acad. Sci. U S A 90(15), 7134–7138 (1993)

Gene Expression Mining for Cohesive Pattern Discovery

Ramkishore Bhattacharyya[1] and Balaram Bhattacharyya[2]

[1] Microsoft India (R&D) Pvt. Ltd., Gachibowli, Hyderabad – 500 032, India
[2] Dept of Computer and System Sciences, Visva-Bharati Universty, Santiniketan – 731235, India
ramb@microsoft.com, balaramb@gmail.com

Abstract. Identification of genes that share common biological functions is one of the main objectives in molecular biology. The present work focuses on developing an appropriate mechanism for discovery of such genesets from microarray data. We introduce a conceptual property 'cohesion' among genes as representative of common biological function, under influence of which a geneset behave coherently. Such genesets are marked as 'cohesive'. Depending on "100% cohesion" equivalence relation, the entire set of associated genes is decomposed into a number of disjoint equivalence classes, each with unique behavior. The equivalence classes form several disjoint affinity groups, members within a group having pair-wise direct interaction. Each group may be called a cohesive gene cluster. A data mining technique for cohesive geneset discovery is developed and applied on expression data to discover intra-cluster gene relationships for extracting natural coherent genesets. Experiments with some cancer datasets discover thousands of long confident patterns within reasonable time, showing its superiority over classical association discovery techniques. Results can provide important insight into molecular biology and biomedical research.

Keywords: Association rule, cohesion, direct interaction, gene cluster, microarray mining.

1 Introduction

DNA microarray technology opens up the scope of studying genes' collective behavior and analyzing different groups of genes to have insight into common biological properties that bind them together. Scopes are further improved in providing series of experimental results on samples under different cellular and physical conditions. It produces expression profiles of thousands of genes in a single experiment, widely known as gene expression data. Similar experiments on hundreds of samples of identical species under the same physical and biological conditions are usually conducted to capture possible variations in expression levels of genes under unchanged environments. Such datasets are particularly important in studying gene regulatory pathways.

Series of similar experiments yield a dataset comprising of thousands of columns and hundreds of rows. While a row represents expression levels of large number of genes of the cell under study, a column represents expression pattern of a particular gene distributed over hundreds of the same cellular samples under identical conditions. Apart from experimental errors, large variation in expression levels of particular genes over the experimental samples is an indication of change in regulatory function while external physical condition remained unchanged. Expression data on the same cell type but under different physical or biological conditions like heat treatment or in diseased state are valuable to study their effect on genes' function. Analysis on such dataset can yield an insight into temporal behavior of genes' activity, individual or collectively.

Thus, gene expression profiles become an enormous source for extracting knowledge on gene function, their inter-dependence and co-occurring. Exploring the datasets is important for finding an insight into gene regulation mechanism, metabolic pathways and detecting deviations in gene expression patterns or gene network under diseased conditions. Findings of such analyses can provide insights into molecular biology and biomedical researches. While such datasets are valuable source for knowledge extraction on gene functions, high level of experimental errors generates a motivation on utilizing the datasets for studying genes' collective behavior, rather than individual. Hence, extracting sets of genes that have high confidence in co-existence is a logical step for discovery of knowledge on gene's function from such datasets.

Association Rule Discovery (ARD) [1] can be a foundational technique in predicting cellular control on expression of their genes and intra-cellular activity by deciphering functional relationships among genes [7]. An association rule is an expression $A \xrightarrow{\sigma,\mu} C$, where A, C are itemsets and $A \cap C = \emptyset$, A and C are called antecedent and consequent respectively. Support σ of the rule is given by the percentage of records containing the itemset $A \cup C$. Confidence μ of the rule is the conditional probability that a record, containing A, also contains C and is given as $\mu = \sigma(A \cup C) / \sigma(A)$. A rule with genes g_1, g_2, and g_3 may be of the form: g_1 [+] $\xrightarrow{\sigma,\mu} g_2$ [+] and g_3 [-]. This means when gene g_1 is expressed, gene g_2 is expressed and gene g_3 is repressed with a certainty factor of μ% and they appear together in σ% of total sample experiments.

ARD is an unsupervised data mining technique that provides means to explore interesting associations constrained by a minimal frequency of occurrence. While this might be an effective idea from efficiency perspective of algorithms, two important drawbacks are prominent. First, satisfying minimal frequency constraint is not an essential criterion for a rule to be interesting. The rule {Down's syndrome} $\rightarrow$ {trisomy 21} (chromosomal defect) is almost 100% confident but with only around 0.1% frequency and hence may not meet the criteria. A lower choice for minimal frequency would entrap an algorithm into combinatorial explosion of patterns. Furthermore, in a set of millions of rules, finding the really interesting ones poses another challenge. Second, every application domain has its own natural properties that bind the associated items into several affinity groups. An algorithm must regard those properties in order to discover natural and reliable associations. Unfortunately, classical support-based techniques do not. Items in such a group tend to appear mostly united rather being scattered. So, it does not matter how many times they appear together. What really matters is how confidently they appear together. We introduce a new concept cohesion of itemsets that keeps track of this information. It is defined as the ratio of number of records containing the itemset to the number of records containing any non-empty subset of that itemset.

Cohesive patterns are much more efficient in predicting any consequence. Consider the rule {malaria}→{weakness, fever}. As fever and weakness are one of the most prevalent symptoms, the consequent itemset is not at all cohesive. And indeed, weakness and fever cannot diagnose to malaria. Consider another rule {hepatitis}→{anorexia, jaundice, hepatomegaly}. To some extent, the symptoms at the consequent are cohesive and together they can confidently diagnose to hepatitis. Thus, cohesive patterns produce more informative and acceptable rules. We argue that only such rules are worthy to be considered in any important decision making system. Thus, cohesion will be instrumental in discovery of natural binding properties prevailing in the application domain for which data is collected. Further, it solves the problem of exploring confident associations that are not constrained by minimal frequency.

Classical ARD concentrates only on high-support and high-confidence rules for sake of its scalability. Thus, highly confident rules remain unearthed for their low support [9]. Count of such rules in a microarray dataset might be enormous, carrying valuable insight about genetic interaction. Setting extremely low values of support, e.g., 0.1% to 12% to catch such rules is not the solution. It causes gene pattern and rule explosions as their cardinalities are bounded by 2^n and 3^n, n being the number of frequent genes. Choosing the right set of rules from such the huge for biological interpretation creates a problem. Above all, as the process is knowledge discovery, setting parameters for unknown target seems futile. Thus, cohesive itemset mining will be more suitable for high confidence geneset discovery from microarray data.

Microarray datasets typically contain a handful few experiments (rows), each described by the activity levels of thousands of genes (columns). Dimensionality of such datasets poses great challenge to the existing family of frequent itemset mining algorithms [2][8][11][12]. These support-based techniques restrict their search space by taking as few as 5% of the entire set of genes into account. As a result, most of the potential interesting gene relationships get slipped away. A new family of support-based row-enumeration and closed pattern mining algorithms [10][4][6][5], designed specially for gene expression analysis, perform significantly better than the previous ones. However, support-dependency of such algorithms still creates a problem in discovery of numerous low-support high-confidence associations which may not have been hypothesized yet.

Differently put, high confidence of a rule indicates a high value for the joint probability that a record contains the consequent, given that it contains the antecedent; no matter how many times both antecedent and consequent appear together. That is, we have to look for a new property that can reveal information on tendency of coexistence of items within a set. We term this property as *cohesion* of an itemset.

Gene expression data usually produces a number of sets that exhibit 100% cohesion. This indicates that all genes within a set appear as a group with their respective expression levels. None of them appear in any experiment without being into its group. Such a set behaves as a single entity producing several 100% confident rules. We prove that "100% cohesion" is an equivalence relation that decomposes the entire gene set into a number of disjoint equivalence classes. Thus effectively, number of genes that are to be involved in the mining process reduces to the number of different equivalence classes.

We propose a notion of *direct interaction* between a pair of genes. Based on the idea, genes are classified into several affinity groups, gene clusters. All possible pairs within such a cluster have direct interactions, ensuring that a cluster contains mostly inter-dependent genes.

Finally, we develop a new cohesion-based association mining algorithm dCAR-DIAC[1] and apply it within each cluster for generating cohesive patterns which are the gateway in discovery of confident gene associations.

2 Cohesion: A Concept

Let $I = \{1, 2,\ldots, m\}$ be a set of items and $T = \{1, 2,\ldots,n\}$ be a set of transaction identifiers or tids. For an itemset X, $t(X)$ is the set of all tids containing X as a subset.

Definition 1. (Cohesion) Cohesion of an itemset X is a property prevailing among its items that can mathematically be defined as the ratio of cardinalities of two sets viz. the set of all tids containing X as a subset and the set of all tids containing at least one item x of X as a subset and is given by,

$$\xi(X) = \left|\bigcap_{x\in X} t(x)\right| / \left|\bigcup_{x\in X} t(x)\right| . \tag{1}$$

Definition 2. (Cohesive itemset) An itemset X is called cohesive, if $\xi(X)$ is no less than pre-specified threshold.

So, cohesion of an itemset X needs both $\bigcap_{x\in X} t(x)$ and $\bigcup_{x\in X} t(x)$ to be computed. For the sake of conciseness, we choose $\lambda(X) = \bigcap_{x\in X} t(x)$ and $\rho(X) = \bigcup_{x\in X} t(x)$.

Lemma 1. Cohesive itemsets retain downward closure property [2].

Proof. For all itemsets X and Y such that $X \subseteq Y$, $|\lambda(X)| \geq |\lambda(Y)|$ and $|\rho(X)| \leq |\rho(Y)|$. So, $\xi(X) = |\lambda(X)| / |\rho(X)| \geq |\lambda(Y)| / |\rho(X)| \geq |\lambda(Y)| / |\rho(Y)| = \xi(Y)$. So, given a threshold for cohesion, if an itemset Y is cohesive, then so are all its subsets. Hence, all subsets of a cohesive itemset must also be cohesive and downward closure property follows.

3 Association Discovery Using Cohesion

Mining cohesive genesets in microarray data is carried out in three sub-steps.

3.1 Identifying "100% Cohesive" Genesets

It has been noticed that microarray datasets generate several patterns that are quite infrequent but their cohesion is 1 (100%). Interestingly, some of the patterns contain 30 or 40 or even more genes. Due to downward closure property of cohesive item-

[1] dCARDIAC stands for <u>d</u>iffset-based <u>C</u>onfident <u>A</u>ssociation <u>R</u>ule <u>D</u>iscovery using <u>I</u>temset <u>C</u>ohesion; the second 'A' is gratuitous.

sets, all non-empty subsets of such a pattern must also have cohesion equals to 1. So any cohesive itemset mining algorithm, that attempts to mine such patterns, must explore their power set which involves mining as many as 2^{30} or 2^{40} patterns. The question naturally arises how to explore such important associations without generating power sets.

We define a relation $\mathcal{R}$ as follows: two elements g_i and g_j are $\mathcal{R}$-related if $t(g_i) = t(g_j)$. This implies $\{t(g_i) \cap t(g_j)\} = \{t(g_i) \cup t(g_j)\}$ and $\xi(g_i g_j) = | t(g_i) \cap t(g_j) | / | t(g_i) \cup t(g_j) | = 1$. Hence, $g_i \mathcal{R} g_j$ implies cohesion of $(g_i g_j)$ is 1.

It can easily be proved that '=' relation between two sets is reflexive, symmetric, transitive and hence equivalence. The relation $\mathcal{R}$ therefore decomposes the entire geneset into several disjoint equivalence classes. Due to downward closure property, all itemsets within such a class have cohesion equals 1. Such classes produce several essential gene associations with 100% confidence. As mining proceeds, each of these classes is treated as a single entity with an identifier and a tidset. Genes within a class appear to be bound by some common biological property.

3.2 Cluster Formation

As stated earlier, there possibly exist some common biological properties that bind a set of genes together. Under different cellular conditions, expression levels of these genes are much inter-related compared to other genes that are not within that group. From this point of view, we partition the entire set of equivalence classes into several clusters, each containing a number of genes having pair-wise direct interaction. Important intra-cluster associations are mined using cohesive itemset discovery algorithm. Clustering entire geneset has some advantages. Let $\xi_1, \xi_2, \ldots, \xi_n$ be threshold cohesions for different clusters for mining intra-cluster cohesive patterns. These thresholds are largely dependent on whether a cluster is tightly bound or loosely bound. For a tightly bound cluster, a lower choice for threshold cohesion involves innumerable patterns, resulting in storage scarcity. For a loosely bound, a higher choice does not explore significant associations. Had it been a single cluster containing all genes, threshold for cohesion must be set to max $(\xi_1, \xi_2, \ldots, \xi_n)$ to avoid storage scarcity. As a result, several gene relations at moderate cohesion would remain in dark. We exploit the concept of clustering to keep this possibility alive by choosing different thresholds for different clusters. Inasmuch, it may be noted that some relations between genes that belong to different clusters may be skipped, but that is very nominal as most of the prominently expressed genes are captured into clusters. In fact, ideal situation would be keeping all genes/proteins of a dataset in same cluster and start mining. But this is just infeasible due to huge memory requirement. We must dissociate the dataset in several clusters to unearth information. Success lies in how intelligent the clustering technique is.

Definition 3. (Direct interaction) Two genes g_i and g_j have direct interaction between them if $g_i \xrightarrow{\mu_1} g_j$ and $g_j \xrightarrow{\mu_2} g_i$, together written as $g_i \overset{\mu_1}{\underset{\mu_2}{\longleftrightarrow}} g_j$, μ_1 and μ_2, being confidences of the two associations, are no less than pre-specified threshold μ. Also the product of μ_1 and μ_2 is no less than another pre-specified threshold τ.

Associations, with confidence below 50%, have more counter examples than examples. Therefore choice of μ should not be below 50%. A suitable choice for τ disregards those associations for which both μ_1 and μ_2 are smaller, close to μ. A gene is allowed to enter a cluster if it has direct interaction with all cluster members. Using direct interaction in gene clustering ensures several confident associations. The following lemma justifies our claim.

Lemma 2. Let their exists two direct interactions $g_i \underset{\mu_2}{\overset{\mu_1}{\longleftrightarrow}} g_j$ and $g_i \underset{\mu_4}{\overset{\mu_3}{\longleftrightarrow}} g_k$. Then, there must exist an association $g_i \xrightarrow{\mu_1+\mu_3-1} g_j g_k$.

Proof. Consider only the forward associations.
$g_i \xrightarrow{\mu_1} g_j \Rightarrow g_i \xrightarrow{1-\mu_1} g_j' \Rightarrow |t(g_i g_j')| / |t(g_i)| = (1-\mu_1) \Rightarrow |t(g_i g_j')| = (1-\mu_1)\ |t(g_i)|$. Similarly, $g_i \xrightarrow{\mu_3} g_k \Rightarrow g_i \xrightarrow{1-\mu_3} g_k' \Rightarrow |t(g_i g_k')| / |t(g_i)| = (1-\mu_3) \Rightarrow |t(g_i g_k')| = (1-\mu_3)\ |t(g_i)|$. So, $|t(g_i g_j') \cup t(g_i g_k')| \leq |t(g_i g_j')| + |t(g_i g_k')| = (1-\mu_1+1-\mu_3)\ |t(g_i)|$. So, $|t(g_i g_j') \cup t(g_i g_k')| / |t(g_i)| \leq (2-\mu_1-\mu_3)$. So, $1-(|t(g_i g_j') \cup t(g_i g_k')| / |t(g_i)|) \geq (\mu_1+\mu_3-1)$. Therefore, $|t(g_i g_j g_k)| / |t(g_i)| \geq (\mu_1+\mu_3-1)$. Hence there must exist an association $g_i \longrightarrow g_j g_k$ with confidence lower bound by $(\mu_1+\mu_3-1)$.

For a choice μ_1=0.9 and μ_2=0.8, confidence of the rule $g_i \longrightarrow g_j g_k$ is greater or equal to (0.9+0.8-1) = 0.7. Therefore, a gene which has direct interactions with all cluster members, gives a number of important associations involving them.
Choosing the concept of reversible association in defining direct interaction ensures another important aspect: cohesion of the geneset $\{g_i g_j g_k\}$ has also a lower bound (this is a theoretically proven result). High cohesion ensures that the associated items are mostly united rather being scattered. Hence, the associations they produce have high acceptability.

3.3 Intra-cluster Association Mining

We now introduce a new vertical mining algorithm dCARDIAC for cohesive itemset mining. Let $X = \{x_1, x_2, \ldots, x_n\}$ be an itemset. We follow the notation X' for $\{x_1', x_2', \ldots, x_n'\}$. If $t(X)$ is the set of tids containing X as a subset, $t(X')$ is the set of tids containing X' as a subset i.e. the set of tids containing none of $x \in X$ as a subset. For an itemset X,

$$\rho(X) = T \setminus \lambda(X') \,. \tag{2}$$

$$|\rho(X)| = n - |\lambda(X')| \,. \tag{3}$$

Proof. The two sets $\rho(X)$ and $\lambda(X')$ are mutually disjoint while their union is the entire dataset T. Therefore equation (**2**) follows directly.

As $\lambda(X') \subseteq T$, $|T \setminus \lambda(X')| = |T| - |\lambda(X')| = n - |\lambda(X')|$. Hence equation (**3**) follows.

Lemma 3. For all itemsets X, Y such that $Y \supseteq X$, $\lambda(Y') \subseteq \lambda(X')$.

Proof. Since, $Y \supseteq X$, $\rho(Y) \supseteq \rho(X)$. So, $T \setminus \rho(Y) \subseteq T \setminus \rho(X)$. Hence, $\lambda(Y') \subseteq \lambda(X')$.
So, instead of maintaining the set $\rho(X)$ of monotonic increasing cardinality, it is wise to maintain $\lambda(X')$ which is of monotonic decreasing cardinality. This equally serves the purpose.

We now turn our attention to the diffset-based algorithm dCARDIAC for cohesive itemset mining. Diffset of a k-itemset is the set of tids, obtained by difference of its tidset from the tidset of its (k-1)-length prefix. So if P be a prefix and x be an item, with $t(P)$ and $t(x)$ being the tidsets respectively, diffset of the itemset Px, $d(Px) = t(P) \setminus t(x)$. If P is null set, $t(\varnothing)$ is simply T. So, diffset of an item X, $d(x) = T \setminus t(x)$ which is simply the set of tids not containing x as a subset.

In a diffset-based algorithm, all that are maintained are only the diffsets of cohesive itemsets instead of tidsets. So how is to compute diffset of an itemset from the diffsets of the two generating itemsets?

$$d(Pxy) = d(Py) \setminus d(Px)\,. \tag{4}$$

$$|\lambda(Pxy)| = |\lambda(Px)| - |d(Pxy)|\,. \tag{5}$$

$$d(P'x'y') = d(P'y') \setminus d(P'x')\,. \tag{6}$$

$$|\rho(Pxy)| = |\rho(Px)| + |d(P'x'y')|\,. \tag{7}$$

Proof. Let tid be a transaction identifier. By definition,

$$\begin{aligned}
d(Pxy) &= \{tid\colon Px \in tid \text{ and } Py \notin tid\}\\
&= \{tid\colon Px \in tid\} \setminus \{tid\colon Py \in tid\}\\
&= t(Px) \setminus t(Py)\\
&= \{tid\colon P \in tid \text{ and } Py \notin tid\} \setminus \{tid\colon P \in tid \text{ and } Px \notin tid\}\\
&= \{t(P) \setminus t(Py)\} \setminus \{t(P) \setminus t(Px)\}\\
&= d(Py) \setminus d(Px) && \text{[Eq. (\textbf{4}) follows].}\\
|\lambda(Pxy)| &= |\{tid\colon Pxy \in tid\}|\\
&= |\{tid\colon Px \in tid\}| - |\{tid\colon Px \in tid \text{ and } Py \notin tid\}|\\
&= |\lambda(Px)| - |d(Pxy)| && \text{[Eq. (\textbf{5}) follows].}
\end{aligned}$$

Eq. (**6**) can be proved in the same way as that of Eq. (**4**) only by replacing the prefix and items by their complements.

$$\begin{aligned}
|\rho(Pxy)| &= n - |\lambda(P'x'y')| && \text{[from eq. (\textbf{3})]}\\
&= n - \{\,|\lambda(P'x')| - |d(P'x'y')|\,\} && \text{[from eq. (\textbf{5})]}\\
&= n - |\lambda(P'x')| + |d(P'x'y')|\\
&= |\rho(Px)| + |d(P'x'y')| && \text{[Eq. (\textbf{7}) follows]}
\end{aligned}$$

Below in **Fig. 1**, we present dCARDIAC algorithm for cohesive itemset mining.

```
dCARDIAC([P],mincohesion)
for all $X_i \in [P]$ do
  for all $X_j \in [P] \mid j > i$ do
    $X = X_i \cup X_j$;
    $d(X) = d(X_j) \setminus d(X_i)$; $|\lambda(X)| = |\lambda(X_i)| - |d(X)|$;
    $d(X') = d(X_j') \setminus t(X_i')$; $|\rho(X)| = |\rho(X_i)| + |d(X')|$;
    $\xi(X) = |\lambda(X)| / |\rho(X)|$;
    if $\xi(X) \geq$ mincohesion
      $S = S \cup X$;
  if $S \neq \emptyset$ then call dCARDIAC($S$,mincohesion)
```

Fig. 1. The dCARDIAC algorithm

Cohesion aims at discovering the vital associations by generating as few genesets as possible. But that does not necessarily mean it is incapable of exploring as much associations as support measure does. The following lemma justifies it.

Lemma 4. Under same threshold for cohesion and support, the set of all cohesive itemsets is a superset of the set of all frequent itemsets.

Proof For an itemset X, $\xi(X) = |\lambda(X)| / |\rho(X)| \geq |\lambda(X)| / |T| = \sigma(X)$ as $|\rho(X)|$ is upper bounded by $|T|$. So, with same threshold for cohesion and support, all frequent itemsets are cohesive. But there possibly exists cohesive itemsets that are not frequent. For a dataset count of such itemsets can be large enough as will be shown in the experimental section. The additional itemsets that come to the picture actually contribute to the low support high-confidence rules.

3.4 Data Preparation and Illustration of dCARDIAC Algorithm

Fig. 2a presents a sample matrix for gene expression data in its traditional format. The rows denote different cellular conditions while the columns denote different genes. M[t,g] contains a real number which is the quantitative expression for gene g under treatment t. Here, the example contains only positive integers but it may

Tid	a	b	c	d	e	f	g
1	65	32	72	30	54	19	35
2	70	48	85	44	33	15	28
3	46	56	78	27	39	27	41
4	38	29	53	22	59	32	47
5	57	45	92	37	26	12	21
6	35	25	41	19	64	19	52

Fig. 2a. Sample matrix for gene expression data

Tid	a	b	c	d	e	f	g
1	1	0	1	0	1	0	0
2	1	1	1	1	0	0	0
3	0	1	1	0	0	1	1
4	0	0	0	0	1	1	1
5	1	1	1	1	0	0	0
6	0	0	0	0	1	0	1

Fig. 2b. Processed boolean matrix

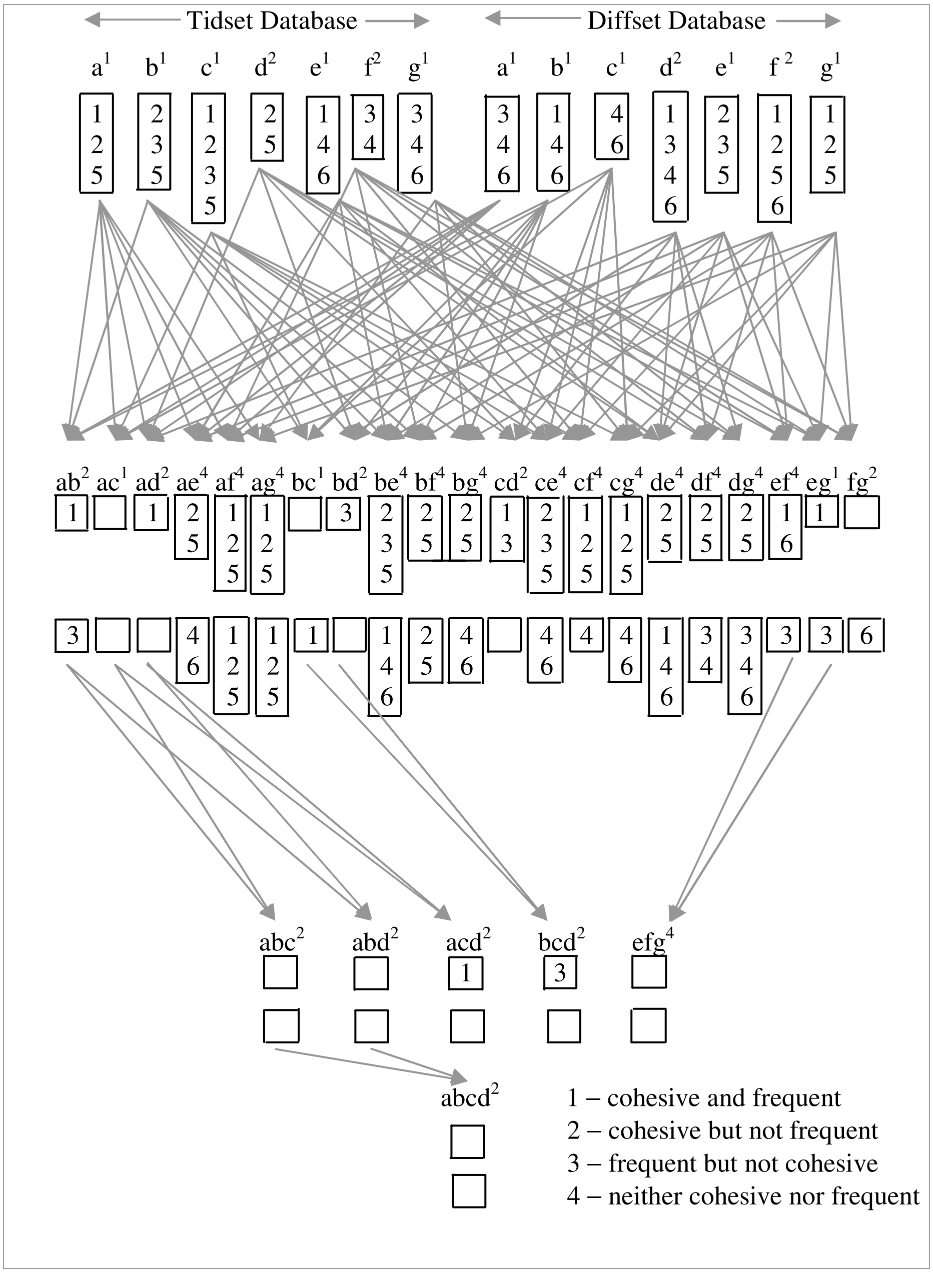

Tidset Database
Diffset Database
a1 b1 c1 d2 e1 f2 g1
ab2 ac1 ad2 ae4 af4 ag4 bc1 bd2 be4 bf4 bg4 cd2 ce4 cf4 cg4 de4 df4 dg4 ef4 eg1 fg2
abc2 abd2 acd2 bcd2 efg4
abcd2
1 – cohesive and frequent
2 – cohesive but not frequent
3 – frequent but not cohesive
4 – neither cohesive nor frequent

Fig. 3. Illustration of dCARDIAC algorithm with vertical format of processed dataset

contain negative values as well indicating under activity of a gene. Activity levels of the genes are converted to two categorical values, either expressed or repressed. So after preprocessing, number of items in a dataset can be at most double the number of genes involved in the experiment. This is an essential preprocessing step for an ARD algorithm to take over.

We adopt "*max* minus x%" technique, presented in literature [3], to convert each quantitative expression into boolean value. For a particular gene g_i with maximum expression level *max*, an expression level greater or equal to (1-x/100)**max* is assigned '1' indicating presence of gene g_i. Otherwise, it is assigned '0' indicating absence of gene g_i. Choosing x=25%, the dataset in Fig. 2a is converted into the boolean matrix in Fig. 2b.

Fig. 3 illustrates the dCARDIAC algorithm, threshold for cohesion being set to 50%. It starts with the vertical format of the processed boolean datasets in Fig. 2b, either with tidset or with diffset format. The superfix for each itemset indicates its status which is defined below. We see that itemset abcd is present in 2 out of 6 records. So, it is not frequent. But its cohesion is 2/4 i.e. 0.5. Therefore abcd is cohesive and so are all its subsets owing to downward closure property. Notice that the set of cohesive itemsets is a proper superset of the set of frequent itemsets. The additional itemsets actually contribute to the high confidence associations which have below 50% support. In experimental section, we will see that count of such low support and high confidence rules might be greatly which are mined out efficiently using cohesion.

4 Experiment and Analysis

Experiments have been conducted on a 2GHz AMD Athlon PC with 1GB primary memory. Algorithms are implemented in C++ and run on Debian Linux platform. For performance evaluation purpose, we have chosen two cancer datasets. The **Ovarian Cancer** (**OC**) dataset is for identifying proteomic patterns in serum that distinguish ovarian cancer from non-cancer cases. It contains 253 samples, each described by activity level of 15,154 proteins. In two normalization techniques "*max* – 30%" and "*max* – 50%", average record lengths are 1360.16 and 4472.01 respectively. The **OC** dataset has no negative activity level. Therefore number of different items after processing is same as that of number of associated proteins. The **Prostate Cancer** (**PC**) dataset contains 157 samples, each described by activity level of 12600 proteins. Average record lengths are 614.185 and 1398.56 in two normalization techniques "*max* – 30%" and "*max* – 50%" respectively. With two category levels for each protein (one for +ve, another for –ve activity level), number of different items in **PC** dataset becomes 22,989.

Result of the step for identifying equivalence classes is tabulated in **Fig. 4**. Dimension of a dataset after processing has been indicated within bracket. For **OC**, number of different items is 15,154 which incidentally same as that of number of different proteins associated to the dataset, reason being discussed above. In contrast to the **OC** dataset, **PC** contains both +ve and –ve activity levels. Hence in **PC** dataset, the figure is 22,989 which is almost double as that of number of proteins associated to the dataset. The figure may not be exactly double since every gene may not have both +ve and -ve activity levels. In **OC** dataset, number of different classes reduces to 9810 and 12955 in two respective normalization techniques. In **PC** dataset, number of different

Normalization technique	Ovarian Cancer (253×15,154)			Prostate Cancer (157×22,989)		
	# equiv. class	class size (1st three)	time (sec)	# equiv. class	class size (1st three)	time (sec)
"*max* – 30%"	9810	130, 82, 58	7.08	11358	1454, 987, 242	16.62
"*max* – 50%"	12955	81, 45, 18	8.01	16434	873, 204, 8	21.13

Fig. 4. Performance of 100% cohesive class identification technique

equivalence classes is 11358 and 16434 respectively. For both datasets, cardinalities of the first three equivalence classes have been shown, both for the two techniques. Effect of the technique can easily be understood just by seeing high cardinality of a class.

Fig. 5a and Fig. 5b show performance and number of different clusters, threshold confidence (i.e. μ in definition of direct interaction) has been decremented step-wise. The value of τ has been chosen to the average of μ and μ^2. A higher choice for μ generates several clusters with less number of genes within each. A lower choice will tend the genes to belong to same cluster, cluster number becomes extremely less. Appropriate choice for μ is largely dependent on the application dataset. In subsequent experiments, we choose μ as 0.7 and 0.5 for **OC** and **PC** datasets respectively.

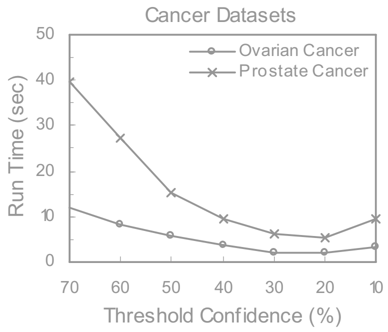


Fig. 5a. Performance study

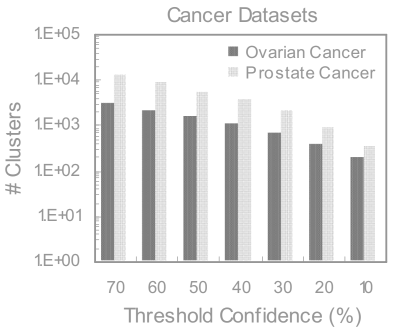


Fig. 5b. Number of clusters

For each of the two datasets, we choose two clusters. One is tightly bound producing numerous long patterns at higher cohesion. Another is loosely bound that does not produce so many patterns at higher cohesion. Performance of dCARDIAC algorithm for geneset mining in the four clusters has been tested in Fig. 6. In the tightly bound clusters of the OC and PC datasets, threshold cohesion cannot be decremented below 84% and 86% respectively. On the other hand, for the two loosely bound clusters, threshold can go down to 65%. Memory runs dry below those thresholds. Had we not decomposed the genes into clusters, we could not decrease thresholds cohesion below 84% and 86% for **OC** and **PC**. All patterns, mined below that cut-off cohesion, would remain in dark. Results show dCARDIAC is very much scalable. Obviously higher memory is required for mining at even lower threshold.

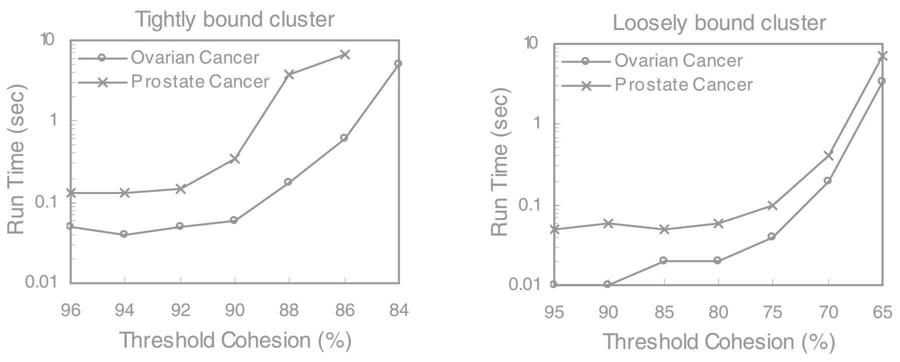


Fig. 6. Performance of dCARDIAC for two different clusters of each dataset

Fig. 7 and Fig. 8 show count of genesets and single-antecedent rules respectively for those four clusters. Threshold confidence of the rules is set to 80%. Observe count of genesets and rules for the tightly bound and loosely bound clusters generated at 90% thresholds. A tightly bound cluster generates many more patterns and rules than a loosely bound cluster at same threshold cohesion. Therefore we cannot decrease

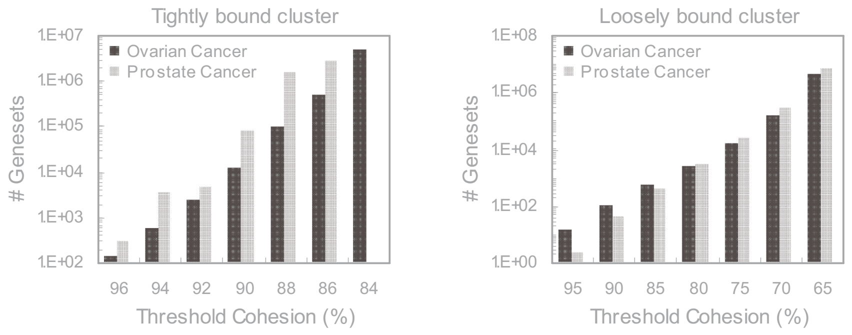


Fig. 7. Count of genesets under varying threshold cohesion

threshold cohesion much for a tightly bound cluster as we can for a loosely bound one. The most interesting part is that if we replace cohesion by support measure, not a single pattern or rule is generated at corresponding threshold support. Thus, at p% threshold cohesion, all patterns and rules that are generated have support below p%. To capture those patterns using support-based techniques, threshold support has to be set to such a low value that algorithms will take unaffordable space-time. If at all they are successful, the rule miner process will be burdened with millions of patterns. Further, there will be a big question on acceptability of a discovered rule. So, efficient post-processing strategies will be indispensable. As cohesion has a correspondence with coexistence of genes, the rules it produces, have high acceptability.

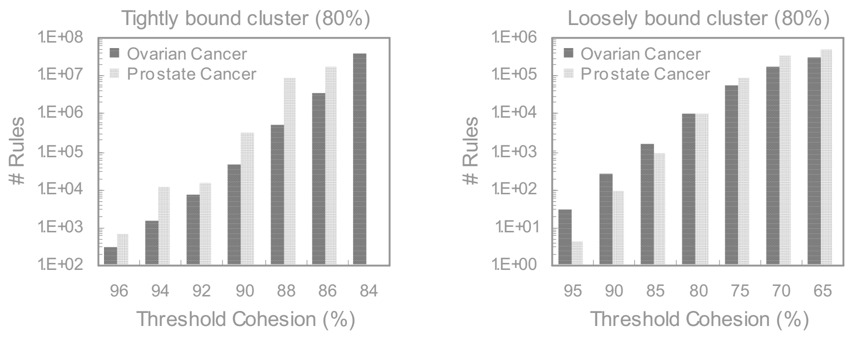


Fig. 8. Count of rules under varying threshold cohesion

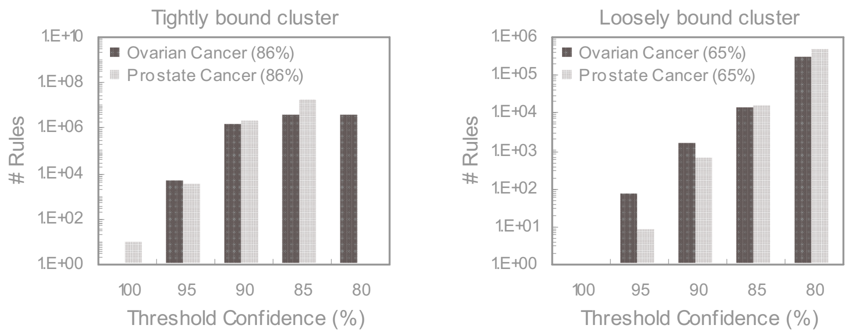


Fig. 9. Count of rules under varying threshold confidence

Finally, Fig. 9 depicts count of rules under varying threshold confidence, cohesion value is indicated within bracket. Again we see that a tightly bound cluster generates much more rules than a loosely bound cluster at particular threshold confidence. Still, a loosely bound cluster gives many associations at 85% to 90% threshold. The idea of cohesive clustering allows them to be discovered.

5 Conclusion and Future Plan

The present work addresses the problem of discovering important gene relationships in high-dimensional microarray data. Traditional support-based algorithms suffer from two foremost drawbacks. First, they overlook confident but infrequent associations. Second, in order to explore them, mining at very low threshold support either fails or generates too many associations that finding the right set becomes another difficulty. We propose a new clustering concept and a cohesion-based association mining technique. Our method creates scope for plentiful confident associations, both frequent and infrequent under classical measure, to be surfaced. Furthermore, high cohesion ensures that most of the associations are formed among genes that appear collectively rather being individual. This extends the scope of studying collective behavior of genes which is helpful in studies on gene function, metabolism and network. Deviations marked in collective properties in cases of diseased samples can play important role in biomedical research. Therefore cohesion-based associations have high acceptability in analyzing gene expression data.

In a subsequent work, we wish to employ the cohesion concept in developing an efficient closed pattern mining algorithm. This can be a novel idea in mining confident and reliable associations without being stuck into non-closed pattern explosion problem at early stage.

Acknowledgements

The authors would like to thank Dr. R N Bhattacharyya and the anonymous reviewers for their most helpful advice and comments.

References

1. Agarwal, R., Imielinski, T., Swami, A.: Mining Association Rules between Sets of Items in Large Databases. In: Proceedings of ACM Int'l Conf. on Management of Data (SIGMOD), Washington DC, May 1993, pp. 207–216 (1993)
2. Agarwal, R., Srikant, R.: Fast Algorithm for Mining Association Rules. In: Proceedings of 20th Very Large Database (VLDB) Conf., pp. 487–499 (1994)
3. Becquet, C., Blachon, S., Jeudy, B., Boulicaut, J., Gandrillon, O.: Strong association-rule mining for large scale gene expression data analysis: a case study on human SAGE data. Genome Biology 3(12), 1–16 (2002)
4. Cong, G., Tan, K.-L., Tung, A., Pan, F.: Mining frequent closed patterns in microarray data. In: Proceedings of IEEE Int'l Conf. on Data Mining (ICDM), vol. 4, pp. 363–366 (2004)
5. Cong, G., Tan, K.-L., Tung, A.K., Xu, X.: Mining TOP-K covering rule groups for gene expression data. In: Proceedings of ACM Int'l Conf. on Management of Data (SIGMOD), pp. 670–681 (2005)
6. Cong, G., Tung, A., Xu, X., Pan, F., Yang, J.: FARMER: Finding interesting rule groups in microarray datasets. In: Proceedings of ACM Int'l Conf. on Management of Data (SIGMOD), pp. 143–154 (2004)
7. Creighton, C., Hanash, S.: Mining gene expression database for association rules. Bioinformatics 19(1), 79–86 (2003)
8. Han, J., Pei, J., Yin, Y.: Mining frequent patterns without candidate generation. In: Proceedings of ACM Int'l Conf. on Management of Data (SIGMOD) (2000), pp. 1–12 (2000)
9. McIntosh, T., Chawla, S.: High Confidence Rule Mining for Microarray Analysis. ACM/IEEE Transactions on Computational Biology and Bioinformatics, `http://www.it.usyd.edu.au/~chawla/`
10. Pan, F., Cong, G., Tung, K., Yang, J., Zaki, M.J.: CARPENTER: Finding closed patterns in long biological datasets. In: Proceedings of ACM Int'l Conf. on Knowledge Discovery and Data Mining (SIGKDD), pp. 637–642 (2003)
11. Zaki, M., Scalable Algorithms, J.: for Association Rule Mining. IEEE Trans. On Knowledge and Data Engineering 12(3), 372–390 (2000)
12. Zaki, M.J., Gouda, C.: Fast vertical mining using diffsets. In: Proceedings of ACM Int'l Conf. on Knowledge Discovery and Data Mining (SIGKDD), Washington DC, pp. 326–335 (2003)

Identifying Subcellular Locations from Images of Unknown Resolution

Luís Pedro Coelho[1,2] and Robert F. Murphy[1,2,3]

[1] Lane Center for Computational Biology,
[2] Joint Carnegie Mellon University–University of Pittsburgh Ph.D. Program in Computational Biology,
[3] Departments of Biological Sciences, Biomedical Engineering, and Machine Learning, Carnegie Mellon University

Abstract. Our group has previously used machine learning techniques to develop computational systems to automatically analyse fluorescence microscope images and classify the location of the depicted protein. Based on this work, we developed a system, the Subcellular Location Image Finder (SLIF), which mines images from scientific journals for analysis.

For some of the images in journals, the system is able to automatically compute the pixel resolution (the physical space represented by each pixel), by identifying a scale bar and processing the caption text. However, scale bars are not always included. For those images, the pixel resolution is unknown. Blindly feeding these images into the classification pipeline results in unacceptably low accuracy.

We first describe methods that minimise the impact of this problem by training resolution-insensitive classifiers. We show that these techniques are of limited use as classifiers can only be made insensitive to resolutions which are similar to each other. We then approach the problem in a different way by trying to estimate the resolution automatically and processing the image based on this prediction. Testing on digitally down-sampled images shows that the combination of these two approaches gives classification results which are essentially as good as if the resolution had been known.

1 Introduction

Fluorescent microscopy is one of the methods of choice for determining the subcellular location of proteins. Methods for automatically analysing subcellular patterns in fluorescence microscope images have been extensively developed, allowing such determinations to be performed in a high-throughput, comprehensive manner (for reviews see [1,2]).

A very important property of a cell image is its pixel resolution (i.e., how big the space represented by a pixel is). This depends on the imaging approach used to collect the image (e.g., for widefield fluorescence microscopy using a digital camera, it depends on the magnification of the lens(es) used and the pixel spacing of the camera). Higher resolution images carry more information and detail. However, many considerations may lead to acquisition of images with lower resolution.

As an outgrowth of our location proteomics work, we have developed a system, the Subcellular Location Image Finder (SLIF), that analyses images (and their associated captions) from scientific publications [3,4]. This system identifies figure panels likely to contain fluorescence microscope images and then attempts to analyse the subcellular patterns within. Of course, these images vary widely in magnification, and are often annotated with a scale bar from which the pixel resolution can be inferred. This approach has two problems. First, it relies on successful identification of both the scale bar in the image and the caption text that describes its size. We have obtained acceptable, but far from perfect, results for this task [3]. A more fundamental problem is that not all images are so annotated, in which case the pixel resolution is unknown.

This work is focused on classifying images whose resolution is unknown. We start by simply ignoring the problem and feeding images of unknown resolution into the classification pipeline. We next consider approaches to estimating resolution from the images.

2 Methods

For the work described here, we used a publicly-available collection of two-dimensional images of HeLa cells previously obtained by our group using wide-field fluorescence microscopy [5]. It consists of immunofluorescence images of 9 proteins often used as markers for particular organelles or structures (one each for the endoplasmic reticulum, lysosomes, endosomes, mitochondria, the actin cytoskeleton, the tubulin cytoskeleton, and nucleoli, and two for the Golgi complex) as well as parallel images of a DNA-binding fluorescent probe to mark the nucleus. The pixel resolution of the images is $0.23\mu m$/pixel, and out-of-focus fluorescence in each image was estimated and removed using nearest neighbor deconvolution.

In order to investigate the effects of lowering the resolution, the images were digitally down-sampled. All data and software used in this paper are available from http://murphylab.web.cmu.edu/software.

2.1 Processing Images of Unknown Resolution

The first approach we used was to make the system insensitive to resolution, either by using features that are insensitive to image resolution or by training classifiers on examples from different resolutions so that they are able to classify any incoming image. Our group has previously pursued this line of reasoning with some success [6].

Some features can be designed in such a way as to make them roughly independent of resolution (e.g., SLF7.5, the ratio of the largest to smallest object in an image, which, discounting quantization effects, has the same value after image resampling). However, some informative features cannot be transformed so that they become resolution independent. Haralick texture features [7], for example, are both very informative and resolution dependent.

Whether a classifier can be trained to handle multiple resolutions is an empirical question. We measured how gracefully a classifier degrades when tested outside its training resolution and found that its performance drops very fast as the difference between the training and the testing resolutions increases. For example, a classifier trained on images with a high resolution of $0.23\mu m$/pixel achieves only 45% accuracy when classifying images at resolution $1.15\mu m$/pixel. Comparatively, a classifier trained at that resolution can achieve 83%. Thus, ignoring the issue is not a viable procedure.

An alternative to building a resolution-independent classifier is to train it on multiple resolutions by including, for each image in the training set, several down-sampled copies of it. This approach showed better results (data not shown). For classifiers trained in a small set of nearby resolutions, no accuracy is lost when classifying images in any of those resolutions. In fact, there seems to be a small boost from training with multiple copies of the same image, as previously reported [3].

This approach, however, scales badly to a large set of resolutions. When training on resolutions which are very different, there is a performance cost. For example, a classifier trained on images at both 0.23 and $3.68\mu m$/pixel has only 71% accuracy on the low resolution images, while a classifier trained only on those images obtains 79%. The classifiers thus obtained also degrade poorly to resolutions which were not part of their training sets. Furthermore, the increase in size of the problem has huge computational costs (training a classifier goes from minutes to several hours).

2.2 Inferring Resolution

We propose a different approach for handling images of an unknown resolution: infer the resolution, based only on the image. We shall see that this complements the approach above.

If one was approaching this problem manually, without fast computers, one could start by counting how many pixels wide the nucleus of the cell appears to be. Given the knowledge that a real cell has a nucleus of around 20 μm, one can obtain an estimate of how large a pixel is. This idea underlies the approach we outline below.

For predicting resolution, we therefore define numerical features which attempt to capture the size of the nucleus. We start by thresholding the image by retaining only the pixels that are above average. To remove small objects, which are likely to be noise, we smooth the binarized image with a majority filter (implemented in Matlab by the *bwmorph* function). Finally, using the Matlab function *convhull*, we compute the convex hull of the resulting binary image. On the basis of this, we compute:

1. The number of pixels in the hull (the area).
2. The square root of the hull area.

3. Its perimeter (measured as the number of pixels on the edge).
4. Number of pixels across its semi-major and semi-minor axes. This was calculated as illustrated by Prokop and Reeves [8].

We also include the inverse of all of these features as the resolution scales linearly with the inverse of the size. These features can be calculated on either the protein channel or on the DNA channel, if available. We refer to this feature set as SLF28 if calculated on the protein image and SLF29 if calculated (separately) on the protein and DNA channels.

Algorithms. Starting with the description of the image given by the features, we attempt to predict the resolution. There are two possible ways to handle this as a machine learning problem: *classification* (by deciding on a few representative classes, for example) or *regression*. Our initial trials in using classification showed that, for a large number of classes, the learning algorithms took too long to converge. Thus, we focus on regression, namely linear regression. Regression parameters were learned by minimizing the squared error.

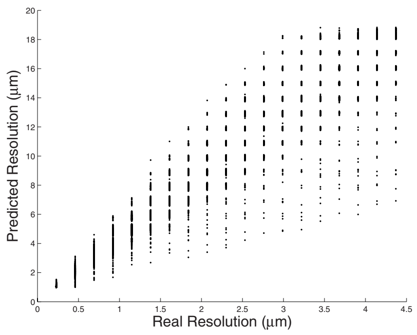
Initial tests revealed two properties:

– The range of of the training set is important. A set of parameters learned on the downsampling values $(1, 2, 3, 4, 5, 6)$ will do very well on test images of those values, but performance downgrades extremely fast outside of it (e.g., an image down-sampled by a factor of 10 will often be predicted to have been down-sampled by 15). On the other hand, inclusion of intermediate values is not as important as the range (i.e., training on $(1, 2, 3, 4, 5, 6)$ will do as well at handling images down-sampled by 3, a class in the training set, as training on $(1, 2, 4, 6)$ which does not include it).
– Breadth of training data (i.e., the difference between the largest and smallest resolution in the set) has a negative effect on accuracy.

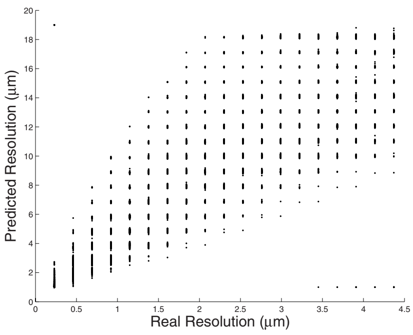
This suggested a iterated regression scheme. We call $\text{Estimate}(i; \boldsymbol{r})$ the prediction for the resolution of image i given by the estimator trained on the set of resolutions $\boldsymbol{r}$. To process an incoming image, we first predict its accuracy on the whole range of values ($p_1 = \text{Estimate}(i; 1, 2, \ldots, N)$). A second prediction is defined by $p_2 = \text{Estimate}(i; p_1 - 2, p_1 - 1, p_1, p_1 + 1, p_1 + 2)$), where the first prediction is used to lookup the correct parameters for a refined prediction. Finally, we output the value $\hat{p} = \text{Estimate}(i; p_2 - 1, p_2, p_2 + 1)$.

2.3 Evaluation of Resolution Prediction Schemes

Figure 1(a) shows the results for predictions made using both the DNA and protein channels for each image. As we can see, the error is very small for high resolution images, but increases for low resolution ones. This is explained by quantization effects. Even if the nucleus always measured a perfect $20\mu m$, this translates to 87 pixels at $0.23\mu m$/pixel resolution, which can clearly be told apart from the 43 pixels it takes when the images are down-sampled by 2 to $0.46\mu m$/pixel resolution. However, at resolutions lower than $4\mu m$/pixel, a $20\mu m$



(a) With DNA channel



(b) Without DNA channel

Fig. 1. Resolution Inference Results. Each dot is the result obtained for one test image, on the *x-axis* the original resolution is shown, on the *y-axis*, we show the output of the system. The results have not been rounded. A perfect result would be on the diagonal.

object takes only 4 pixels. If we consider that the nucleus size has some variation itself, it becomes clear that low resolutions cannot be told apart, even in principle.

When no DNA channel is available, one expects the variation in the size of the hull to be much larger (e.g., in the case of f-actin, the hull will probably contain the whole cell, while a DNA tag will only show the nucleus). To test this, we measured the coefficient of variation (the observed standard deviation divided by the standard mean, expressed as a percentage) of our measured features.

Table 1. Coefficient of Variation. The coefficient of variation of the features introduced in this work, when calculated on the DNA and protein channels.

	$\frac{\sigma}{\mu}$ DNA	$\frac{\sigma}{\mu}$ protein
Area	27%	126%
sqrt(Area)	13%	62%
Perimeter	14%	63%
Semi-Major Axis	16%	64%
Semi-Minor Axis	17%	66%

As Table 1 makes clear, the variation in features calculated on the protein channel is much greater than that calculated on the DNA channel. This explains why the results of inferring resolution based on the protein channel, presented on Figure 1(b), are not as good as those obtained using the DNA channel. We tested introducing SLF7DNA features into the regression model, followed by feature selection with stepwise discriminant analysis and regression on this set of variables. This brought about a small improvement in the results, but not enough to match the results with DNA features.

Looking at whether the iterated linear regression scheme makes a difference, one finds that the errors at the second level are lower than those at the first level, while the third level brings only a very minor improvement.

We tested the effects of removing half the resolutions used for training, while still testing on every resolution. This procedure simulates a situation where the image resolution was not in the training set. Results show that there is no accuracy penalty for this (data not shown).

2.4 Classification Pipeline

In the context of our work, the final goal of image processing is the classification output and the system must be evaluated on its accuracy there. First, we bring together the elements described above into an integrated classification pipeline.

For each image in our training set, we generated copies of it at lowered resolutions. We trained a classifier for each downsampling level, but included images from the level above and below it (to make it partially insensitive to resolution changes). To process an image, we estimate its resolution, and classify it using the classifier that was trained centred on the estimated resolution.

In order to evaluate our results, given that we expect classification accuracy to decrease with resolution due to lowered image quality, we compared our system

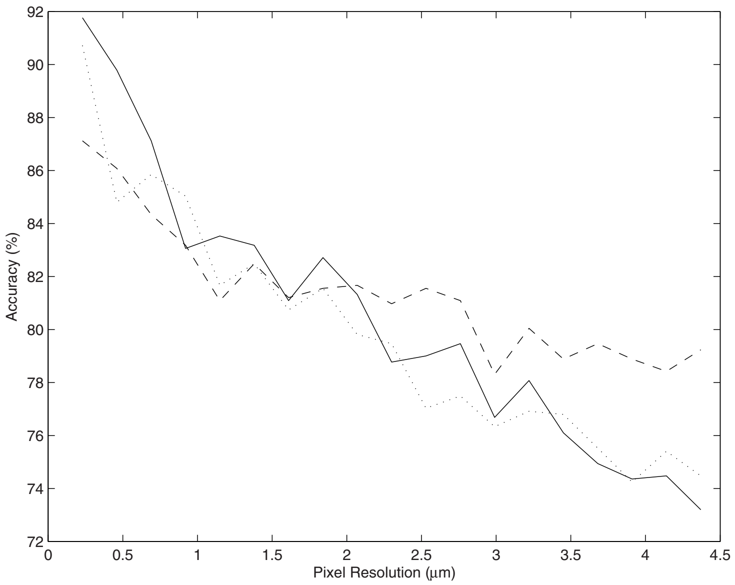


Fig. 2. Final Accuracy Results. The solid line shows the accuracy obtained in the case where the images were processed with resolution known. The dashed line shows the accuracy obtained when the resolution is inferred from the image, using only the protein channel, and this estimate is used for further processing. The dotted line shows the accuracy obtained when the resolution estimate is based on the DNA channel.

against a baseline where the image resolution is known. Figure 2 shows this comparison. We conclude that our system is capable of overcoming the unknown resolution problem.

3 Discussion

The SLIF system (which analyses images from published scientific journals) developed by our group needs to process images of differing and often unknown resolutions. Thus, we needed to adapt the image processing pipeline to handle such images.

We tackled the problem of handling images of unknown resolution by predicting it. Our solution was based on the calculation of simple features, which tried to capture the size of the nucleus, when a DNA channel was present, or the size of the cell, when it was not. These features were used for iterated linear regression. The resulting estimate predicted the resolution very well, if the DNA channel was available as the nucleus provides a known reference point in each cell. On the basis of only the protein channel, the prediction error increases by around two-fold.

For integration into the SLIF system, where images often contain multiple cells, the images will have to be segmented as a preprocessing step. In the case where a DNA channel is available (usually represented by one of the image's color channels), segmentation is easier as nuclei tend to be separable. Since the whole image is at the same resolution, results from different cells can be averaged together to obtain the final prediction.

Acknowledgments

This work was supported in part by NIH grant R01 GM078622. Facilities and infrastructure support were provided by NSF grant EF-0331657, NIH National Technology Center for Networks and Pathways grant U54 RR022241, and by NIH National Center for Biomedical Computing grant National U54 DA021519.

Luís Pedro Coelho was partially supported by a fellowship from the Fulbright Program and by the Portuguese Science and Technology Foundation (fellowship SFRH/BD/37535/2007).

References

1. Chen, X., Velliste, M., Murphy, R.: Automated interpretation of subcellular patterns in fluorescence microscope images for location proteomics. Cytometry 69 A, 631–640 (2006)
2. Glory, E., Murphy, R.: Automated Subcellular Location Determination and High-Throughput Microscopy. Developmental Cell 12(1), 7–16 (2007)
3. Murphy, R.F., Velliste, M., Yao, J., Porreca, G.: Searching online journals for fluorescence microscope images depicting protein subcellular location patterns. In: BIBE 2001: Proceedings of the 2nd IEEE International Symposium on Bioinformatics and Bioengineering, Washington, DC, USA, pp. 119–128. IEEE Computer Society, Los Alamitos (2001)

4. Murphy, R.F., Kou, Z., Hua, J., Joffe, M., Cohen, W.W.: Extracting and structuring subcellular location information from on-line journal articles: The subcellular location image finder. In: IASTED International Conference on Knowledge Sharing and Collaborative Engineering, pp. 109–114 (2004)
5. Boland, M.V., Murphy, R.F.: A neural network classifier capable of recognizing the patterns of all major subcellular structures in fluorescence microscope images of HeLa cells. Bioinformatics 17(12), 1213–1223 (2001)
6. Chen, X., Murphy, R.: Interpretation of Protein Subcellular Location Patterns in 3D Images Across Cell Types and Resolutions. In: Lecture Notes in Computer Science, pp. 328–342. Springer, Heidelberg (2007)
7. Haralick, R.M.: Statistical and structural approaches to texture. Proceedings of the IEEE 67, 786–804 (1979)
8. Prokop, R.J., Reeves, A.P.: A survey of moment-based techniques for unoccluded object representation and recognition. CVGIP: Graph. Models Image Process. 54(5), 438–460 (1992)

E-BioFlow: Different Perspectives on Scientific Workflows

Ingo Wassink[1], Han Rauwerda[2], Paul van der Vet[1], Timo Breit[2], and Anton Nijholt[1]

[1] Human Media Interaction Group, University of Twente
{i.wassink, p.e.vandervet, a.nijholt}@ewi.utwente.nl
[2] Micro Array Department, University of Amsterdam
{rauwerda, breit}@science.uva.nl

Abstract. We introduce a new type of workflow design system called e-BioFlow and illustrate it by means of a simple sequence alignment workflow. E-BioFlow, intended to model advanced scientific workflows, enables the user to model a workflow from three different but strongly coupled perspectives: the control flow perspective, the data flow perspective, and the resource perspective. All three perspectives are of equal importance, but workflow designers from different domains prefer different perspectives as entry points for their design, and a single workflow designer may prefer different perspectives in different stages of workflow design. Each perspective provides its own type of information, visualisation and support for validation. Combining these three perspectives in a single application provides a new and flexible way of modelling workflows.

1 Introduction

Workflow systems have proven to be successful for modelling both business processes [1,2] and life science experiments [3,4,5,6,7,8,9]. The way workflow models are used in these two areas differs tremendously. Business workflow models are control flow-oriented (defining the order in which work has to be done), whereas life science workflow models are data flow-oriented (describing the information flow between tasks) [10]. However, there is growing demand for a more controlled approach to workflows in the life science domain [11]. Workflow systems in this domain lack the facilities to model advanced control structures such as conditional branches and iteration, and miss the functionality to easily switch between resources (also known as agents) for executing the tasks of workflow models. For example, we have developed the RShell plugin [1] for Taverna, which is now part of the standard Taverna distribution [6,12,13]. This plugin provides the RShell processor for executing R-scripts in a Taverna workflow. Our plugin requires a local or remote installation of R [14] in combination with RServe [15], which turns R into an R server. However, such a local installation of R in combination

[1] http://www.ewi.utwente.nl/ biorange/rshell

with the required R packages are not always available. Therefore, the specification of the location of the R server should be postponed until the workflow is actually executed.

Workflow systems support two tasks: workflow design and workflow enactment (as workflow execution is often called in the literature). The result of workflow design is a workflow model or workflow for short. The result of workflow enactment is actual execution of the model in the correct way. Design and enactment can in principle be performed by different systems as long as there is a common language to transfer the workflow model from the design system to the enactment system. In practice, most systems support both tasks.

Scientists have to show that their experiment conforms to quality standards in their field [16] whereas business modelers have to create consistent, optimized, and possibly automated business processes [1,17,18]. When workflows become complex, we need tools to manage this complexity, for example, to perform automatic validation of workflows. Workflow models enable formal checking from a *control flow perspective* (soundness [19]) as well as from a *data flow perspective* [20] and *resource perspective* [17,18]. From a control flow-perspective, a suitable tool can also simulate the workflow to give the designer an idea of the course of events and the flow of data through the system. Workflow models, and diagrams in general, are very suitable as visualisation and communication means [21,17,22]. However, most workflow design systems mainly focus on structuring and connecting tasks; the visualisation aspect often gets little attention. Workflow visualisation is not limited to showing dependency relations between tasks, but can also show the types of data that flow between the tasks and what types of resources are able to perform the tasks. In existing systems, these different aspects (if supported at all) are often combined in a single diagram which results in cluttering of information [23]. This is counterproductive; if anything, visualisation should advance rather than hinder understanding.

We introduce *e-BioFlow*, a workflow design system that relies on an existing system to have the workflow enacted. It is available under the GNU General Public Licence through sourceforge[2]. E-BioFlow enables the user to model workflows from the three mentioned perspectives previously. E-BioFlow is inspired by the context of scientific collaborative environments, such as the e-BioLab [24]. The workflow system enables scientists to describe tasks in multidisciplinary life science experiments [25]. Workflow models for these types of experiments need to be flexible with respect to resources, which can be web services, scientists or machines in the laboratory. The models made by means of E-BioFlow can be enacted by the open-source workflow system Yawl [26]. In the next section, we will discuss the requirements of a workflow design system for modelling processes. After that, we will introduce our approach, e-BioFlow, the three perspectives it provides, and how these perspectives are related. The benefit of these perspectives will be illustrated using an example of a workflow that performs a simple sequence alignment. Our approach will be compared to related work and we will end with a discussion.

[2] http://sf.net/projects/e-bio-flow

2 Perspectives of a Workflow Model

Workflow models are often not designed for a single case but describe types of cases [18]. Each case is unique; cases can differ in the way tasks are executed, the data that flows between these tasks, but also the resources that execute the tasks. Therefore, a workflow model should provide a perfect balance between generalisation over the case type and adaptation to the specific cases.

Van der Aalst [26] and Jablonski and Bussler [17] distinguish three perspectives on workflows:

Control flow perspective: Tasks can seldom if ever be performed in an arbitrary order. The control flow perspective defines dependencies between tasks and the way tasks will be executed (sequential, parallel, conditional or iterative).

Data flow perspective: Tasks can consume and produce information. The data flow perspective defines these producer/consumer relations between tasks.

Resource perspective: Tasks can often be executed by a class of resources rather than by a single resource. The resource perspective defines the relation between tasks and the classes of resources that can execute them.

As we will explain later, these three perspectives are not orthogonal but interact and therefore deserve equal attention in a workflow design tool. Most workflow design systems, however, are either control flow-oriented or data flow-oriented [27]. Control flow-oriented workflows neglect the data flowing between tasks or only support them indirectly using task and net variables. Data flow-oriented workflows lack the presence of advanced control flow structures, such as conditional branching and loops. Workflow design systems that focus on both the control flow and data flow perspective are called hybrid workflow systems [27].

Ideally, a workflow can be reused for every instantiation of a certain case type. Therefore, it is important to abstract from resources and to delay resource-task binding until the workflow is enacted. E-BioFlow supports this and thus adds an important feature to the designer's toolkit.

3 E-BioFlow: A New Type of Workflow Design System

E-BioFlow is a visual workflow design system that provides all three perspectives to users for designing their workflows. The three perspectives are explicitly present in e-BioFlow by means of different tabs in the user interface. The workflow designer is able to work in a single perspective at a time without being restricted to the functionality of a single perspective. Changes in one perspective are propagated to the two other perspectives wherever appropriate.

Every workflow has at least two tasks, namely the start and the end task. These two tasks are used respectively to provide the workflow's input data and to collect the workflow's output data. Hierarchy is a very important property of workflow models, because it helps to structure large diagrams and provides a means for abstraction [19]. E-BioFlow supports hierarchical workflows; a task can be composite. We can choose to decompose a composite task into a sub-workflow, in which case the composite task is white boxed, but we can alternatively choose to ignore the way the composite task is structured, leaving it black boxed. The workflow specification is a container for workflows. However, in every container only one workflow is marked as the root workflow; the other workflows are decompositions of composite tasks. Two or more composite tasks can be white boxed to the same sub-workflow.

The concepts used in e-BioFlow to represent workflows are:

Task: A task is an abstraction of work to be done. This is also known as an activity [28].

Atomic task: An elementary representation of work [29].

Composite task: A task that can be black boxed or white boxed; in the latter case, we can also call it a sub-workflow.

Workflow: A workflow defines a set of tasks, the dependencies between the tasks, the data that flows among the tasks and the required capabilities to execute the tasks.

Specification: A specification is a container for workflow models. One of the workflows is the root model, the others are sub-workflows.

Dependency (Control Flow perspective): A relation between two tasks that defines enactment order: a certain task cannot start until another task has finished [27].

Dependency condition (Control Flow perspective): Every task has a start condition and an end condition describing, respectively, the way the task depends on prior tasks and the way it should activate next tasks [26].

Port (Data Flow perspective): A task can have multiple input and output ports for consuming and producing data, respectively. Ports are also known as parameters [23].

Object type (Data Flow perspective): The object type describes the type of information an input port accepts and an output port delivers.

Pipe (Data Flow perspective): A pipe defines a data dependency between two tasks, where data produced by the prior task is consumed by the next task [27].

Role (Resource perspective): A role describes the required capabilities to execute a task [28].

Actor (Resource perspective): An actor is a resource capable to fulfil a particular role and therefore to perform a certain class of tasks [28].

3.1 Three Perspectives to Design a Workflow

The e-BioFlow language extends the Yawl workflow language [26]. Yawl is a formal workflow language based on the Petri net formalism. This formalism, originally introduced for representing concurrent processes, provides powerful analysis techniques to validate workflow [19]. Yawl enables one to model almost all workflow patterns described by Van der Aalst and others [30]. Workflows designed in e-BioFlow can be enacted by the Yawl system [29]. The Yawl system itself also comes with a design tool that, however, only enables one to design control flow structures and does not explicitly consider the data flow and resource perspectives. E-BioFlow complements Yawl by adding these two perspectives. Next we will describe the three perspectives as they are offered to the designer by e-BioFlow. The use of these perspective will be illustrated using a simple life science case in section 3.2.

Control flow perspective. Due to dependencies between tasks, there is often a specific order in which tasks can be executed. The control flow perspective describes these dependencies. Tasks can be executed in sequential, parallel, conditional and iterative order [18]. The way tasks depend on each other is described using conditions. The control flow perspective visualises both the dependencies between the tasks and the conditions on these dependencies.

A task is visualised as a box. A dependency between two tasks is visualised as an arrow from one task to the next. Like in Yawl, the way a task depends on others is controlled by *join* and *split* types. The split type defines the way a task should activate next tasks. The join type defines the way in which a task has to wait for prior tasks. The join and split types are attached to, respectively, the left side and right side of the task's box. Van der Aalst et al. [26] distinguish four different types of splits: SINGLE (a task has only one next task to be activated), AND (a task should activate all next tasks), OR (a task should activate one or more of the next tasks), and XOR (only one of the next tasks should be activated).

Similar types of join exist: SINGLE (a task depends on just one prior task), AND (a task has to wait for all prior tasks to finish), OR (a task has to wait for one or more of the prior tasks to finish), and XOR (a task has to wait for one of the prior tasks to finish).

The symbols used in the Yawl language are confusing and not easy to remember. Therefore, we have defined our own symbols, which are presented in figure 1. The visualisation of the "Single" type contains a single line, which shows that only one connection is allowed. The "And", "Or" and the "Xor" splits and joins are represented by the first letters of their meanings: 'A', 'O' and 'X' respectively.

Data flow perspective. The data flow perspective is often seen in scientific workflow systems, where most tasks are executed by web services or computer applications. In a pure data flow representation, constructs such as loops are not included [27].

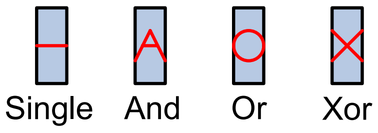


Fig. 1. Symbols representing the four different join and split types

As in the control flow perspective, tasks are visualised as boxes. The input and output ports are represented as small horizontal lines, distributed over respectively the left and the right borders of the task box. If the number of input ports or output ports becomes large, it can be difficult to distinguish the ports. Therefore, the size of a task box grows proportionally to the number of ports. The names of the ports become visible when the user moves the mouse cursor over the port. Both input and output ports are tagged with the attributes *object type* and *cardinality*. The object type describes the type of data a port can consume or produce. To support a wide range of object types, e-BioFlow provides an abstract Java interface for object types which can be adapted to object types for a specific domain. By defining a repository, these object types can be fed to e-BioFlow. For example, we are currently implementing support for the BioMOBY [31] data types. The cardinality defines the amount of items that can be produced or consumed. Two types of cardinality are supported: *UNIT* and *COLLECTION*. The first means that one item at a time is produced or consumed; the latter that a set of items can be produced or consumed. Pipes are visualised as arrows between the corresponding output port and input port. Using the object types and the cardinality, e-BioFlow prepares for full data compatibility checking. If a pipe is valid, it is coloured black and labelled corresponding to its object types, otherwise it is coloured red.

Resource perspective. E-BioFlow abstracts from resources by means of a ternary relationship between tasks, actors and roles. Actors are the real resources, such as web services. The role describes the abilities a resource is required to have to be able to perform the task [18]. Put simply, the role defines the *type* of service required. Roles can be played by different resources and resources can be able to play different roles, possibly at the same time [32,18]. If a resource plays a certain role, it acts as a contractor and it is responsible for the work it accepts. The loose coupling between task and actor makes a workflow model reusable, even if some actors are not available [8]. However, a role description should contain enough information to choose a suitable actor for playing the role and executing the task [33]. An actor is able to execute a task if and only if it is able to fill the role assigned to that task [18]. In this view, the enactment engine is responsible to perform the actor-role binding while the designer only specifies constraints on the binding by means of roles. To do this, the engine needs a mapping function $f_{RA} : (Role \rightarrow Actor)$ to select a suitable actor for a given role. The implementation of this function is domain-dependent. It can be based on, for example, a repository of the available actors, sorted by the type of service they can deliver.

The workflow is visualised as a graph in the resource perspective, too. The visualisation of the resource perspective is closely related to the control flow perspective, in order to keep the dependency relations in sight. However, the join and split condition information is left out. Each role is painted as a box around the task it is assigned to and contains the name of the role. Users can assign roles to tasks by dragging roles from a repository and dropping them on tasks.

In the current implementation, the resource perspective limits the user to only assign roles to atomic tasks on the ground that composite tasks will be expanded by the workflow enactment engine. An alternative would be to link every composite task to a role called "enactment engine" with the intended meaning that at execution the composite task is executed by the particular workflow enactment engine that happens to be running the parent workflow. The result would be a framework that encompasses both atomic and composite tasks. But it would also introduce a potential source of confusion, because for most atomic tasks a role entails a choice that will be made when the workflow is enacted. For composite tasks there is never such a choice: a composite task is always enacted in the framework of its parent task.

3.2 A Simple Life Science Case: Sequence Alignment

The three different perspectives will help the scientist to deal with the complexity of scientific workflows. Figure 2 shows the three different perspectives of a simple workflow for doing a sequence alignment. The workflow consists of a start and an end task, two tasks to collect the sequences and finally a task to actually perform the alignment. The control flow (Figure 2(a)) shows the order of the task execution. First, two sequences are collected and after that, an alignment is performed. When the alignment is performed, a next iteration of the sequence alignment is started or the end task is activated. The data flow (Figure 2(b)) shows only the data transfer between the tasks. The alignment task gets input sequences from both prior tasks; the end task gets the results of the alignment.

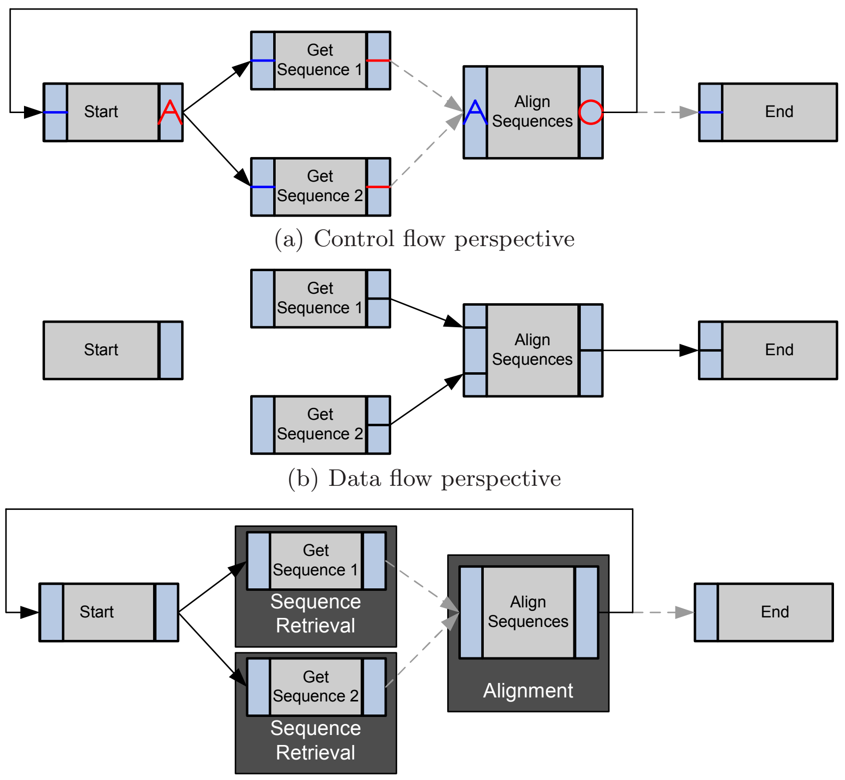
The roles of the tasks are shown in figure 2(c). The start and end task do not need roles, since these are used by the enactment engine to provide input data and collect output data. Both "Get Sequence" tasks require a sequence retrieval actor, such as an EBI retrieval service, so a sequence retrieval role is attached to each of the two tasks. An alignment role is attached to the "Align Sequence" task, for example by in-house software or again over a web service. The binding of the tasks to the actors will be done by the enactment engine.

Each perspective complements the other perspectives and shows only limited information about the workflow in order to keep the workflow diagram usable and comprehensible.

3.3 Dependencies among the Perspectives

In all perspectives, the tasks of a workflow are represented as vertices of the graph. To simplify switching between the perspectives, tasks positions and task sizes remain the same in all perspectives.

To illustrate the tight coupling between the perspectives, we will briefly discuss two scenarios. In one scenario, Figure 3(a), the designer has drawn two data pipes in the data flow perspective. E-BioFlow detects a dependency between the tasks involved, because in this example Align Sequences cannot start before the two "Get Sequence" tasks have delivered their data. Such a dependency is called an *inferred dependency* and it is inserted automatically in the control



(a) Control flow perspective

(b) Data flow perspective

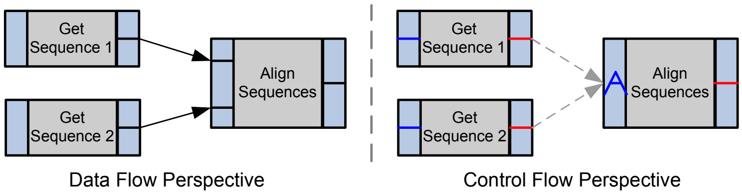
(c) Resource perspective

Fig. 2. An example of different perspectives for the workflow of a sequence alignment

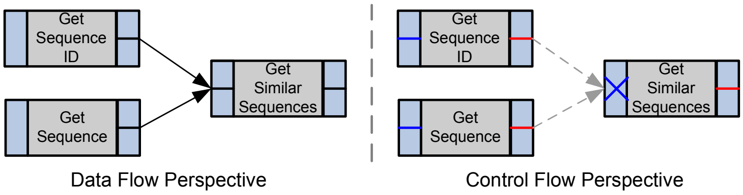
flow perspective as a dashed line. If the designer would later remove the data pipes, e-BioFlow automatically removes the dependencies in the control flow perspective.

In the other scenario, Figure 3(b), the designer has first inserted dependencies between the tasks in the control flow perspective. These are shown as solid lines. Later, the designer inserts data pipes in the data flow perspective. The solid lines in the control flow perspective are not affected because they are not inferred but inserted explicitly by the designer. For the same reason, if the designer later removes the data pipes, the dependencies in the control flow perspective are not removed by e-BioFlow.

Additionally, a relation exists between the resource perspective and the data flow perspective. The role definition depends more or less on the input and output types of a task, because not every actor can deal with all types of data. This means that the role description describes the ability to consume and to produce respectively the input and output data. Therefore, if an actor plays a role, it should be able to work with the input and output data [17]. The ability to



(a) The alignment task requires two inputs (left) and therefore it has to wait till both prior tasks have finished (right)



(b) The task that searches for similar sequences requires either a sequence id or a sequence as input (left) and therefore has to wait till one of the prior task has finished (right)

Fig. 3. Two examples showing the relationship between the data flow and the control flow perspective

check roles based on input and output types in BioMOBY fashion [31] is further work and requires more detailed descriptions of roles.

3.4 An Architectural View and Some Implementation Details

All three perspectives use, visualise and modify the same underlying workflow model. If this model is modified in a certain perspective, the other perspectives have to be notified to update their visualisations to reflect the change. Therefore, each perspective is registered to a software component called the *specification controller* (see figure 4). The specification controller works on top of the workflow specification. It has two main purposes.

First, it notifies all perspectives when the specification model is modified. These changes concern structural changes (i.e., a new task is inserted, a dependency is removed) as well as graphical changes (i.e., a task is repositioned, the zooming level is changed or the graph is repositioned using scrollbars).

Second, the specification controller is the only component that is allowed to modify the specification. If an action is performed in a certain perspective, then this perspective sends a request to the specification controller to execute this action. Normally, the specification controller executes the action and sends a notification event to all perspectives. The specification controller also takes care of the undo/redo history.

Using a central specification controller for all perspectives, it is easier to introduce new components working on top of the workflow model, such as checkers for each perspective. It is possible to integrate a workflow enactment engine in

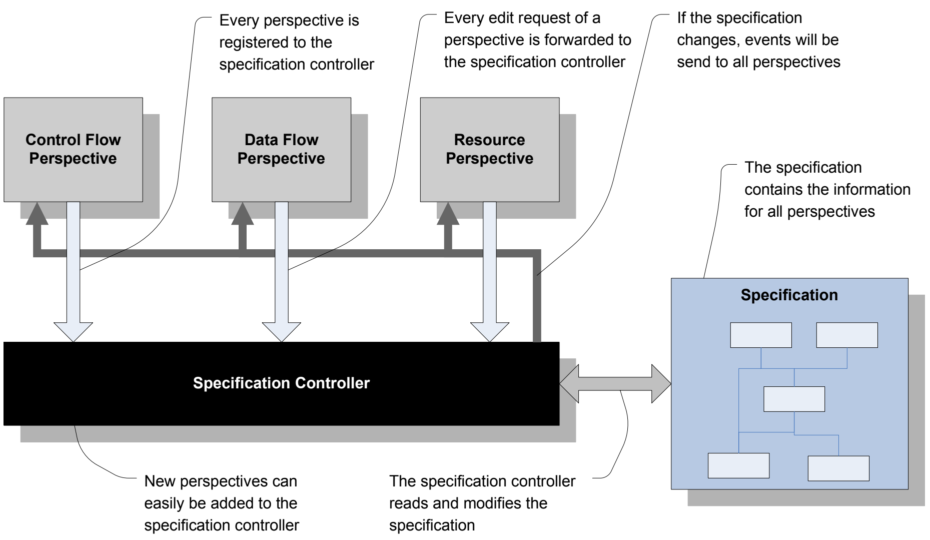

Every perspective is registered to the specification controller
Every edit request of a perspective is forwarded to the specification controller
If the specification changes, events will be send to all perspectives
Control Flow Perspective
Data Flow Perspective
Resource Perspective
The specification contains the information for all perspectives
Specification
Specification Controller
New perspectives can easily be added to the specification controller
The specification controller reads and modifies the specification

Fig. 4. The specification controller links the different perspectives to the specification

e-BioFlow. This will result in a workflow environment that can help scientists to design, execute, (partially) redesign and re-start the workflow, which is ongoing work.

E-BioFlow is implemented in Java[3] and uses the JGraph[4] graph package. The default implementation of e-BioFlow is supplied with a limited set of object types and roles. However, it is not restricted to this limited set, because both object types and roles are accessed from repositories, which can easily be extended or replaced. New (local and remote) repositories can be created by extending existing repositories or adapting the provided abstract Java interfaces for these repositories. E-BioFlow has its own file format for storing workflow specifications. The control flow and the data flow perspectives of the specifications can be exported to the Yawl enactment engine format to execute workflow. The data flow perspective can be mapped to Yawl by translating the data flows to task variables, net variables and XPath expressions. The Yawl enactment engine does not support late binding, which hnders the implementation of our ideas on the resource perspective. We are working on this to remove this obstacle. Furthermore, e-BioFlow is able to export the control flow perspective in XPDL format [34], which is maintained by the Workflow Management Coalition (WfMC). E-BioFlow can import Yawl and Scufl, the language of Taverna [6,35]. Due to the use of these central workflow formats, the workflow design system can be separated from the execution system [36].

[3] http://www.java.sun.com (last visited: 31-01-2008)
[4] http://www.jgraph.com (last visited: 31-01-2008)

4 Related Work

The focus in life science is on data. The traditional design interfaces of business workflow systems do not fulfil the requirements of this domain. Therefore, most scientific workflow systems use a data flow oriented language for representing workflows. However, the problem with most of these systems is that they have difficulties to support advanced control structures. For example in Discovery Net [3] , SCIRun [7], Taverna [6,12] and Knime [9] it is difficult if not impossible to construct advanced control structures, such as conditional branching and iterations. Triana [5] supports these control structures, however, like the other two tools, it has no ability to abstract from resources. From our point of view, this is cumbersome as resources, in particular web services, may be unavailable due to network errors, server overload, and similar problems. In Kepler [37,8], Bowers and Ludäscher [38] have tried to tackle this problem by defining primitives for actor replacements. However, the replacement is still done at design time instead of instantiation time.

E-BioFlow is a hybrid workflow system. Another hybrid workflow system is JOpera [23]. Its designers do not speak of a workflow system but rather of a web services composition engine. For modern workflows, the distinction no longer matters because many resources are remote anyway. The convergence of the fields of workflow studies, web services composition, and scientific data processing is one of the more exciting developments in the field today. Like e-BioFlow, JOpera emphasises the visualisation aspects of workflow design. It enables the user to design workflows in two interacting perspectives, namely control flow and data flow, and also supports automatic detection of inferred dependencies. One key difference between e-BioFlow and JOpera is the fact that JOpera is not based on a formal model. Another key difference is the way resources are treated. JOpera only supports late binding using special constructions whereas e-BioFlow supports late binding by default. Of course, this results in special needs of the workflow enactment system, but we believe it is better to use late binding in order to keep a workflow reusable for different but similar experiments.

5 Discussion and Future Work

In life science during the past decade, the importance of computer-supported or dry-lab experimentation has sharply increased. Workflow models support dry-lab experiments because controlling and managing huge volumes of data is cumbersome if not impossible without them. Workflow models are used to automate these experiments and to manage the huge amount of data collected and generated during these experiments [39].

In most scientific workflow systems, it is neither possible to model human tasks nor to model machine tasks; only web service tasks can be modelled. Modelling human tasks and machine tasks would make it possible to model both the wet-lab and the dry-lab parts of a scientific experiment. LIMSs, traditionally used for the wet-lab part of an experiment, are very often based on built-in (and thus inflexible) workflows [40]. Modelling the wet-lab and dry-lab parts of an experiment in a single framework carries several advantages, among which the presence of a unified model of the entire experiment that is of help in designing the experiment and is a tool for automating the lab journal when the experiment runs. E-BioFlow is prepared for the task of modelling entire experiments. The control flow and data flow perspective each provides its own type of validation, which makes it easier to find inconsistencies in the model. Validation in the resource perspective is currently not available. Such a validation could be based on the types of inputs and outputs of a task and the specification of the role, containing the types of data it can respectively consume and produce. However, more investigation is required to support a formal validation in the resource perspective.

We suggest a redesign of the way workflow engines currently operate. The enactment engine is not only responsible for triggering tasks to start execution, but also for task assignments to actors, based on role descriptions. Currently, we are integrating the Yawl workflow engine into e-BioFlow, to be able to enact workflows designed by e-BioFlow within the e-BioFlow application. Some modifications of the Yawl workflow engine are needed to support the late binding of actors to tasks. As a proof of concept, we will use BioMOBY [31] to create a life science problem solving environment. The data types defined in BioMOBY's data ontology can easily be translated to the object types used in e-BioFlow. The service types of BioMOBY will be translated to roles in e-BioFlow. The hierarchy of service types in BioMOBY is not fully mature. Therefore, further investigation is required to support automated role-base selection of BioMOBY services. Instead of being only applicable to a specific problem, workflow models designed using e-BioFlow become a general solution for a group of problems, independent of specific resources. A workflow model may thus become an ideal scaffold for a problem-solving environment [41].

Acknowledgement

We like to thank Matthijs Ooms for his work related to e-BioFlow.

This work was part of the BioRange programme of the Netherlands Bioinformatics Centre (NBIC), which is supported by a BSIK grant through the Netherlands Genomics Initiative (NGI).

References

1. Georgakopoulos, D., Hornick, M., Sheth, A.: An overview of workflow management: From process modeling to workflow automation infrastructure. Distributed and Parallel Databases 3(2), 119–153 (1995)
2. Santos, I., Valle, C., Raposo, A., Gattas, M.: A multimedia workflow-based collaborative engineering environment for oil & gas industry. In: Brown, J., Cai, Y. (eds.) Proceedings of the 2004 ACM SIGGRAPH, pp. 112–119. ACM, New York (2004)

3. Rowe, A., Kalaitzopoulos, D., Osmond, M., Ghanem, M., Guo, Y.: The discovery net system for high throughput bioinformatics. Bioinformatics 19(1), 225–231 (2003)
4. Addis, M., Ferris, J., Greenwood, M., Li, P., Marvin, D., Oinn, T., Wipat, A.: Experiences with e-Science workflow specification and enactment in bioinformatics. In: Cox, S. (ed.) e-Science All Hands Meeting 2003, Nottingham, United Kingdom, pp. 459–466 (2004)
5. Majithia, S., Shields, M., Taylor, I., Wang, I.: Triana: A graphical web service composition and execution toolkit. In: Jain, H., Liu, L. (eds.) IEEE International Conference on Web Services (ICWS 2004), San Diego, California, USA, pp. 514–521. IEEE Computer Society, Los Alamitos (2004)
6. Oinn, T., Addis, M., Ferris, J., Marvin, D., Senger, M., Greenwood, M., Carver, T., Glover, K., Pocock, M.R., Wipat, A., Li, P.: Taverna: a tool for the composition and enactment of bioinformatics workflows. Bioinformatics 20(17), 3045–3054 (2004)
7. Macleod, R., Weinstein, D., de St. Germain, J., Brooks, D., Johnson, C., Parker, S.: SCIRun/BioPSE: Integrated Problem Solving Environment for Bioelectric Field Problems and Visualization. In: Proceedings of the International Symposium on Biomedical Imaging, Arlington, VA, USA, April 2004, pp. 640–643 (2004)
8. Ludäscher, B., Altintas, I., Berkley, C., Jones, M., Lee, E., Tao, J., Zhao, Y.: Scientific workflow management and the Kepler system. Concurrency and Computation: Practice and Experience 18(10), 1039–1065 (2006)
9. Berthold, M., Cebron, N., Dill, F., Gabriel, T., Kötter, T., Meinl, T., Ohl, P., Sieb, C., Thiel, K., Wiswedel, B.: KNIME: The Konstanz information miner. In: Studies in Classification, Data Analysis, and Knowledge Organization (GfKL 2007), Springer, Heidelberg (to appear, 2007)
10. Wainer, J., Weske, M., Gottfried, V., Bauzer Medeiros, C.: Scientific workflow systems. In: Proceedings of the NSF Workshop on Workflow and Process Automation in InformationSystems, Athens, Georgia, pp. 1–5 (1997)
11. Kochut, K., Arnold, J., Sheth, A., Miller, J., Kraemer, E., Arpinar, B., Cardoso, J.: IntelliGEN: A distributed workflow system for discovering protein-protein interactions. Distributed and Parallel Databases 13(1), 43–72 (2003)
12. Oinn, T., Greenwood, M., Addis, M., Alpdemir, M., Ferris, J., Glover, K., Goble, C., Goderis, A., Hull, D., Marvin, D., Li, P., Lord, P., Pocock, M., Senger, M., Stevens, R., Wipat, A., Wroe, C.: Taverna: Lessons in creating a workflow environment for the life sciences. Concurrency and Computation: Practice and Experience Grid Workflow 18(10), 1067–1100 (2006)
13. Oinn, T., Li, P., Kell, D.B., Goble, C., Goderis, A., Greenwood, M., Hull, D., Stevens, R., Turi, D., Zhao, J.: Taverna/myGrid: Aligning a Workflow System with the Life Sciences Community. In: Workflows for e-Science, pp. 300–319. Springer, Heidelberg (2007)
14. Ihaka, R., Gentleman, R.: R: a language for data analysis and graphics. Journal of computational and graphical statistics 5(3), 399–414 (1996)
15. Urbanek, S.: Rserve – a fast way to provide r functionality to applications. In: Hornik, K., Leisch, F., Zeileis, A. (eds.) Proceedings of the 3rd International Workshop on Distributed Statistical Computing (DSC 2003), Vienna, Austria, pp. 20–22 (2003)
16. Barga, R., Gannon, D.: Scientific versus Business Workflows. In: Workflows for e-Science, pp. 258–275. Springer, Heidelberg (2007)
17. Jablonski, S., Bussler, C.: Workflow management. Modeling concepts, architecture and implementation. Modeling concepts. Thomson, London, UK (1996)

18. van der Aalst, W., van Hee, K.: Workflow management: models, methods, and systems. The MIT Press, Cambridge (2002)
19. van der Aalst, W.: The application of Petri nets to workflow management. The Journal of Circuits, Systems and Computers 8(1), 21–66 (1998)
20. Belhajjame, K., Embury, S., Paton, N.: On characterising and identifying mismatches in scientific workflows. In: Leser, U., Naumann, F., Eckman, B. (eds.) DILS 2006. LNCS (LNBI), vol. 4075, pp. 240–247. Springer, Heidelberg (2006)
21. Reisig, W.: A Primer in Petri Net Design. Springer, Berlin (1992)
22. Dourish, P.: Process descriptions as organisational accounting devices: the dual use of workflow technologies. In: Ellis, C., Zigurs, I. (eds.) Proceedings of the 2001 International ACM SIGGROUP Conference on Supporting Group Work, Boulder, Colorado, USA, vol. 60, pp. 52–60. ACM Press, New York (2001)
23. Pautasso, C., Alonso, G.: The JOpera visual composition language. Journal of Visual Languages and Computing 16(1–2), 119–152 (2005)
24. Rauwerda, H., Roos, M., Hertzberger, B., Breit, T.: The promise of a virtual lab in drug discovery. Drug Discovery Today 11(5-6), 228–236 (2006)
25. van der Vet, P., Kulyk, O., Wassink, I., Fikkert, F., Rauwerda, H., van Dijk, E., van der Veer, G., Breit, T., Nijholt, A.: Smart environments for collaborative design, implementation, and interpretation of scientific experiments. In: Huang, T., Nijholt, A., Pantic, M., Pentland, A. (eds.) Workshop on AI for Human Computing (AI4HC), Hyderabad, India, pp. 79–86 (2007)
26. van der Aalst, W., ter Hofstede, A.: YAWL: Yet another workflow language. Information Systems 30(4), 245–275 (2005)
27. Shields, M.: Control- Versus Data-Driven Workflows. In: Workflows for e-Science, pp. 258–275. Springer, Heidelberg (2007)
28. Barthelmess, P., Wainer, J.: Workflow systems: a few definitions and a few suggestions. In: Comstock, N., Ellis, C. (eds.) Proceedings of the Conference on Organizational Computing Systems (COOCS 1995), Milpitas, California, USA, pp. 138–147 (1995)
29. van der Aalst, W., Aldred, L., Dumas, M., ter Hofstede, A.: Design and implementation of the YAWL system. In: Persson, A., Stirna, J. (eds.) CAiSE 2004. LNCS, vol. 3084, pp. 142–159. Springer, Heidelberg (2004)
30. van der Aalst, W., ter Hofstede, A.: Workflow patterns: On the expressive power of (Petri-net-based) workflow languages. In: Jensen, K. (ed.) Fourth Workshop on the Practical Use of Coloured Petri Nets and CPN Tools (CPn 2002), Aarhus, Denmark, DAIMI, pp. 1–20 (2002)
31. Sowa, J.: Knowledge Representation: Logical, Philosophical, and Computational Foundations. Brooks Cole Publishing Co., Pacific Grove (1999)
32. Wroe, C., Goble, C., Greenwood, M., Lord, P., Miles, S., Papay, J., Payne, T., Moreau, L.: Automating experiments using semantic data on a bioinformatics grid. IEEE Intelligent Systems 19(1), 48–55 (2004)
33. Wilkinson, M.D., Links, M.: BioMOBY: an open source biological web services proposal. Brief Bioinform. 3(4), 331–341 (2002)
34. The Workflow Management Coalition: Workflow process definition interface – xml process definition language. Technical Report WFMC-TC-1025, The Workflow Management Coalition (2002)
35. Oinn, T., Addis, M., Ferris, J., Marvin, D., Greenwood, M., Goble, C., Wipat, A., Li, P., Carver, T.: Delivering web service coordination capability to users. In: Feldman, S., Uretsky, M. (eds.) Proceedings of the 13th international conference on World Wide Web, pp. 438–439. ACM Press, New York (2004)

36. Prior, C.: Workflow and Process Management. In: The Workflow Handbook 2003, pp. 17–25. Future Strategies Inc. (2003)
37. Altintas, I., Berkley, C., Jaeger, E., Ludäscher, B., Mock, S.: Kepler: An extensible system for design and execution of scientific workflows. In: Hatzopoulos, M., Manolopoulos, Y. (eds.) 16th International Conference on Scientific and Statistical Database Management (SSDBM 2004), Santorini Island, Greece, pp. 423–424 (2004)
38. Bowers, S., Ludäscher, B.: Actor-oriented design of scientific workflows. In: Delcambre, L.M.L., Kop, C., Mayr, H.C., Mylopoulos, J., Pastor, O. (eds.) Conceptual Modeling – ER 2005, Klagenfurt, Austria. LNCS, vol. 3716, pp. 369–384. Springer, Heidelberg (2005)
39. Greenwood, M., Goble, C., Stevens, R., Zhao, J., Addis, M., Marvin, D., Moreau, L., Oinn, T.: Provenance of e-Science experiments - experience from bioinformatics. In: Cox, S. (ed.) Proceedings of UK e-Science All Hands Meeting 2003, Nottingham, UK, pp. 223–226 (2003)
40. Reuss, T., Vossen, G., Weske, M.: Modeling samples processing in laboratory environments as scientific workflows. In: Wagner, R.R. (ed.) 8th International Workshop on Database and Expert Systems Applications (DEXA 1997), pp. 49–54. IEEE Computer Society, Los Alamitos (1997)
41. Gallopoulos, E., Houstis, E., Rice, J.: Computer as thinker/doer: Problem-solving environments for computational science. IEEE Computational Science and Engineering 1(2), 11–23 (1994)

Knowledge Acquisition Focused Cooperative Development of Bio-ontologies – A Case Study with BIO2Me

Dominic Mainz[1], Ingo Paulsen[1], Indra Mainz[1], Katrin Weller[1], Jochen Kohl[1], and Arndt von Haeseler[2]

[1] Heinrich-Heine-University, Universitaetsstr. 1, 40225, Duesseldorf, Germany
[2] Center for Integrative Bioinformatics Vienna, Max F. Perutz Laboratories, University of Vienna, Medical University of Vienna and Veterinary University of Vienna, Dr. Bohr-Gasse 9, 1030, Vienna, Austria
Phone: (+49 0211) 81 13716; Fax: (+49 0211) 81 15767
{mainzd, paulseni, weller, kohlj}@uni-duesseldorf.de,
mainzi@biophys.uni-duesseldorf.de,
arndt.von.haeseler@univie.ac.at

Abstract. We diagnose the need for collaborative ontology engineering approaches exemplified by the life sciences. With BIO2Me we present our case study, a structured knowledge base in the field of bioinformatics tools and methods. We point out the difficulties in knowledge elicitation and concerning the cooperation of domain experts and ontology designers we experienced during the engineering process of our ontology. Finally, we demonstrate our solutions for an ontology developing environment by discussing some aspects such as publication relevant tagging and semantically binding wiki pages with ontologies to integrate domain experts into the ontology development process.

1 Introduction

Among the different scientific disciplines, the life sciences have proved themselves as particularly interested in the use of ontologies as a means for formal knowledge representation and effective information integration. This is due to the vast amount of data produced in this field, as well as to inconsistent terminologies and definitions within this large research area, which make scientific exchange eminently difficult.

Ontologies are knowledge representation systems which are defined as formal conceptualizations of a knowledge domain [1]. They are used to represent domain knowledge in a machine-readable and unambiguous way. Ontologies as considered in this article are written in specialized languages. The currently most prominent and widely supported ontology language is OWL [2]. Ontologies in OWL consist of three basic types of different constructs: classes or concepts, properties or relations and individuals. Classes define a set of individuals that share relevant types of properties. Individuals asserted to a class are called instances of this class. For example, a concept within an ontology on bioinformatics tools could be 'Alignment_Program', instances could be an alignment program such as 'ClustalW2' [3], or even more precisely, a certain program version. For each instance we may add datatype properties that relate them with exact datatype values (for the ClustalW2 example that could be 'has_publication' value 'doi:10.1093/bioinformatics/btm404') and object properties which relate them to other instances within the ontology (e. g. 'reads_data'). Figure 1 displays the corresponding detail from a domain ontology for bioinformatics.

Ontologies are a key component for an increasingly diverse set of knowledge based applications in areas like bioinformatics. They provide global information infrastructures in terms of e-Science [4,5], the Semantic Web [6] and the Semantic Grid [7]. In these contexts ontologies are used to annotate content in digital libraries and archives, act as a composition layer for the integration of multiple digital resources and as knowledge base for information systems. They are also often intended to be a shared conceptualization of a domain of interest in terms of a consensual knowledge base for a community [1,8].

This article was inspired by ongoing work in the Ontoverse research project. The Ontoverse project develops a Web-based platform for collaborative ontology engineering and management in life sciences [9]. Here, we will concentrate on explaining the importance of collaboration between domain experts and ontology designers for ontology engineering in general and exemplified for the design and implementation of an application domain ontology for bioinformatics called BIO2Me (**B**io**I**nformatics **O**ntology for **To**ols and **Me**thods). We discuss conceptual and technical problems in providing experts' knowledge and present our ideas and current solutions. We have not developed an entirely new methodology for ontology engineering but have incorporated new and existing approaches into a system environment.

1.1 Bio-ontologies

In bioinformatics and life sciences in general ontologies are becoming increasingly important (e. g. [10,11,12]). Some of the currently most highly regarded ontologies are the Gene Ontology [13], the TAMBIS Ontology [14], the RiboWeb Ontology [15,16], The Ontology for Molecular Biology and EcoCyc [17]. To reduce redundancy and foster interoperability between the constantly accumulating amount of ontologies the OBO collection [18], an umbrella community for a range of ontologies designed for biomedical domains, has been established. The Gene Ontology (GO) is part of the OBO collection and was founded as a collaborative effort to address the need for consistent descriptions of gene products in different databases. With currently approximately 100 applications that use the GO, it is one of the most prominent indications of the growing importance of ontologies for the further development of bioinformatics.

The particular and still growing interest in ontologies within the life sciences can be ascribed to different reasons. Due to the exponential growth of data and publications researchers more and more face the challenge of data survey, aggregation and integration. This coerces the scientific community into finding

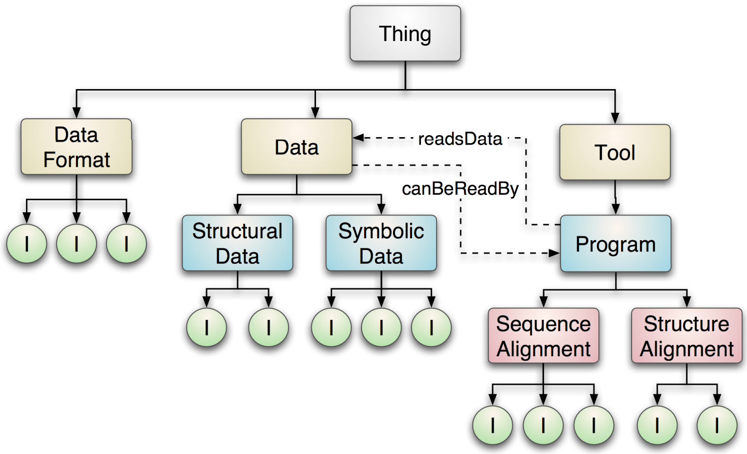


Fig. 1. Depiction of a detail of the BIO2Me ontology, with a basic hierarchical structure and additional interrelations. Instances (I) can be added to the concepts.

sophisticated structuring, accessibility, context information and information integration solutions. More than that, domains in the life sciences themselves are very complex. Concepts and classifications are subject to constant evolution [19] because new findings possibly entail new classifications. For example, the dogma of molecular biology (DNA $\rightarrow$ RNA $\rightarrow$ protein) was challenged by the discovery of the manifold functions (e. g. regulatory functions as transcription factors) of RNA. RNA does not only serve as messenger between DNA and protein anymore, but assumes specific functions in regulatory networks. Ontologies provide a potentiality for flexible knowledge modeling and are therefore well suited to overcome these challenges. Another big problem in the life sciences is the use of historical founded different vocabularies. This complicates collaboration between sub-disciplines, but also intra-disciplinary cooperations suffer from it. Term definitions may vary even for key concepts. A prominent example is the inconsistent definition of 'gene' used in different scientific databases [20]. On the other side, there exist e. g. several names for one species arosen from different scientific communities [21]. Both, the distinguishing of different views on the same concept and the unification of synonyms, have to be captured and formalized to enable more efficient collaboration. In addition, some domains seem to form ontological structures naturally: Biological objects are often linked to each other in various ways, "forming a highly interconnected graph of relationships" [19]. Furthermore, fields such as zoology uphold traditions in establishing taxonomies for a long time [11]. Ontologies can now be viewed as a new dimension in these ordering approaches.

BIO2Me – An Ontology for Bioinformatics. We have built an ontology in the domain of bioinformatics tools and methods, the **Bio**I**nformatics** **O**ntology for **To**ols and **Me**thods (BIO2Me) which currently contains about 400 classes

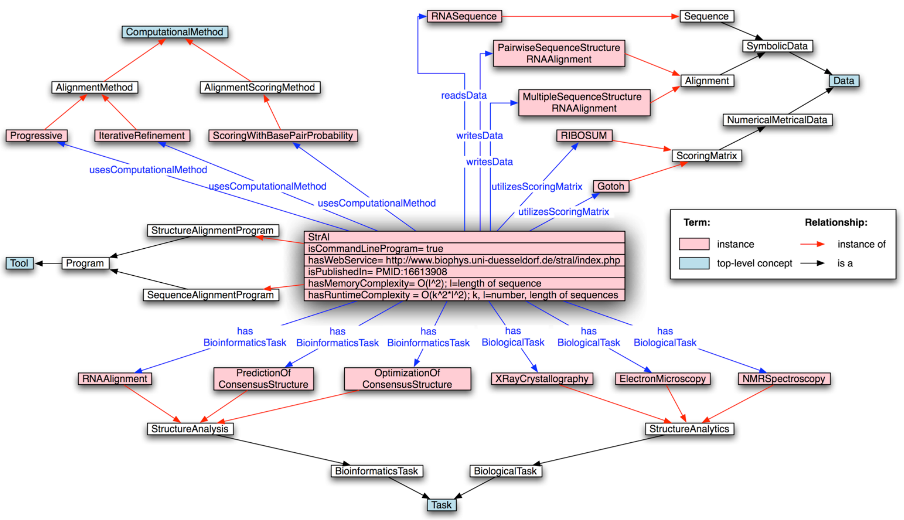
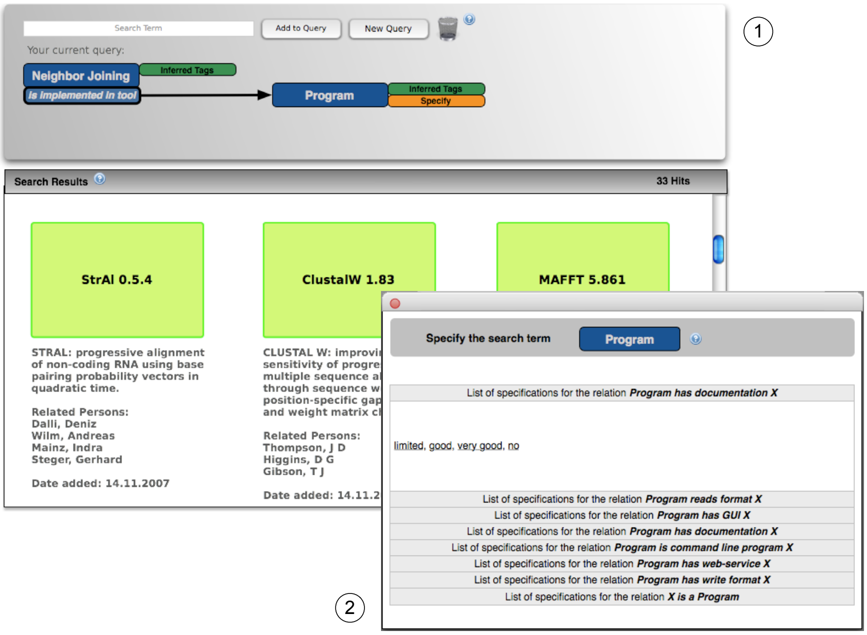


Fig. 2. Schematic depiction of a program's modeling in the BIO2Me ontology

and individuals. During this design process we determined deficiencies of available ontology editors in the support of collaborative ontology engineering and experienced (collaborative) ontology design problems in general (see section Collaborative Ontology Engineering).

The ontology was motivated by some earlier work from members of our current research team, where the performance of several alignment tools was compared with respect to selected input data [22]. This study showed that the most popular and mostly employed program in this field, ClustalW, provided not necessarily the best results for each data set. This pointed out a big challenge in bioinformatics: There is a variety of programs, packages, databases etc. dealing with various problems like the efficient processing of experimental data, sequence analysis and structure prediction and visualization. But these can currently not be surveyed with reasonable effort. Even for experts in a specific domain of bioinformatics it is sometimes hard to decide which tool fits the given requirements best. More than that, there is a plethora of programs which incorporate miscellaneous computational, mathematical, and biological approaches to solve even the same problems.

With BIO2Me we aim at collecting detailed information of bioinformatics tools in a structured way. The practical aim to make these tools easily accessible highly influences the actual structure of the ontology, the whole ontology conceptualization was focused on this practical applicability (sometimes dominating over strictly logical representations). Tools are categorized according to their application ranges (with bioinformatics perspective). Supported biological tasks, utilized computational methods, processed data formats and support information of tools are captured, too. With the help of a tool's users it is also possible to gather usage reports from their experiences. Figure 2 shows a



StrAl 0.5.4

STRAL: progressive alignment of non-coding RNA using base pairing probability vectors in quadratic time.

Related Persons:
Dalli, Deniz
Wilm, Andreas
Mainz, Indra
Steger, Gerhard

Date added: 14.11.2007

ClustalW 1.83

CLUSTAL W: improvi sensitivity of progre multiple sequence al through sequence w position-specific gap and weight matrix c

Related Persons:
Thompson, J D
Higgins, D G
Gibson, T J

Date added: 14.11.2

MAFFT 5.861

List of specifications for the relation ***Program has documentation X***

limited, good, very good, no

List of specifications for the relation ***Program reads format X***
List of specifications for the relation ***Program has GUI X***
List of specifications for the relation ***Program has documentation X***
List of specifications for the relation ***Program is command line program X***
List of specifications for the relation ***Program has web-service X***
List of specifications for the relation ***Program has write format X***
List of specifications for the relation ***X is a Program***

Fig. 3. Schematic illustration of a search result basing on BIO2Me. The results in the bottom of the depiction (1) can be narrowed by further specification of the search terms in the specification interface (2). The user can constitute features of 'Program' which are modeled relations in the ontology.

schematic illustration of the characterization of the program StrAl (Structure Alignment) [23]. One can see various features of the program which are modeled in the ontology.

The ontology will provide a basis to search for tools that meet the users needs. Moreover it will be able to offer additional information about certain tools and computational methods. It will answer questions like: "Which tools and methods exist, that deal with given problems?", "Which data output do they provide?" etc. Figure 3 depicts a possible search procedure: a user is looking for a program which utilizes the computational method 'Neighbor Joining'. In the bottom box current search results will be illustrated. To narrow down these results, the user can specify his search terms. Therefore a list of all existing relations (features) of the concept to specify is dynamically created from the ontology to let the user choose additional information about it.

A scientist's motivation to use the BIO2Me ontology is very diverse. A BIO2Me search can be helpful for a fast familiarization with a new research task in the field of bioinformatics and will help newcomers in the field to find relevant references quickly. They can get information about tools and how other scientists addressed a certain problem. On the other hand, a lot of tools developed in diplom or bachelor/master theses are not published although they provide good approaches which are worthwhile to pursue. Hence it will be useful to have easy tools that allow an implementation of such unpublished work in the ontology and therefore make also unpublished methods accessible to the scientific community. Furthermore, experimental biologists can use BIO2Me to find an adequate tool for their data analyses or for the planning phase of experiments. Even for bioinformaticians it is useful to get a review of available tools and to have a quick reference to differences between versions and tools, to publications and additional features. The information about input and output formats of a tool facilitates the pipeline of tools.

The search can only yield auxiliary results if the underlying ontology is substantial and is constantly enlarged and updated. This leads to a very important point: We are by far not able to provide the information of each bioinformatics tool on our own, but we are reliant on the cooperation of the tools' developers and users, the domain experts. We had to find a way to better support the involvement of domain experts because the currently available ontology engineering tools are insufficient for these purposes (see section Collaborative Ontology Engineering). Furthermore, there have to be experts who check the ontology periodically for its consistency, accuracy and correctness. One problem we faced during the building of BIO2Me was the acquisition of domain experts. Particularly in the beginning of an ontology project it could be hard to find participants which help to provide their knowledge. That is why we additionally developed a semi-automated methodology of ontology extension (see section Knowledge Acquisition Focused Ontology Editing – A Case Study with BIO2Me). This approach utilizes the tagging of publications basing on the concepts of the ontology. The user gets hints of text passages which could be relevant for the ontology.

Lessons learned from the construction of BIO2Me:

– The domain of BIO2Me eminently points out the need for collaborative ontology engineering. To represent bioinformatics tools with their applications, the whole bioinformatics research field and biology itself have to be displayed adequately in a structured way. Different fields of expert knowledge are needed to characterize different functions and application areas of bioinformatics tools. Furthermore, a vital community is needed to add information on the different tools which should be represented.
– In long term, the major challenge with BIO2Me will be to keep it up-to-date. It will be necessary to keep track of new developments in bioinformatics, e. g. as new tools or new versions of existing tools may be published.
– We realized the problem with recruiting domain experts who are willing to share their knowledge, so we have also developed an alternative way of collecting relevant background knowledge (in form of publications) for extending the ontology. Though this approach will never be able to replace the intervention and mental power of a domain expert.
– The basic challenge of this particular ontology was to define its basic structure. This is where the highest quality control is needed, because it is most difficult to change fundamental structures at a later point in time. It is also the part of the work which requires the most discussion and planning. Less

fundamental aspects, like adding new instances to existing concepts, can however easily and freely be handled by a large community.

2 Collaborative Ontology Engineering

Several approaches have been made to support collaborative ontology engineering e. g. in [24,25]. Our work combines the practical work of developing an ontology with the development of a supportive tool and working environment for this task. We started our ontology BIO2Me in a very clear team, wherein in general one member edited the formal ontology and the other members contributed their knowledge and discussed occurring problems in frequent meetings. This procedure was suitable because we were in close contact, but after inviting other domain experts, not members of the team, we faced the problem how to integrate them into the knowledge capturing process. We also had an issue of visualizing and preparing the information within the ontology so that even people not familiar with ontologies can take part in evaluation (i. e. its coverage of a particular domain and the richness, complexity and granularity of that coverage). Current ontology editors do not sufficiently handle these problems. In the next sections we explain our collaborative approach and the prototype system deriving from the mentioned problems.

We also communicated with the GO Consortium to learn from their experiences as during the last ten years a community has formed up to use and discuss GO [11]. Nearly 25,000 terms of GO are maintained by a core team of about ten GO experts. Only these few ontology designers are in charge of deciding which concepts are integrated in the ontology and in which way they are interlinked to others. Additionally, anyone may post requests for new terms and suggest changes via the GO Website (helpdesk, mailing lists). The community has very limited influence on the actual structure of the ontology. This example also shows a division of two levels of ontology collaborators that we will describe in the next section.

2.1 Cooperation between Domain Experts and Ontology Designers

Our focus in collaborative ontology engineering is, basing on our experiences, placed on the support of a heterogeneous community integrated in a social network closely combined with a Web-based ontology editor. Potential users differ in their fields of interest and skills: On the one hand knowledge and expertise is needed from domain experts (DEs). On the other hand ontology languages can only be fully exploited by ontology designers (ODs).

DEs, which are typically potential users of the ontology, are mainly responsible for evaluating an ontology from a domain perspective, but also regarding its understandability and actual advantage of use. They are involved in ontology development through their requests and by evaluation processes. ODs are mainly entrusted with the translation of domain knowledge into a formal ontological representation (or transforming proto-ontological into ontological data). This stage is established by a dialog with human experts in order to elicit knowledge. In the following stage the OD codes the knowledge explicitly in the ontology. This process iterates until the ontological coding is judged to be satisfactory by the DEs.

Aspects of cooperation in the context of ontology development can be divided into two different categories of collaboration:

1. Collaboration of ODs during knowledge formalization.
2. Cooperation between DEs and ODs for knowledge acquisition and ontology evaluation and quality control, respectively.

The presented BIO2Me case study focuses on the cooperation between DEs and ODs regarding workflow integrated tools for support. This paper presents our first preliminary results, further tests on usability and applicability are processed.

2.2 Knowledge Acquisition Focused Ontology Editing – A Case Study with BIO2Me

From Expert Knowledge to Ontologies. A principal challenge for ontology projects is the acquisition of the knowledge that has to be modeled. Besides textual information the most important knowledge resource are DEs. DEs combine explicit knowledge, which can be or has already been articulated in written form, with tacit knowledge. Even though tacit knowledge (aspects of knowledge that cannot be codified, but gained by personal experience and passed on through training) by definition is not directly formalizable it is in fact the foundation for the identification of ontology relevant concepts, instances and their relations. To harness DEs' knowledge more efficiently we adapted the common principals of wiki engines and tagging for collaborative knowledge acquisition.

Tagging with keywords [26] has become popular for social Web applications and is a very useful way of categorizing items that makes it easy for users to search and browse objects. Tag clouds display the most common tags as the largest, which makes a great starting point to allow people to discover objects on a website. Another usage for tags is to find related objects that share most of the same tags.

Figure 4 shows the publications' selecting and tagging process by DEs (in the case of BIO2Me bioinformaticians). The imported abstracts of the selected publications are mapped with constructs' labels of BIO2Me to identify candidates for ontology concept extension. Alternatively ontology extension can be achieved via expert knowledge on wiki pages, in which proposals are commented by DEs. Ontology population with new entities is also possible with experts' recommendation in wiki entries or directly by manually tagged publications.

In the following subsections we describe our experiences with tagging, context related abstract identification and an adapted wiki engine during the population of BIO2Me.

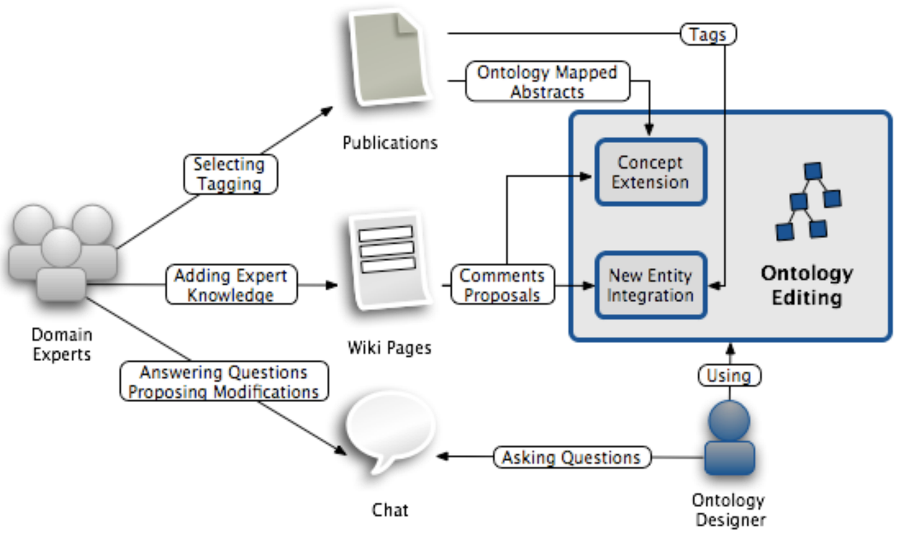
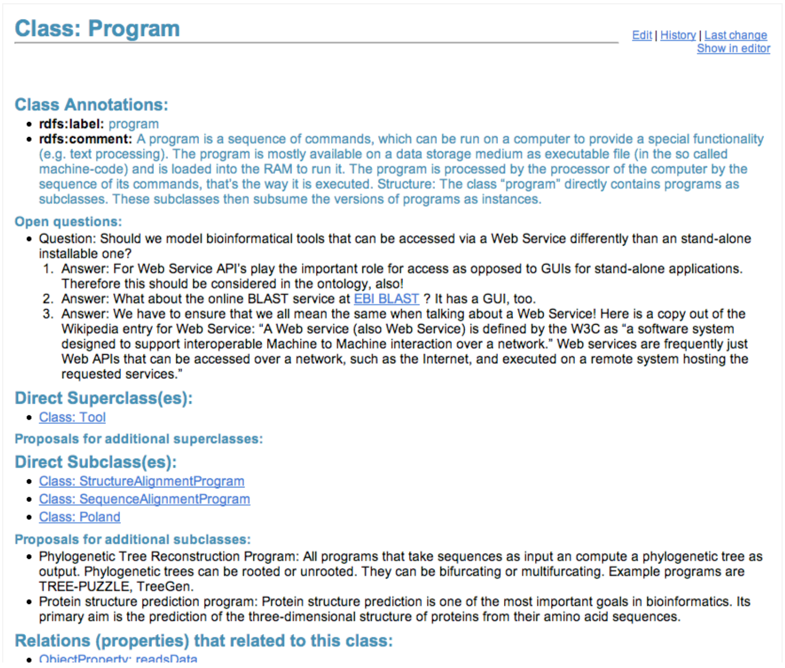


Fig. 4. From Expert Knowledge to Ontology Extension

A Wiki for the Knowledge Elicitation from Domain Experts. The building part and the connection to the other architectural parts of our system is an *ontology project*. Thus, BIO2Me would be maintained and developed within one project, but it is possible to plan and build other ontologies with the system, each within a specific ontology project. A project is characterized by a name and description, information about its members, the creation data together with the name of the founder, and of course the ontology itself. Every ontology project has its own collection of wiki pages, which allows all project members to create, edit, and display relevant articles through a Web inferface.

For new ontology projects the wiki provides predefined pages for the Ontology Requirements Specification Document (ORSD modified from [27]) and the informal collection of concepts and their relations [9]. ORSD captures information about the modeled domain (that is the field of knowledge, which is formalized in the ontology), the development goal, design criteria (like naming conventions), available knowledge sources and a competency questionnaire (a collection of questions, which should be answered by the ontology).

During the evolutionary development of an ontology our system automatically generates wiki pages for the different elements of the ontology which can be modified by the users of the system. The benefit of such an approach is twofold: 1) wiki pages are used to capture additional annotations about the ontology's constructs that would be hard to integrate in a user-friendly way into the graphical user interface of an ontology editor, 2) wiki pages can be used by the domain experts to add their knowledge to the development process. In this way we can ensure the separation of concern between the ontology designers who have to be able to edit the formal model of the ontology and the domain experts who can provide crucial information about the domain. Predefined sections help



Class: Program

Edit | History | Last change
Show in editor

Class Annotations:

- rdfs:label: program
- rdfs:comment: A program is a sequence of commands, which can be run on a computer to provide a special functionality (e.g. text processing). The program is mostly available on a data storage medium as executable file (in the so called machine-code) and is loaded into the RAM to run it. The program is processed by the processor of the computer by the sequence of its commands, that's the way it is executed. Structure: The class "program" directly contains programs as subclasses. These subclasses then subsume the versions of programs as instances.

Open questions:

- Question: Should we model bioinformatical tools that can be accessed via a Web Service differently than an stand-alone installable one?
  1. Answer: For Web Service API's play the important role for access as opposed to GUIs for stand-alone applications. Therefore this should be considered in the ontology, also!
  2. Answer: What about the online BLAST service at EBI BLAST ? It has a GUI, too.
  3. Answer: We have to ensure that we all mean the same when talking about a Web Service! Here is a copy out of the Wikipedia entry for Web Service: "A Web service (also Web Service) is defined by the W3C as "a software system designed to support interoperable Machine to Machine interaction over a network." Web services are frequently just Web APIs that can be accessed over a network, such as the Internet, and executed on a remote system hosting the requested services."

Direct Superclass(es):

- Class: Tool

Proposals for additional superclasses:

Direct Subclass(es):

- Class: StructureAlignmentProgram
- Class: SequenceAlignmentProgram
- Class: Poland

Proposals for additional subclasses:

- Phylogenetic Tree Reconstruction Program: All programs that take sequences as input an compute a phylogenetic tree as output. Phylogenetic trees can be rooted or unrooted. They can be bifurcating or multifurcating. Example programs are TREE-PUZZLE, TreeGen.
- Protein structure prediction program: Protein structure prediction is one of the most important goals in bioinformatics. Its primary aim is the prediction of the three-dimensional structure of proteins from their amino acid sequences.

Relations (properties) that related to this class:

- ObjectProperty: readsData

Fig. 5. The wiki page for the concept 'Program'. In this example two proposals for additional subclasses have been entered by DEs: 'Phylogenetic Tree Reconstruction Program' and 'Protein structure prediction program'.

DEs to get information about the modeled term and to enter additional expert knowledge. ODs can pose questions related to constructs in the corresponding wiki pages which can then be answered by DEs. In this way crucial modeling decisions can be discussed between DEs and ODs without the need to give DEs a deep insight into the schema of the ontology.

During BIO2Me extension it turned out that ontology versioning and maintenance, respectively, is a challenge that affects the knowledge capturing as the wiki contents have to be kept synchronized. If a new concept, instance or property appears, or an existing construct is refined, these changes must be reflected in the wiki. During the case study we encountered additional problems that could not be supported by the implemented wiki module. We added for example the entity 'NatureAnalogMethod' as an instance of 'ComputationalMethod'. The system generated a wiki page for this instance that was modified by DEs. After some time ODs decided to remove the instance from the ontology and to add a new concept 'NatureAnalogMethod' which is intended to act as a superclass for all computational methods and algorithms, that are copied from nature. This in turn triggered the system to flag the corresponding wiki page for the instance 'NatureAnalogMethod' as being related to a deleted instance of the ontology and created a new wiki page for the class 'NatureAnalogMethod'. This again led to the undesired effect that the historical relation between the wiki pages for the instance and the class got lost.

Using Tagging of Papers for Ontology Extension. Vast amounts of biomedical and life science journal literature are available in digital archives like PubMed[1]. In order to exploit the information contained in these archives for ontology extension purposes, relevant publications have to be initially retrieved. For our case study we implemented an application module to support users (ODs and DEs likewise) to add project specific publications to a BIO2Me document collection. The users can search for publications in PubMed using search terms and then decide which of them should be added to the collection. For the selected publications the basic bibliographic data is fetched from PubMed including the abstract, the authors names, used citations and existing keywords. In addition to potentially existing PubMed keywords the users are able to annotate the retrieved abstracts with unrestricted self-defined tags.

Within the case study we were able to use these tags primarily for the extension of BIO2Me's class hierarchy and for ontology population. Using the tags added by our domain experts we extended e. g. the sub-classes for the concept 'Program' with class 'StructureAlignmentProgram' which itself was modeled to directly contain corresponding structure alignment programs as subclasses and the program versions of these as instances.

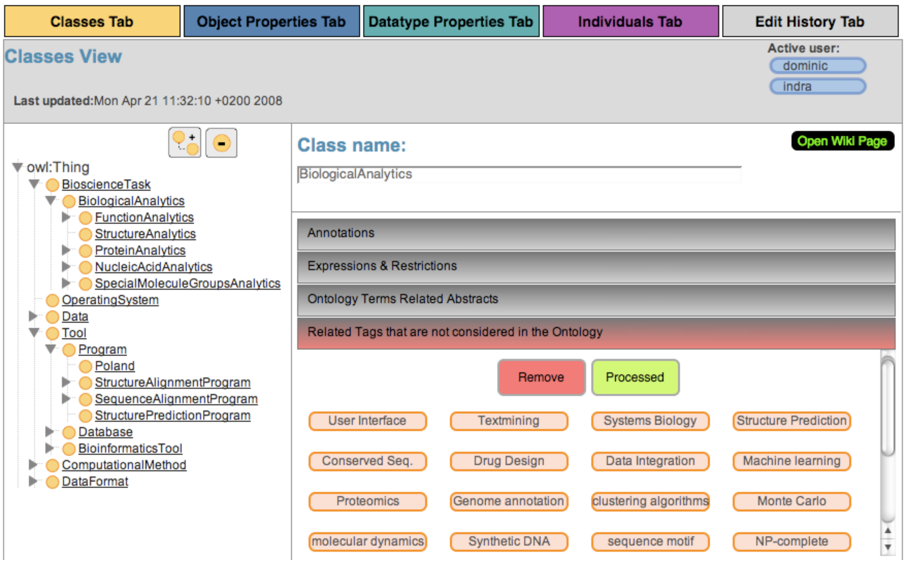
Tags that have been used for ontology modification and were added to a new construct as labels are automatically removed from the tag list (see Fig. 6). In case the tag is not used as a label but has already been incorporated indirectly (e. g. misspellings) an OD can label this tag as processed. In a similar way the OD can remove it from the list. In this case the tag is saved as being not relevant for the extension of the ontology and will not be displayed in the editor again.

Exploitation of Term-related Abstracts for Ontology Sophistication. Finding entities, e. g. by capturing new tags associated to domain relevant publications as described above, alone is not sufficient; most of the important information is contained within the relations between entities [28,29]. To improve the extraction of new entities as well as their attributes and relations we developed a straightforward extension for our ontology development environment. Scientific publications related to the bioinformatics domain get automatically annotated using the RDFS labels of BIO2Me's constructs together with the search library Ferret[2]. Abstracts of these publications that a) belong to the BIO2Me paper collection and b) contain at least one term from the ontology can then be accessed via the ontology editor user interface (see Fig. 7).

In first tests we dicovered that applying this relatively simple approach could support our DEs by identifying new entities and extending given concepts. Instead of manually identifying domain related publications, which is particularly crucial for small DE groups (as in case of BIO2Me), the system offers

[1] PubMed Central (PMC) `http://www.pubmedcentral.nih.gov`

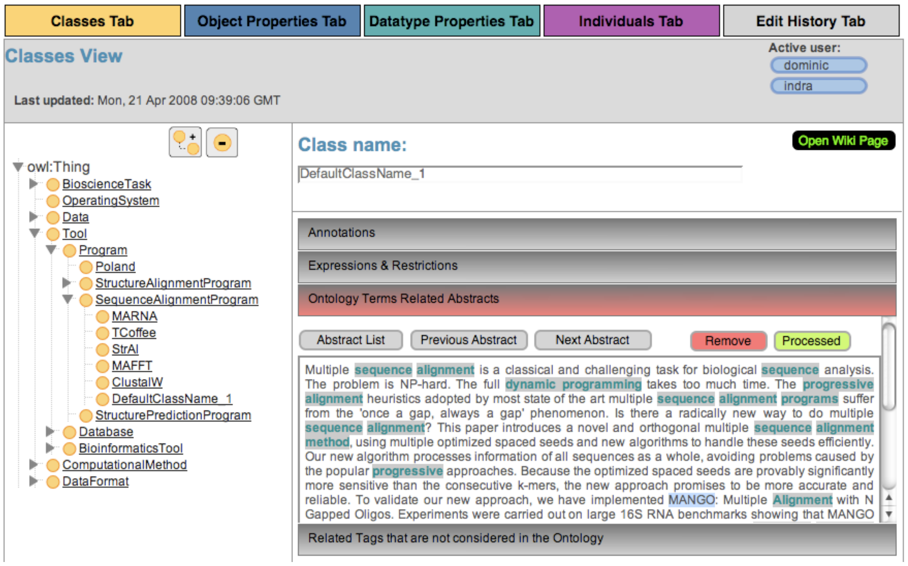
[2] `http://ferret.davebalmain.com`



Last updated: Mon Apr 21 11:32:10 +0200 2008

Class name: BiologicalAnalytics

Fig. 6. Using tagged papers for ontology population and extension



Last updated: Mon, 21 Apr 2008 09:39:06 GMT

Class name: DefaultClassName_1

Multiple sequence alignment is a classical and challenging task for biological sequence analysis. The problem is NP-hard. The full dynamic programming takes too much time. The progressive alignment heuristics adopted by most state of the art multiple sequence alignment programs suffer from the 'once a gap, always a gap' phenomenon. Is there a radically new way to do multiple sequence alignment? This paper introduces a novel and orthogonal multiple sequence alignment method, using multiple optimized spaced seeds and new algorithms to handle these seeds efficiently. Our new algorithm processes information of all sequences as a whole, avoiding problems caused by the popular progressive approaches. Because the optimized spaced seeds are provably significantly more sensitive than the consecutive k-mers, the new approach promises to be more accurate and reliable. To validate our new approach, we have implemented MANGO: Multiple Alignment with N Gapped Oligos. Experiments were carried out on large 16S RNA benchmarks showing that MANGO

Fig. 7. Use of term related abstracts in the ontology editor

pre-processed textual context for ontology sophistication. Within the BIO2Me project e. g. the identification of the biological questions for which a bioinformatics tool can be applied turned out to be difficult for those areas where none of our ODs had expertise. With our new tool however we were able to retrieve those abstracts containing relevant information by looking at abstracts that contained both the RDFS label of a bioinformatics tool and at least one label of a modeled bioscience task (e. g. nucleic acid electrophoresis) from the ontology.

3 Conclusion and Future Works

With the growing importance of bio-ontologies the necessity of tools for sustainable ontology maintenance becomes relevant. We suggest that this can be achieved best, if a community of experts with different abilities jointly develops and maintains an ontology. Some straightforward aspects for a collaborative approach to create, modify and extend ontologies through a deeper integration of DEs into the knowledge capturing process were proposed.

We adapted a wiki engine in consideration of knowledge acquisition aspects together with the exploitation of publication tagging for ontology extension. A first prototype of this adaption was implemented to support a sustainable and knowledge acquisition focused cooperative development of bio-ontologies.

Our positive experiences with these new tools encourages further developments on the way to a common platform for scientific domain driven bio-ontology development. For future extensions communication between DEs and ODs should not only be transformed to technical systems but also further extended by making use of sophisticated tagging mechanisms (like extreme tagging – tagging of tags combined with text mining tools [30], ontology coupled wiki systems, and recommendation systems to extend and populate ontologies. In particular text mining tools could broaden the approach we presented for the exploitation of term related abstracts through supporting iterative improvements to ontologies by determining and highlighting new concepts. These tools may find relations between terms similar as in latent semantic analysis, where relationships between document corpora are analyzed.

Additional possibilities to support the tagging process within publications come from the feature extraction research field. An implementation of this technique could be used to detect recurring word-usage patterns in publication abstracts, which can help to identify tags as candidates for ontology enrichment. However, difficulties in accessing full text articles remain a drawback for indexing and tagging publications as we experienced during this case-study. As a consequence, searching and document processing are often limited to abstracts. While this has started to improve, it still inhibits large-scale activities to extend and populate ontologies.

Acknowledgments

Financial support was provided by the german Bundesministerium für Bildung, Wissenschaft, Forschung und Technologie (BMBF). A. von Haeseler is also supported by the Wiener Wissenschafts-, Forschungs und Technologiefonds (WWTF).

References

1. Gruber, T.R.: A Translation Approach to Portable Ontologies. Knowledge Acquisition 5(2), 199–220 (1993)
2. McGuinness, D., van Harmelen, F.: OWL Web Ontology Language Overview. W3C Recommendation (2004), http://www.w3.org/TR/owl-features/ (retrieved January 10, 2008)
3. Larkin, M.A., Blackshields, G., Brown, N.P., Chenna, R., McGettigan, P.A., McWilliam, H., Valentin, F., Wallace, I.M., Wilm, A., Lopez, R., Thompson, J.D., Gibson, T.J., Higgins, D.G.: ClustalW2 and ClustalX Version 2. Bioinformatics 23(21), 2947–2948 (2007)
4. Goble, C., Corcho, O., Alper, P., De Roure, D.: E-Science and the Semantic Web: A Symbiotic Relationship. In: Todorovski, L., Lavrač, N., Jantke, K.P. (eds.) DS 2006. LNCS (LNAI), vol. 4265, pp. 1–12. Springer, Heidelberg (2006)
5. Hey, T., Trefethen, A.: Cyberinfrastructures for e-Science. Science 308, 817–821 (2005)
6. Berners-Lee, T., Hendler, J., Lassila, O.: The Semantic Web. Scientific American, 28–37 (2001)
7. Corcho, O., Alper, P., Kotsiopoulos, I., Missier, P., Bechhofer, S., Goble, C.: An overview of S-OGSA. A Reference Semantic Grid Architecture. Journal of Web Semantics 4(2), 102–115 (2006)
8. Gruber, T.: Ontology of Folksonomy. A Mash-Up of Apples and Oranges. In: 1st On-Line Conference on Metadata and Semantics Research (MTSR 2005) (2005), http://tomgruber.org/writing/ontology-of-folksonomy.htm (retrieved January 10, 2008)
9. Paulsen, I., Mainz, D., Weller, K., Mainz, I., Kohl, J., von Haeseler, A.: Ontoverse: Collaborative Knowledge Management in the Life Sciences Network. In: Proceedings of the Germany eScience Conference 2007, Max Planck Digital Library (2007) ID 316588.0
10. Baclawski, K., Niu, T.: Ontologies for Bioinformatics. MIT Press, Cambridge (2006)
11. Bodenreider, O., Stevens, R.: Bio-Ontologies: Current Trends and Future Directions. Briefings in Bioinformatics 7(3), 256–274 (2006)
12. Burger, A., Davidson, D., Baldock, R. (eds.): Anatomy Ontologies for Bioinformatics. Principles and Practice. Springer, Goldaming (2008)
13. Ashburner, M., et al.: Gene Ontology: Tool for the Unification of Biology. Nat. Genet. 25, 25–29 (2000)
14. Baker, P.G., Goble, C.A., Bechhofer, S., Paton, N.W., Stevens, R., Brass, A.: An Ontology for Bioinformatics Applications. Bioinformatics 15(6), 510–520 (1999)
15. Chen, R.O., Felciano, R., Altman, R.B.: RiboWeb: Linking Structural Computations to a Knowledge Base of Published Experimental Data. In: Proceedings of the 5th International Conference on Intelligent Systems for Molecular Biology, pp. 84–87. AAAI Press, Menlo Park (1997)
16. Altman, R., Bada, M., Chai, X.J., Whirl Carillo, M., Chen, R.O., Abernethy, N.F.: RiboWeb: An Ontology-Based System for Collaborative Molecular Biology. IEEE Intelligent Systems 14(5), 68–76 (1999)
17. Karp, P., Paley, S.: Integrated Access to Metabolic and Genomic Data. Journal of Computational Biology 3(1), 191–212 (1998)

18. Smith, B., Ashburner, M., Rosse, C., Bard, J., Bug, W., Ceusters, W., Goldberg, L.J., Eilbeck, K., Ireland, A., Mungall, C.J., Leontis, N., Rocca-Serra, P., Ruttenberg, A., Sansone, S.A., Scheuermann, R.H., Shah, N., Whetzel, P.L., Lewis, S.: The OBO Foundry: Coordinated Evolution of Ontologies to Support Biomedical Data Integration. Nature Biotechnology 25, 1251–1255 (2007)
19. Rojas, I., Ratsch, E., Saric, J., Wittig, U.: Notes on the Use of Ontologies in the Biochemical Domain. Silico Biology 4, 9 (2003)
20. Stevens, R., Goble, C.A., Bechhofer, S.: Ontology-based Knowledge Representation for Bioinformatics. Briefings in Bioinformatics 1(4), 398–416 (2000)
21. Godfray, H.C.J.: Challenges for Taxonomy. Nature 417, 17–19 (2002)
22. Wilm, A., Mainz, I., Steger, G.: An Enhanced RNA Alignment Benchmark for Sequence Alignment Programs. Algorithms for Molecular Biology 1(19) (2006)
23. Dalli, D., Wilm, A., Mainz, I., Steger, G.: StrAl: Progressive Alignment of Noncoding RNA Using Base Pairing Probability Vectors in Quadratic Time. Bioinformatics 22(13), 1593–1599 (2006)
24. Bao, J., Honavar, V.: Collaborative Ontology Building with Wiki@nt. A Multiagent Based Ontology Building Environment. In: Proceedings of the 3rd International Workshop on Evaluation of Ontology-based Tools (EON), Hiroshima, pp. 1–10 (2004)
25. Fensel, D.: Ontologies: Silver Bullet for Knowledge Management and Electronic Commerce. Springer, Heidelberg (2004)
26. Peters, I., Stock, W.G.: Folksonomy and Information Retrieval. In: Proceedings of the 70th Annual Meeting of the American Society for Information Science and Technology, vol. 45, pp. 1510–1542 (2007) (CD-ROM)
27. Sure, Y., Staab, S., Studer, R.: Methodology for Development and Employment of Ontology Based Knowledge Management Applications. ACM SIGMOD Record, Special Section on Semantic Web and Data Management 4(31), 18–23 (2002)
28. Buitelaar, P., Cimiano, P., Magnini, B. (eds.): Ontology Learning from Text: Methods, Evaluation and Applications. Frontiers in Artificial Intelligence and Applications, vol. 123. IOS Press, Amsterdam (2005)
29. Staab, S., Studer, R.: Handbook on Ontologies. Springer, Heidelberg (2004)
30. Cohen, K.B., Hunter, L.: Natural Language Processing and Systems Biology. In: Dubitzky, W., Pereira, F. (eds.) Artificial intelligence and systems biology, Springer, Berlin (2004)

Nested q-Partial Graphs for Genetic Network Inference from "Small n, Large p" Microarray Data

Kevin Kontos and Gianluca Bontempi

ULB Machine Learning Group
Computer Science Department
Université Libre de Bruxelles
Boulevard du Triomphe CP 212, 1050 Brussels, Belgium
{kkontos,gbonte}@ulb.ac.be
http://www.ulb.ac.be/di/mlg/

Abstract. Gaussian graphical models are widely used to tackle the important and challenging problem of inferring genetic regulatory networks from expression data. These models have gained much attention as they encode full conditional relationships between variables, i.e. genes. Unfortunately, microarray data are characterized by a low number of samples compared to the number of genes. Hence, classical approaches to estimate the full joint distribution cannot be applied. Recently, limited-order partial correlation approaches have been proposed to circumvent this problem. It has been shown both theoretically and experimentally that such graphs provide accurate approximations of the full conditional independence structure between the variables thanks to the sparsity of genetic networks. Alas, computing limited-order partial correlation coefficients for large networks, even for small order values, is computationally expensive, and often even intractable. Moreover, problems deriving from multiple statistical testing arise, and one should expect that most of the edges are removed. We propose a procedure to tackle both problems by reducing the dimensionality of the inference tasks. By adopting a screening procedure, we iteratively build nested graphs by discarding the less relevant edges. Moreover, by conditioning only on relevant variables, we diminish the problems related to multiple testing. This procedure allows us to faster infer limited-order partial correlation graphs and to consider higher order values, increasing the accuracy of the inferred graph. The effectiveness of the proposed procedure is demonstrated on simulated data.

1 Introduction

Reverse engineering of genetic regulatory networks (GRNs) from expression data is an essential step toward the modeling of genetic networks. The inference of these networks from expression data alone is far from trivial because of the combinatorial nature of the problem and the poor information content of the data [1].

Graphical models [2,3,4] have been widely used to address this important and challenging problem. Among these, Gaussian graphical models have become increasingly popular. These models encode full conditional relationships between genes. Hence, they enable to distinguish direct from indirect interactions. Standard multivariate methods for structure learning of Gaussian graphical models require the estimation of the full joint probability distribution. Unfortunately, typical microarray data sets describe a large number of variables (on the order of hundreds or thousands) but only contain comparatively few samples (on the order of tens or hundreds), which renders this estimation an ill-posed problem. Hence, standard multivariate methods cannot be applied directly.

Recently, limited-order partial correlation graphs have been proposed as an alternative to Gaussian graphical models [5,6,7,8,9]. Because genetic networks are sparse, the former provide accurate approximations of the latter. However, for these approximations to be as accurate as possible, the order values should not be taken too small. Unfortunately, for large networks, inferring limited-order partial correlation graphs even for small order values is computationally expensive, and often even intractable, unless very small order values are considered. Moreover, problems deriving from multiple statistical testing arise, and one should expect that most of the edges are removed [5].

We propose a procedure to tackle both problems by reducing the dimensionality of the inference tasks. First, we adopt a screening procedure, recently introduced in the context of feature selection [10]. Under the assumption of faithfulness, we exploit an inclusion relation of high-order partial correlation graphs in lower order ones: we iteratively build nested graphs by discarding the less relevant edges. Moreover, by conditioning only on relevant variables, we diminish the problems related to multiple testing.

The advantage of this procedure is that it considerably speeds up the inference of limited-order partial correlation graphs. As a consequence, this approach enables us to consider higher order values, increasing the accuracy of the inferred graph. We demonstrate the effectiveness of the proposed procedure on simulated data.

The paper is cast in the framework of the recently proposed q-partial graph theory [5]. q-Partial graphs are a generalization of limited-order and full-order partial correlation graphs. They clarify the connection between the sparseness of the concentration graph and the usefulness of limited-order partial correlation graphs.

The outline of the paper is as follows. Section 2 reviews the state-of-the-art graphical models used to infer regulatory networks. The theory of q-partial graphs is given in Sect. 3. The q-nested procedure for inferring q-partial graphs is presented in Sect. 4. Instances of its application to simulated data are given in Sect. 5. Finally, Sect. 6 concludes the paper.

2 Graphical Models for Genetic Regulatory Network Inference

This section reviews the most important graphical models used to infer genetic regulatory networks (GRNs) from high-dimensional microarray data.

2.1 Independence Graphs

The first and simplest model to infer GRNs from "small n, large p" microarray data is that of independence graph, also known as covariance graph and gene relevance network. It was proposed by [11] and is built as follows. First, the correlation matrix is inferred from data. Next, pairs of genes are connected if their respective correlation exceeds a given threshold. Independence graphs therefore represent the marginal independence structure of the genes. Despite their relative ease of construction, these networks suffer a major drawback: marginal independence is a strong indicator for independence, but a weak criterion for measuring dependence, since more or less all genes will be marginally (i.e., directly or indirectly) correlated [12] (see Fig. 1 for an example).

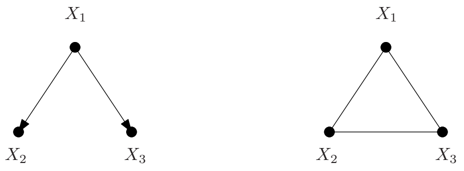


Fig. 1. A simple network consisting of 3 genes (left). Genes X_2 and X_3 are highly correlated with each other because both are regulated by gene X_1. The spurious relation between genes X_2 and X_3 will therefore most probably be inferred in the independence graph (right). Note that the directions of the identified connections are not inferred in the independence graph.

2.2 Concentration Graphs: Full-Order Partial Correlation Graphs

In order to overcome this shortcoming, concentration graphs, also known as Gaussian graphical models[1] (GGMs), have become popular to infer GRNs [13,14]. In these models, missing edges denote zero full-order partial correlations, and therefore, correspond to conditional independence relationships.

Full-order partial correlation measures the association between two genes while taking into account all the remaining observed genes. Therefore, GGMs have an important advantage compared to independence graphs: they enable to distinguish direct from indirect interactions between genes due to intermediate genes (sequential pathways) or directly due to other genes (common causes).

However, computing full-order partial correlations requires the full joint distribution of genes. This is problematic in the "small n, large p" data setting: the maximum likelihood estimate of the population concentration matrix needed to infer GGMs requires that the sample covariance matrix has full rank and this holds, with probability one, if and only if $n > p$ [15].

To circumvent this problem, the first approach proposed in the literature restricts the analysis to very small numbers of genes or gene clusters (as to satisfy $n > p$) [16,17,18,19,20], which is unsuited for inferring GRNs. Two

[1] These two names are hereafter used interchangeably.

alternative approaches have thus been introduced: one uses regularization techniques [21,13,14,22,23,24], while the other uses limited-order partial correlations [5,6,7,8,9]. This latter approach is discussed in this paper.

3 q-Partial Graphs: From Independence Graphs to Gaussian Graphical Models

A q-partial graph is inferred from q-order partial correlation coefficients which measure the correlation between pair of variables when conditioning on q other variables, with $q \in \{0, \ldots, p-2\}$. The recently introduced theory of q-partial graphs "clarifies the connection between the sparseness of the concentration graph and the usefulness of marginal distributions in structure learning, under the assumption of faithfulness" [5]. Hence, q-partial graphs provide a common framework for graphs inferred from limited-order partial correlations and GGMs.

Let $X_\mathcal{V}$ be a random vector indexed by $\mathcal{V} = \{1, \ldots, p\}$ with probability distribution $P_\mathcal{V}$ and let $G = (\mathcal{V}, \mathcal{E})$ be an undirected graph. For a subset $\mathcal{U} \subseteq \mathcal{V}$, we denote by $X_\mathcal{U}$ the subvector of X indexed by $\mathcal{U}$, and by $P_\mathcal{U}$ the associated marginal distribution.

We now give the definition of q-partial graphs [5].

Definition 1 (q-partial graph). *For a random vector $X_\mathcal{V}$ and an integer $0 \leq q \leq (p-2)$, the q-partial graph of $X_\mathcal{V}$, denoted by $G_{(q)} = \left(\mathcal{V}, \mathcal{E}_{(q)}\right)$, is the undirected graph where the edge $(i,j) \notin \mathcal{E}_{(q)}$ if and only if there exists a set $\mathcal{U} \subset \mathcal{V}$ with $|\mathcal{U}| \leq q$ and $i, j \notin \mathcal{U}$ such that $X_i \perp\!\!\!\perp X_j \mid X_\mathcal{U}$ holds in $P_\mathcal{V}$.*

In this definition, $\perp\!\!\!\perp$ is the usual notation for independence [25]. Note that we consider undirected graphs and do not distinguish between (i,j) and (j,i).

The random vector $X_\mathcal{V}$ is assumed to have a multivariate normal distribution with mean vector μ and positive definite covariance matrix Σ. Furthermore, we assume that $P_\mathcal{V}$ is both Markov and faithful with respect to $G = \{\mathcal{V}, \mathcal{E}\}$. We say that $P_\mathcal{V}$ is (undirected) Markov with respect to G if it holds that $X_\mathcal{I} \perp\!\!\!\perp X_\mathcal{J} \mid X_\mathcal{U}$ whenever $\mathcal{U}$ separates $\mathcal{I}$ and $\mathcal{J}$ in G [26]. We say that $P_\mathcal{V}$ is faithful to G if all the conditional independence relationships in $P_\mathcal{V}$ can be read off the graph G through the Markov property [26].

Thus, for a subset $\mathcal{U} \subset \mathcal{V}$ with $i, j \notin \mathcal{U}$ it holds that $X_i \perp\!\!\!\perp X_j \mid X_\mathcal{U}$ if and only if the partial correlation coefficient

$$\rho_{(i,j|\mathcal{U})} = \frac{-k_{ij}^{(\mathcal{W})}}{\sqrt{k_{ii}^{(\mathcal{W})} k_{jj}^{(\mathcal{W})}}} , \tag{1}$$

is equal to zero, where $\mathcal{W} = \mathcal{U} \cup \{i,j\}$, $K^{(\mathcal{W})} = \left\{k_{ij}^{(\mathcal{W})}\right\} = \left(\Sigma_{\mathcal{W}\mathcal{W}}\right)^{-1}$ is the concentration matrix of $X_\mathcal{W}$ (provided that $\Sigma_{\mathcal{W}\mathcal{W}}$ is invertible) [2].

Note that the cases $q = 0$ (conditioning on the empty set) and $q = p-2$ (conditioning on all remaining variables) correspond to the independence graph (see Sect. 2.1) and to the GGM (see Sect. 2.2), respectively.

Before giving Prop. 1, we need the following three definitions. We start with the definition of connectivity [27].

Definition 2 (connectivity). *Let $i \neq j$ be a pair of vertices of an undirected graph $G = (\mathcal{V}, \mathcal{E})$. The connectivity of i and j is defined as the cardinality of the smallest (possibly non unique) set $\mathcal{S}_{(i,j \mid G)}$ of nodes to be removed to make i and j disconnected.*

Next, we give a slightly different definition of connectivity of two vertices [5].

Definition 3 (outer connectivity). *Let $i \neq j$ be a pair of vertices of an undirected graph $G = (\mathcal{V}, \mathcal{E})$. The outer connectivity of i and j is defined as*

$$d\,(i, j \mid G) = \min_{S \in \mathcal{S}_{(i,j \mid G_{ij})}} |S|$$

where $G_{ij} = (\mathcal{V}, \mathcal{E} \setminus (i,j))$ and $|S|$ is the cardinality of S.

Hence, $d\,(i, j \mid G)$ is the connectivity of i and j in G_{ij} (that is the graph obtained by removing edge (i, j), if present, from graph G).

Finally, we give the definition of the outer connectivity of the set $\overline{\mathcal{E}}$ of missing edges of $G = (\mathcal{V}, \mathcal{E})$ [5], that is, for a pair $(i, j) \in \mathcal{V}$, $(i, j) \in \overline{\mathcal{E}}$ if and only if $i \neq j$ and $(i, j) \notin \mathcal{E}$.

Definition 4 (outer connectivity of the missing edges). *The outer connectivity of the missing edges of $G = (\mathcal{V}, \mathcal{E})$ is defined as*

$$d\left(\overline{\mathcal{E}} \mid G\right) = \max_{(i,j) \in \overline{\mathcal{E}}} d\,(i, j \mid G) \ .$$

The following proposition (the proof of which can be found in [5]) describes when the full-order partial correlation graph can be uncovered by computing only limited-order partial correlation coefficients.

Proposition 1. *Let $G = (\mathcal{V}, \mathcal{E})$ and $G_{(q)} = \left(\mathcal{V}, \mathcal{E}_{(q)}\right)$ be the concentration and the q-partial graph of $X_{\mathcal{V}}$, respectively. Then $G = G_{(q)}$ if and only if*

$$d\left(\overline{\mathcal{E}} \mid G\right) \leq q \ .$$

Note that $d\left(\overline{\mathcal{E}} \mid G\right) \leq q$ if and only if there exists a marginal distribution of $X_{\mathcal{V}}$ of dimension $q + 2$ in which the corresponding variables are conditionally independent. Under the assumption of faithfulness $G \subseteq G_{(q)}$ [5]. Therefore, Proposition 1 states that a missing edge in G is missing also in $G_{(q)}$ if and only if the outer connectivity of the corresponding vertices is smaller or equal to q. Hence, the q-partial graph and the concentration graph are identical if this relation is satisfied for all missing edges in G [5] (see Fig. 2 for an illustration).

Note that sparseness is useful as long as it implies small separators for non-adjacent vertices but there is no direct connection between the degree of sparseness of G and the missing edges' outer degree [5].

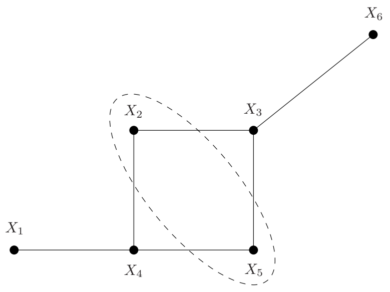


Fig. 2. Outer connectivity illustrated. A simple network G consisting of 6 genes is depicted. All pairs of non-connected genes can be separated by at least one subset of at most 2 genes. For example, X_3 and X_4 can be separated by the set $\{X_2, X_5\}$ (dotted ellipse). Therefore we have $G_{(2)} = G_{(3)} = G_{(4)} = G$.

Intuitively, larger values of q should be preferred. This is confirmed by the following inclusion relation which holds under the assumption of faithfulness [5]:

$$G_{(q)} \subseteq G_{(r)} \qquad \text{if} \quad r \leq q \; . \tag{2}$$

Hence, the higher the value of q, the closer the approximation to the true (concentration) graph. However, as Prop. 1 makes clear, it is possible to obtain a good approximation of the concentration graph, and sometimes even the exact concentration graph, without conditioning on all remaining genes (i.e., $q = p-2$, where p is the number of genes).

In principle, q-order partial correlations can be computed for any $q \in \{0, \ldots, p-2\}$. However, in practice, four problems arise.

Problem 1. Computing a partial correlation coefficient of order q requires the inversion of a $n \times (q+2)$ submatrix of the concentration matrix (see (1)), where n is the number of samples, which requires that this submatrix has full rank and this holds [15], with probability one, if and only if

$$q < n - 2 \; . \tag{3}$$

Problem 2. The computation of $\binom{p-2}{q}$ q-partial correlation coefficients for each of the $p\,(p-1)\,/2$ possible pairs of genes can be computationally intensive, and often even intractable for large networks, except if q takes on (very) small or (very) large values. Therefore, unless full-order partial correlations are considered, limited-order partial correlations are restricted to $q \leq 3$ in the literature [6,8,7,9], except for the approach proposed in [5] where higher values of q are considered because only a (random) subset of the $\binom{p-2}{q}$ q-partial correlation coefficients are computed.

Problem 3. An edge is added to the q-partial graph if all of $\binom{p-2}{q}$ null hypotheses are rejected. But if the value of $\binom{p-2}{q}$ is large then most, or even all, of the edges are removed because the probability that at least one hypothesis of zero

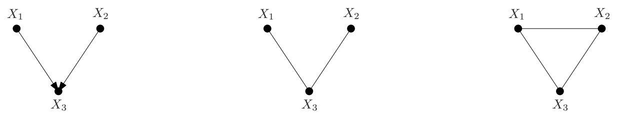


Fig. 3. A simple network consisting of 3 genes (left). Gene X_3 is regulated by both genes X_1 and X_2, which are not correlated. Genes X_1 and X_2 are therefore not connected in the independence graph (center), i.e. the 0-partial graph. In case of lack of faithfulness, a spurious connection between X_1 and X_2 may be inferred when conditioning on gene X_3 in the 1-partial graph (right).

q-order partial correlation is wrongly non-rejected increases with the number of performed tests [5].

Problem 4. The assumption of faithfulness is sometimes violated. This implies that a missing edge in $G_{(q)}$ may be present in $G_{(r)}$, with $r > q$ (as illustrated in Fig. 3). Note that this undesirable effect is well-known in the literature on causal inference [26,28] where it is referred to as the "explaining away effect".

The first problem is not very stringent as microarray data typically consist of several tens or hundreds of samples. The second one, however, drastically limits the order q of the partial graph. The third issue is equally serious, since it hinders the applicability of high-order q-partial graphs. Finally, the last problem has typically a weak impact on the estimates of partial correlation [5].

Hence, the second and third problem need to be addressed for limited-order partial correlation graphs to be applicable in practice. It is the aim of our procedure, which we present in the next section, to tackle both problems.

4 The q-Nested Procedure

We now present an original approach to infer q-partial graphs. This method takes advantage of the inclusion relation (2) to reduce the number of pairs of genes for which to compute the q-partial correlation coefficients. It is inspired by a simplified version of the PC-algorithm [28] recently proposed for variable selection in high-dimensional linear models [10]. Furthermore, we show that for each pair of genes only a small number of q-partial correlation coefficients (out of $\binom{p-2}{q}$ possible coefficients) have to be computed. This further reduces the dimensionality of the inference problem.

Let us first assume that partial correlations are known.

4.1 Population Version

Recall from (2) that, under the assumption of faithfulness, $G_{(q)} \subseteq G_{(r)}$ if $r \leq q$. This inclusion relation implies that every missing edge in $G_{(r)}$ is also missing in $G_{(q)}$. Hence, if we have inferred $G_{(r)}$, we only need to compute q-partial correlations between genes connected in $G_{(r)}$ to infer $G_{(q)}$, and not for all $p\,(p-1)\,/2$

possible edges. Assuming the concentration graph is sparse, this screening may substantially reduce the dimensionality of the problem.

More specifically, let $\mathcal{E}^{(-1)} = \mathcal{V} \times \mathcal{V} = \{(i,j) \mid i,j \in \mathcal{V}, i \neq j\}$ be the set of all possible edges (i.e., of all unordered pairs of genes). We can do screening according to marginal correlations by building the set:

$$\mathcal{E}^{(0)} = \left\{(i,j) \in \mathcal{E}^{(-1)} \mid \rho_{(i,j)} = \rho_{(i,j|\emptyset)} \neq 0\right\} .$$

We then continue screening using higher-order partial correlations and building the sets

$$\mathcal{E}^{(q)} = \left\{(i,j) \in \mathcal{E}^{(q-1)} \mid \rho_{(i,j|S)} \neq 0, \text{ for all } S \subseteq \mathcal{V} \setminus \{i,j\} \text{ with } |S| = q\right\} ,$$

for $q \in \{1, \ldots, p-2\}$, ending up with a nested sequence of sets:

$$\mathcal{E}^{(0)} \supseteq \mathcal{E}^{(1)} \supseteq \cdots \supseteq \mathcal{E}^{(k)} \supseteq \cdots \supseteq \mathcal{E}^{(p-2)} .$$

The dimensionality of the problem can be further reduced as the following corollary shows. Note that a similar approach is used for the inference of directed acyclic graphs [28]. First, we need the following proposition. The boundary of vertex i in G, i.e., the set of vertices adjacent to i, is denoted by $\mathrm{bd}_G(i)$.

Proposition 2. *Let $(i,j) \in \mathcal{E}^{(q)}$ for a fixed $q \in \{0, \ldots, p-3\}$. If there exists a set S, with $|S| = q+1$, such that $\rho_{(i,j|S)} = 0$, then there exists two sets $S_i \subseteq bd_{G_{(q)}}(i)$ and $S_j \subseteq bd_{G_{(q)}}(j)$ such that $\rho_{(i,j|S_i)} = 0$ and $\rho_{(i,j|S_j)} = 0$.*

Proof. We have $X_i \perp\!\!\!\perp X_j \mid X_S$ and thus $X_i \perp\!\!\!\perp X_j \mid X_{\mathcal{V}\setminus\{i,j\}}$. Using the global Markov property [2], we have that $X_i \perp\!\!\!\perp X_j \mid X_{\mathrm{bd}_{G_{(p-2)}}(i)}$ and $X_i \perp\!\!\!\perp X_j \mid X_{\mathrm{bd}_{G_{(p-2)}}(j)}$. Since $\mathrm{bd}_{G_{(p-2)}}(i) \subseteq \mathrm{bd}_{G_{(q)}}(i)$ and $\mathrm{bd}_{G_{(p-2)}}(i) \subseteq \mathrm{bd}_{G_{(q)}}(i)$ (recall (2)), the proof is complete. □

An immediate consequence of Prop. 2 is the following corollary.

Corollary 1. *The screening procedure described above returns the same approximation of the concentration graph whether testing $\rho_{(i,j|S)} \neq 0$ for all $S \subseteq \mathcal{V} \setminus \{i,j\}$, for all $S \subseteq bd_{G_{(q-1)}}(i)$ or for all $S \subseteq bd_{G_{(q-1)}}(j)$ with $|S| = q$.*

The corollary states that we do not need to consider all the $\binom{p-2}{q}$ possible conditioning sets of size q when testing for the presence of an edge (i,j). It suffices to restrict the conditioning sets to sets (of size q) of genes that are adjacent to either node i or node j. In practice, the smallest boundary is chosen.

For all $i,j \in \mathcal{V}$ and $q \in \{0, \ldots, p-2\}$, let

$$\mathrm{bd}_{G_{(q)}}(i,j) = \begin{cases} \mathrm{bd}_{G_{(q)}}(i) & \text{if } |\mathrm{bd}_{G_{(q)}}(i)| \leq |\mathrm{bd}_{G_{(q)}}(j)| \ ; \\ \mathrm{bd}_{G_{(q)}}(j) & \text{otherwise } . \end{cases}$$

The detailed description of our algorithm is given below.

Algorithm 1. q-nested procedure

```
1 $\mathcal{E}^{(0)} \leftarrow \left\{(i,j) \in \mathcal{E}^{(-1)} \mid \rho_{(i,j)} = \rho_{(i,j|\emptyset)} \neq 0\right\}$
2 $q \leftarrow 0$
3 while $\mathcal{E}^{(q)} \neq \mathcal{E}^{(q-1)}$ and $q < p-2$ do
4     $q \leftarrow q+1$
5     $\mathcal{E}^{(q)} \leftarrow \Big\{(i,j) \in \mathcal{E}^{(q-1)} \mid \rho_{(i,j|S)} \neq 0,$
                for all $S \subseteq \mathrm{bd}_{G_{(q-1)}}(i,j)$ with $|S| = q\Big\}$
  end
```

4.2 Sample Version

For finite samples, partial correlation coefficients are computed by replacing the concentration matrix K in (1) by its sample counterpart $\hat{K}$.

In a frequentist approach to inference, we require the distribution function of the sample partial correlation coefficient $\hat{\rho}_{(i,j|S)}$ under the null hypothesis $\rho_{(i,j|S)} = 0$ for all $S \subseteq \mathcal{V} \setminus \{i,j\}$ to address the statistical testing problem of non-zero partial correlation

$$H_0 : \rho_{(i,j|S)} = 0 \qquad \text{versus} \qquad H_1 : \rho_{(i,j|S)} \neq 0 \ .$$

Here we apply Fisher's Z-transform

$$Z_{(i,j|\,S)} = \tanh^{-1} \hat{\rho}_{(i,j|S)} = \frac{1}{2} \log \left(\frac{1 + \hat{\rho}_{(i,j|S)}}{1 - \hat{\rho}_{(i,j|S)}} \right) \ .$$

Using a significance level α, we reject the null-hypothesis $H_0 : \rho_{(i,j|S)} = 0$ against the two-sided alternative $H_1 : \rho_{(i,j|S)} \neq 0$ if

$$\sqrt{n-q-3}\, Z_{(i,j|\,S)} > \Phi^{-1}(1 - \alpha/2) \ , \tag{4}$$

where $q = |S|$ and $\Phi(\cdot)$ denotes the cumulative distribution function of the standard normal distribution $\mathcal{N}(0,1)$. Note that (4) implies

$$q < n - 2 \ . \tag{5}$$

The sample version of the q-nested algorithm is obtained by modifying Algorithm 1 as follows:

- Replace in Step 3 the condition $q < p - 2$ by $q < n - 2$ because of (3). Note that inequality (5) is fulfilled.
- Replace in Steps 1 and 5 the statements about, respectively, $\rho_{(i,j)} = \rho_{(i,j|\emptyset)} \neq 0$ and $\rho_{(i,j|S)} \neq 0$ by the statistical hypothesis test described above. Note that we correct the P-values over the multiple tests for all edges using the Benjamini-Hochberg correction [29] for controlling the false discovery rate.

5 Experiments

5.1 Datasets

We used the package GeneNet [30] for the statistical software R [31] to generate 10 networks with $p = 50$ genes, 123 edges (that is $\sim 10\%$ of all possible edges), and $n = 30$ samples.

More specifically, 10 random "true" full-order partial correlation 50×50 matrices are generated by an algorithm which guarantees that the resulting matrices are always positive definite [13]. The non-zero entries of these matrices correspond to the edges of the "true" networks. Next, for each network, simulated data of the desired sample size $n = 30$ are drawn from the multivariate normal distribution with mean zero and the "true" correlation structure.

Although the model used for data generation is a simplification of real molecular processes, it is important to faithfully evaluate the prediction results. This is possible only if the true structure of the regulatory network is known.

5.2 Methods

We compare our q-nested procedure to the independence graph (see Sect. 2.1) and the 1-partial graph (see Sect. 2.2). Higher-order partial graphs were not considered because they are computationally expensive to infer.

5.3 Validation

A network inference problem can be seen as a binary decision problem where the inference algorithm plays the role of a classifier: for each pair of genes, the algorithm either adds an edge or not. Each pair of genes is thus assigned a positive label (an edge) or a negative label (no edge).

A positive label (an edge) predicted by the algorithm is considered as a true positive (TP) or as a false positive (FP) depending on the presence or not of the corresponding edge in the underlying true network, respectively. The true and false negatives (TN and FN, respectively) are defined analogously.

The use of receiver operator characteristic (ROC) curves is recommended when evaluating binary decision problems in order to avoid effects related to the chosen threshold [32]. However, if there is a large skew in the class distribution, as is typically the case when inferring genetic networks because of their sparsity, precision-recall (PR) curves give a more accurate picture of an algorithm's performance [33].

Let the precision quantity

$$prec = \begin{cases} \frac{TP}{TP+FP} & \text{if } TP + FP > 0 \ ; \\ 0 & \text{otherwise} \ ; \end{cases} \tag{6}$$

measure the fraction of true positives, i.e. true edges, among those inferred as positive, and the recall quantity

$$rec = \begin{cases} \frac{TP}{TP+FN} & \text{if } TP + FN > 0 \ ; \\ 0 & \text{otherwise} \ ; \end{cases} \tag{7}$$

measure the fraction of true edges inferred among all true edges. These quantities depend on the threshold chosen to return a binary decision. The PR curve is a diagram which plots the precision versus recall for different values of the threshold on a two-dimensional coordinate system [34]. The quality of an algorithm is measured by the area under the curve (AUC) of the PR curve [33].

For the independence graph and the 1-partial graph, the quantities considered are the correlation coefficients and the 1-order partial correlation coefficients, respectively. In the case of the q-nested partial graph, if an edge has been "screened out", it is the value of the last partial correlation coefficient (before the edge is discarded) that is taken into consideration. Next, we transform the (partial) correlation coefficients into binary values (i.e., edge or no edge) by means of the statistical procedure described in Sect. 4.2 with $\alpha = 0.2$ and compute the F-measure [34]. This measure is defined as the harmonic mean of the precision and recall quantities:

$$F(prec, rec) = \begin{cases} \frac{2 \cdot prec \cdot rec}{prec + rec} & \text{if } prec + rec > 0 \ ; \\ 0 & \text{otherwise} \ . \end{cases} \tag{8}$$

5.4 Results

Figures 4 and 5 show the boxplots of the AUC values and of the F-measures, respectively. q-Nested graphs seem to be perform slightly better than independence graphs and 1-partial graphs.

More importantly, Table 1 shows the mean (and standard deviation) of the percentage of edges (relative to the $50 \times 49/2 = 1225$ possible edges) remaining in the q-partial graphs after each screening step for values of $q \in \{1, \ldots, 7\}$. This illustrates the important reduction in dimensionality achieved through the screening procedure. Of course, the result from Prop. 2 also drastically reduces the dimensionality of the problem. Moreover, it downsizes the undesirable effects of multiple testing (data not shown).

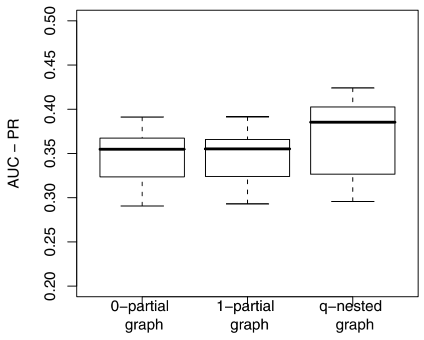


Fig. 4. Boxplots of the AUC values for 10 networks

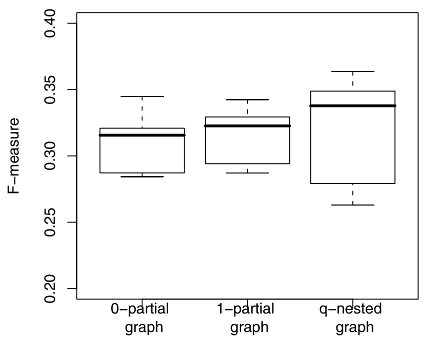


Fig. 5. Boxplots of the F-measures for 10 networks

Table 1. Mean (and standard deviation) of the percentage of edges remaining after each screening step

q	% of rem. edges
1	25.4 (2.92)
2	24.3 (2.55)
3	20.1 (1.53)
4	17.4 (1.16)
5	15.3 (0.94)
6	13.2 (0.68)
7	11.1 (0.67)

6 Conclusion and Future Work

Limited-order partial correlation graphs have become increasingly popular to infer large-scale genetic regulatory networks because of the "small n, large p" problem that arises with microarray data. Their effectiveness to approximate full-order partial correlation graphs has been shown experimentally and theoretically with the recently introduced q-partial graph theory that clarifies the connection between the sparseness of the concentration graph and the usefulness of limited-order partial correlation graphs.

The usefulness of q-partial graphs increases with q, so that a procedure that can be applied for larger values of q is called for. Unfortunately, computing limited-order partial correlation coefficients for large networks, even for small order values, is computationally expensive, and often even intractable. Moreover, problems deriving from multiple statistical testing arise, and one should expect that most of the edges are removed.

We have proposed a procedure to tackle both problems by reducing the dimensionality of the inference task. By adopting a screening procedure recently introduced in the feature selection literature, we iteratively build nested graphs by discarding edges for which lower-order partial correlations are not significantly different from zero. Moreover, we showed that it is not necessary to condition on all possible subsets of a given size, but only on subsets containing genes adjacent to the pair being tested for partial correlation. This result enables to further reduce the dimensionality of the problem and also diminishes the issues related to multiple testing. As we have shown experimentally, our q-nested procedure allows us to faster infer limited-order partial correlation graphs and to consider higher order values, increasing the accuracy of the inferred graphs.

Apart from faithfulness, the q-nested procedure does not require any additional assumptions. The sparseness of the concentration graph is not assumed but exploited when present.

Future work will assess the q-nested approach on a larger number of datasets with different parameter settings. More specifically, the effect of the sample size and of the number of edges in the true underlying network on the proposed procedure will be studied. Comparisons to the procedure presented in [5] as well as to the full-order partial correlation technique of [14] that relies on a shrinkage estimator of the covariance matrix will be undertaken. Finally, our method will be applied to a real gene expression dataset.

Acknowledgments

This work was supported by the Communauté Française de Belgique under ARC grant no. 04/09-307.

References

1. van Someren, E.P., Wessels, L.F.A., Backer, E., Reinders, M.J.T.: Genetic network modeling. Pharmacogenomics 3, 507–525 (2002)
2. Lauritzen, S.L.: Graphical models. Oxford Statistical Science Series. Clarendon Press, Oxford (1996)
3. Whittaker, J.: Graphical Models in Applied Multivariate Statistics. Wiley, Chichester (1990)
4. Edwards, D.: Introduction to Graphical Modelling, 2nd edn. Springer Texts in Statistics. Springer, Heidelberg (2000)
5. Castelo, R., Roverato, A.: A robust procedure for Gaussian graphical model search from microarray data with p larger than n. Journal of Machine Learning Research 7, 2621–2650 (2006)
6. de la Fuente, A., Bing, N., Hoeschele, I., Mendes, P.: Discovery of meaningful associations in genomic data using partial correlation coefficients. Bioinformatics 20, 3565–3574 (2004)
7. Magwene, P., Kim, J.: Estimating genomic coexpression networks using first-order conditional independence. Genome Biology 5, R100 (2004)
8. Wille, A., Zimmermann, P., Vranová, E., Fürholz, A., Laule, O., Bleuler, S., Hennig, L., Preli, A., von Rohr, P., Thiele, L., Zitzler, E., Gruissem, W., Bühlmann, P.: Sparse graphical Gaussian modeling of the isoprenoid gene network in Arabidopsis thaliana. Genome Biology 5, R92 (2004)

9. Wille, A., Bühlmann, P.: Low-order conditional independence graphs for inferring genetic networks. Statistical Applications in Genetics and Molecular Biology 5 (2006) (Article 1)
10. Bühlmann, P., Kalisch, M.: Variable selection for high-dimensional models: partial faithful distributions, strong associations and the PC-algorithm (to appear, 2008), ftp://ftp.stat.math.ethz.ch/Research-Reports/Other-Manuscripts/buhlmann/faithful-varsel-2008.pdf
11. Butte, A., Tamayo, P., Slonim, D., Golub, T., Kohane, I.: Discovering functional relationships between RNA expression and chemotherapeutic susceptibility using relevance networks. Proceedings of the National Academy of Sciences 97, 12182–12186 (2000)
12. Schäfer, J., Strimmer, K.: Learning large-scale graphical gaussian models from genomic data. In: AIP Conference Proceedings, vol. 776, pp. 263–276 (2005)
13. Schäfer, J., Strimmer, K.: An empirical Bayes approach to inferring large-scale gene association networks. Bioinformatics 21, 754–764 (2005)
14. Schäfer, J., Strimmer, K.: A shrinkage approach to large-scale covariance matrix estimation and implications for functional genomics. Statistical Applications in Genetics and Molecular Biology 4, 32 (2005)
15. Dykstra, R.: Establishing the positive definiteness of the sample covariance matrix. The Annals of Mathematical Statistics 41, 2153–2154 (1970)
16. Kishino, H., Waddell, P.: Correspondence analysis of genes and tissue types and finding genetic links from microarray data. Genome Informatics 11, 83–95 (2000)
17. Waddell, P., Kishino, H.: Cluster inference methods and graphical models evaluated on NCI60 microarray gene expression data. Genome Informatics 11, 129–140 (2000)
18. Toh, H., Horimoto, K.: Inference of a genetic network by a combined approach of cluster analysis and graphical Gaussian modeling. Bioinformatics 18, 287–297 (2002)
19. Toh, H., Horimoto, K.: System for automatically inferring a genetic network from expression profiles. Journal of Biological Physics 28, 449–464 (2002)
20. Wu, X., Ye, Y., Subramanian, K.: Interactive analysis of gene interactions using graphical Gaussian model. In: ACM SIGKDD Workshop on Data Mining in Bioinformatics, vol. 3, pp. 63–69 (2003)
21. Dobra, A., Hans, C., Jones, B., Nevins, J., Yao, G., West, M.: Sparse graphical models for exploring gene expression data. Journal of Multivariate Analysis 90, 196–212 (2004)
22. Meinshausen, N., Bühlmann, P.: High dimensional graphs and variable selection with the lasso. The Annals of Statistics 34, 1436–1462 (2006)
23. Li, H., Gui, J.: Gradient directed regularization for sparse Gaussian concentration graphs, with applications to inference of genetic networks. Biostatistics 7, 302–317 (2006)
24. d'Aspremont, A., Banerjee, O., Ghaoui, L.E.: First-order methods for sparse covariance selection. SIAM Journal on Matrix Analysis and its Applications 30, 56–66 (2008)
25. Dawid, A.P.: Conditional independence in statistical theory. Journal of the Royal Statistical Society 41, 1–31 (1979)
26. Pearl, J.: Causality. Cambridge University Press, Cambridge (2000)
27. Diestel, R.: Graph Theory. 3rd edn. Graduate Texts in Mathematics, vol. 173. Springer, Heidelberg (2005)
28. Spirtes, P., Glymour, C., Scheines, R.: Causation, Prediction, and Search, 2nd edn. The MIT Press, Cambridge (2000)

29. Benjamini, Y., Hochberg, Y.: Controlling the false discovery rate: a practical and powerful approach to multiple testing. Journal of the Royal Statistical Society. Series B. Methodological 57, 289–300 (1995)
30. Opgen-Rhein, R., Schäfer, J., Strimmer., K.: GeneNet: Modeling and Inferring Gene Networks, R package version 1.2.1 (2007)
31. R Development Core Team: R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria (2008) ISBN 3-900051-07-0
32. Provost, F., Fawcett, T., Kohavi, R.: The case against accuracy estimation for comparing induction algorithms. In: Proceedings of the Fifteenth International Conference on Machine Learning, pp. 445–453. Morgan Kaufmann, San Francisco (1998)
33. Davis, J., Goadrich, M.: The relationship between precision-recall and ROC curves. In: Proceedings of the 23rd International Conference on Machine Learning, pp. 233–240 (2006)
34. van Rijsbergen, C.J.: Information Retrieval. Buttersworth, London (1979)

A Computational Method for Reconstructing Gapless Metabolic Networks

Esa Pitkänen[1,⋆], Ari Rantanen[2], Juho Rousu[1], and Esko Ukkonen[1]

[1] Department of Computer Science, University of Helsinki, Finland
esa.pitkanen@cs.helsinki.fi
[2] Institute of Molecular Systems Biology, ETH Zürich, Switzerland

Abstract. We propose a computational method for reconstructing metabolic networks. The method utilizes optimization techniques and graph traversal algorithms to discover a set of biochemical reactions that is most likely catalyzed by the enzymatic genes of the target organism. Unlike most existing computational methods for metabolic reconstruction, our method generates networks that are structurally consistent, or in other terms, gapless. As many analyses of metabolic networks, like flux balance analysis, require gapless networks as inputs, our network offers a more realistic basis for metabolic modelling than the existing automated reconstruction methods. It is easy to incorporate existing information, like knowledge about experimentally discovered metabolic reactions or metabolites into the process. Thus, our method can be used to assist in the manual curation of metabolic network models as it is able to suggest good candidate reactions for filling gaps in the existing network models. However, it is not necessary to assume any prior knowledge on composition of complete biochemical pathways in the network. We argue that this makes the method well-suited to analysis of organisms that might differ considerably from previously known organisms. We demonstrate the viability of our method by analysing the metabolic network of the well-known organism *Escherichia coli*.

1 Introduction

Structural models of cellular metabolism have proven to be very successful in answering many relevant biological research questions. The global organization of the metabolism can be discovered from the structural models [10]. In *constraint based modelling* of the metabolism [5,32,35,36,37,27], phenotypic behaviour, robustness and metabolic capabilities of an organism are analysed based on the structural models.

The process of creating a metabolic network model corresponding to the genome of the organism under study is called *metabolic reconstruction* [17]. In order to reconstruct a structural model of metabolism a complete set of metabolic reactions operating in the organism has to be discovered. These reactions define

⋆ Corresponding author.

the topology of an intertwined *metabolic network*, as the reactions are connected to each other by common substrate and product metabolites.

The most common method for reconstructing the structural model of a metabolic network is based on the combination of functional annotation of metabolic genes in the target organism by sequence similarity and manual utilization of literature information [13]. In the first technique, the functions of known enzymatic genes in a database are assigned to homologous genes in the target organism, thus adding the corresponding enzymatic reactions to the reconstructed metabolic network [17]. Knowledge on metabolic function annotation can also be derived from other sources, such as chromosome clustering [18], detection of protein fusion events [29], occurrence profiles [39], phylogenic profiles [7], and regulons [30]. The metabolic network is then curated by manually adding experimentally verified reactions and by fixing the observed inconsistencies.

A *reaction gap* is the most common example of an inconsistency in the reconstructed metabolic network [31]. A reaction gap occurs, when substrates of an internal reaction cannot be produced from the external substrates of the network. In constraint based modelling of metabolism, reaction velocities, or fluxes, in the metabolic network are explicitly modelled. As pathways with gaps cannot carry any flux in steady state conditions, the most accurate results are obtained when the analysis techniques are applied with gapless models.

Manual curation and gap-filling of genome-scale metabolic networks consisting of hundreds, even thousands of reactions is, however, very time-consuming and error-prone [16]. But, because of the indisputable usefulness metabolic network models in system-wide biological studies, manual curation of metabolic networks is considered to be worthwhile for the ever increasing number of organisms ranging from simple bacteria to mammalian cells [6,24,12,13,11,34]. Unfortunately, most reconstructed models still contain some gaps even after the manual curation.

Many computational methods and tools have been developed to assist the metabolic reconstruction and curation of genome-scale metabolic network models. For instance, the Pathway Tools software reconstructs metabolic networks by first determining how large portion of the enzymes of each pathway in a pathway database are present in the organism [23]. This is done by comparing a list of EC numbers given as input to the program against the EC numbers in the MetaCyc pathway database. EC numbers specify a functional classification of enzymatic activity [22]. Then the presence of each pathway is predicted. Roughly, the more enzymes appear to be present, the more confident we are that the pathway exists.

As a drawback, Pathway Tools cannot recreate truly novel subnetworks, as it is limited to the pathways in its database. The studies of central pathways such as TCA cycle, pentose phosphate pathway and glycolysis in microbial metabolic networks have indicated that the structure of these pathways varies in many organisms, from the textbook definition [9]. Thus, it is useful not to let hard-coded definitions of pathway structures based on previous knowledge affect the prediction of pathways in excess in newly sequenced organisms. In addition, Pathway Tools incorporates a second phase, where gaps left from the first phase are filled. The method is called the Pathway Hole Filler, which is a Naive Bayesian classifier combining evidence from genome structure such as chromosome clustering and others [19]. It predicts for each enzyme that was missing from the original reconstruction whether it should be added to the metabolic network.

Recently, an optimization based method called GapFill for filling gaps in a draft metabolic network was introduced by Kumar et al. [28]. In the method gaps are first discovered, then filled one by one by adding reactions to the network or by adding reverse directions for unidirectional reactions in the model (see Section 4 for a more detailed comparison to the present method).

It has been observed that reconstruction benefits from having multiple data sources combined [25]. In the approach of [26], experimental data is combined with the knowledge on metabolic network structure. By assuming that the genes encoding for enzymes on the same pathway are co-regulating, they try to find similarly expressed genes which could then be annotated with enzymatic functions on the same pathway. In [21], a draft metabolic network is automatically curated by minimizing the difference between the experimentally determined metabolic fluxes and the fluxes estimated by the flux balance analysis.

In the present paper, we introduce a new computational method for assisting in the task of metabolic reconstruction. In the method, optimization techniques and graph traversal algorithms are utilized to find a set of biochemical reactions that is most likely catalyzed by the genes of the target organism. The resulting network is guaranteed to be gapless, that is, each reaction in the reconstructed network is connected with a feasible metabolic pathway to external source metabolites, such as glucose. Thus, our method is able to provide feasible suggestions that are backed by the genomic evidence about the topology of the reconstructed metabolic network. This should significantly speed up the curation of metabolic network models. Our method does not assume any existing knowledge on specific pathways. In other words, pathway collections or logical rules coding pathway information are not needed. However, it is possible to utilise easily information about known reactions and metabolites to aid in the reconstruction process. For example, knowledge about experimentally observed metabolites and reactions can be easily exploited in the framework. Thus, the framework can be used both to produce *ab initio* reconstructions from the genome sequence data, as well as to fill the gaps in a draft model of a metabolic network.

To the authors' best knowledge, the presented method is the only computational technique for metabolic reconstruction that integrates the requirement for structural consistency and the search of reactions that are most likely to be catalyzed by the enzymes of the target organism to a single, global optimization task.

2 Methods

We formulate the metabolic reconstruction problem formally in the optimization context, a setting popular in the analysis of reconstructed metabolic networks [5,32]. We start by introducing first the key concept of reaction reachability, and then define metabolic reconstruction as the problem of finding a set of reachable reactions that is likely catalyzed by the genes of the target organism. Finally, a formulation in a mixed integer linear programming framework to solve the problem is given.

2.1 Reaction Reachability

Informally, a reaction gap occurs, when the metabolic network model is unable to supply all substrates of some reaction in the network. The model obviously contains an error: either the reaction in question should be removed, or the network should be modified to be able to provide the substrates.

We give a formulation of this idea by looking at reachability in an *and-or-graph* corresponding to the metabolic network under study [33]. In this context, we relate reactions with and-nodes and metabolites with or-nodes. Particularly, a reaction $r_i = (I_i, P_i)$ is specified by its substrates I_i and products P_i. We then investigate whether reactions can be reached in the network using the following rules, given network input metabolites A. The input metabolites A correspond to the nutrients of the organism under study. A typical example of inputs would include glucose as the carbon source.

- A reaction $r_i = (I_i, P_i)$ is *reachable* from A in R, if each metabolite in I_i is reachable from A in R.
- A metabolite m is *reachable* from A in R, if $m \in A$ or some reaction $r_j = (I_j, P_j)$ such that $m \in P_j$ is reachable from A in R.

We want all reactions in the reconstructed network to be reachable. This corresponds to not having any reaction gaps in the network.

Definition 1 (Feasible metabolic network). *A metabolic network consisting of reactions R is called* feasible *with respect to the source metabolites A, if and only if all reactions $r \in R$ are reachable from A in R.*

We will now formulate the discovery of the largest feasible network consisting of reactions in R with respect to source metabolites A as a linear programming problem. This formulation will be then utilized in the metabolic reconstruction. By v_i we denote the rate, or *flux* of reaction r_i in the network. In other words, v_i models the activity of reaction r_i. Now, the reachability of metabolites in the network can be coded by two constraints on their fluxes. The first constraint requires that the production of each metabolite not in A is equal or greater than its consumption,

$$\sum_i \delta_{ij} v_i - t_j \geq 0, \tag{1}$$

where δ_{ij} is -1, 1 or 0 depending on whether reaction r_i consumes, produces or does not use metabolite m_j, $v_i \geq 0$ is the rate of reaction r_i and $t_j \geq 0$ is the rate of the dilution reaction of metabolite m_j. The dilution reactions are required to ensure that there will be no cycles which are not connected with a path to network inputs A.

This is achieved by constraints requiring that whenever a metabolite m_j is produced by at least one reaction, the flux of the dilution reaction for that metabolite must be greater than zero,

$$t_j \geq \alpha \sum_{i \in P(m_j)} v_i, \tag{2}$$

where $P(m_j)$ is the set of reactions producing metabolite m_j and $0 < \alpha < 1$.

To see that metabolite dilution reactions ensure that there will be no cycles which are not connected to network inputs A, consider a cycle with reaction rates $v_i > 0$. Then, as the influx of some metabolite m_j would be greater than zero, also the corresponding dilution reaction flux $t_j > 0$. Since this dilution reaction participates to the steady state constraint of metabolite m_j, only a part of the flux to m_j can be used in fluxes leaving m_j. Thus, as all metabolites are constrained by (1), all reaction rates v_i (and t_j) in the cycle have to be zero.

We can find all reachable reactions in the network by maximising $\sum_{r_i \in R} v_i$ under the above constraints: the reaction r_i is reachable if and only if $v_i > 0$.

2.2 Metabolic Reconstruction Problem

Next we formulate metabolic reconstruction as a mixed integer linear programming problem involving the choice of reachable reactions to maximise a given score function over the reactions.

Problem 1 (Metabolic reconstruction). Given a set of reactions $\mathcal{R}$, a set of input metabolites A, a threshold value b, and a score function $f_b : \mathcal{R} \to \mathbb{R}$, find a subset R of $\mathcal{R}$ such that

1. $F_b(R) = \sum_{r \in R} f_b(r)$ is maximized and
2. the metabolic network with reactions R is feasible with respect to inputs A.

The corresponding mixed integer linear programming problem is the following.

Problem 2 (Metabolic reconstruction MILP).

$$\max_{\mathbf{x}} \sum_{r_i} f_b(r_i) x_i$$

such that

$$\frac{1}{N} x_i \quad \leq \quad v_i \tag{3}$$

$$v_i \quad \leq \quad M x_i, \tag{4}$$

$$\sum_i \delta_{ij} v_i - t_j \geq 0, \tag{5}$$

$$t_j \geq \alpha \sum_{r_i \in P(m_j)} v_i \texttt{ and} \tag{6}$$

$$x_i \in \mathbb{N} \tag{7}$$

where $x_i \in \{0, 1\}$ specifies whether reaction r_i is included in the result, N and M are appropriately large numbers, $0 < \alpha < 1$, v_i is the rate of reaction r_i, t_j is the dilution reaction rate corresponding to metabolite m_j and $P(m_j)$ is the set of reactions producing metabolite m_j.

Values for x_i, v_i and t_j are chosen during the optimization, while N, M and α are constants. Constraints (3) and (4) require that $x_i = 0 \Leftrightarrow v_i = 0$. Reaction cycles not connected to network inputs are disallowed with constraints (5) and (6).

2.3 Reaction and Network Scores

We derive scores f_b for reactions from the sequence homology evidence from the genome of the organism under study as follows. The score $f_b(r)$ for the reaction r is the homology score corresponding to the best match for a sequence in the target genome and a sequence which already has reaction r as its functional annotation. In other words, scores f_b can be calculated with

$$f_b(r) = \max_{s \in G} \max_{t \in U_r} S(s, t) - b, \tag{8}$$

where s is a sequence in the genome G of the organism under study and t is a sequence chosen from the set of sequences U_r annotated with reaction r in the protein database. Function $S(s, t)$ gives the degree of sequence similarity for sequences s and t; in this study, we use BLAST [2], the mainstay sequence search tool in bioinformatics.

We score a network simply as the sum of scores of reactions R of the network,

$$F_b(R) = \sum_{r \in R} f_b(r).$$

The threshold parameter $b > 0$ specifies a value dividing reactions into two groups. Reactions with a positive score $f_b(r)$ bring positive contribution to the overall network score $F_b(R)$. Thus, if we would not require that all reactions are reached from inputs, the optimization process would add all such reactions to the result, while leaving every reaction with a negative score out. In particular, the optimization may include a reaction with $f_b(r) < 0$ to the network only if the addition makes possible with respect to the feasibility constraint to add also reactions giving positive contribution to the score.

2.4 Divide-and-Conquer Approach

Unfortunately, the exact formulation of the above reconstruction problem is computationally infeasible for genome-scale instances. To tackle with the complexity, we revise the formulation by dividing the original problem into smaller subproblems and solving the subproblems individually. By this heuristic divide-and-conquer approach, we are able to solve realistic genome-scale instances. The central idea is to first find a good acyclic path that will be augmented into a feasible pathway, and repeat the process until we are satisfied with the outcome.

We first generate a random acyclic path $P = (r_1, \dots, r_n)$, $r_i \in \mathcal{R}$, starting from a randomly chosen source metabolite $a \in A$. The next reaction r_{i+1} on the path is selected by taking a random reaction that consumes a random product of r_i while ensuring that the path stays acyclic. Additionally, we require that $d_R(A, r_{i+1}) > d_R(A, r_i)$ for all reactions on the path, where $d_R(A, r)$ is the *production distance* from metabolites A to reaction r in the network R [33]. The production distance is the smallest diameter taken over all minimal feasible pathways from A to r, while diameter of a pathway is the length of the longest acyclic path in the pathway. In effect, the distance $d_R(A, r)$ is the minimum number of successive reactions needed to convert metabolites A into products of the reaction r. If $d_R(A, r)$ is defined, then there exists a feasible pathway connecting A to r. Finally, we take the longest path $P' = (r_1, \dots, r_k)$, $1 \leq k \leq n$, such that the path score is positive, $\sum_{1 \leq i \leq k} f_b(r_i) > 0$.

We then augment this initial path P' by adding all acyclic paths which start from sources A and end in some reaction $r \in P'$, requiring that a product of each reaction $r_i \in P'$, $1 \leq i < n$, is a substrate to the subsequent reaction $r_{i+1} \in P'$. Again, we require that the production distances of the reactions added increase from sources to reaction r. This is done to restrict the potentially very large search space while still ensuring that the reactions in pathway P' can be made feasible by the reactions in these acyclic pathways. Such paths can be found by backtracking from reactions $r_i \in P'$ towards reactions with smaller production distances until a source metabolite is found.

The reactions on the augmented pathway then comprise the reaction set $\mathcal{R}$ of the Metabolic Reconstruction Problem 1. The problem is formulated as a Problem 2 instance and solved. We obtain a result pathway that is feasible with respect to sources A. Further, the pathway is an optimal subset of the augmented pathway.

We repeat the above process of generating and augmenting an acyclic path for a fixed number of iterations, adding the metabolites on the previous result pathway to the set of source metabolites for the generation of the next pathway. Since each metabolite added to sources A is reachable from the initial sources, each successive pathway generated is also feasible. Finally, we take the union of all generated feasible pathways to be the final result.

2.5 Coding of Reaction and Metabolite Evidence

The above method allows for use of pre-existing knowledge of metabolic reactions and metabolites. If we have evidence that a particular enzyme operates in the cell, we can set a constraint to the optimisation problem stating that the reaction has to be present in the solution. Consequently, the reconstructed model would then contain both the reaction and a good-scoring combination of reactions needed to make the network with the added reaction feasible. In the same way evidence on the existence or absence of metabolites can be encoded into the optimisation problem. Such evidence is available via metabolomics experiments, for example.

Particularly, we can take a known metabolic network as a starting point and find best-scoring reactions to add to fill the gaps in the network. This initial network could be for example derived from sequence homology evidence, or be the result of further manual curation. In this way, our method would serve as an additional tool in aiding the curator.

In practice, however, we do not set a hard constraint for reaction existence as described above. Instead, the reaction is rescored to a high value and the optimisation problem is solved. If a reaction can be feasibly connected to the network, the solver adds a good-scoring pathway which precedes the reaction to the network. On the other hand, if the reaction cannot be feasibly connected, the solver returns a solution model that does not contain the reaction. In the formulation above, the solver would simply state that the problem is unsolvable, which is not desirable.

3 Experiments

We implemented our divide-and-conquer method, denoted SCAR for Structurally Consistent Automatic Reconstruction, as a script-driven program. A Python [38] script is used to generate subproblem instances for the optimiser. The mixed integer linear programming solver lp_solve [3], licensed under LGPL, was used to solve the individual subproblems. The Python script merged the results from solved subproblems as the final result.

To test our method, we reconstructed a feasible metabolic network for the well-known organism *Escherichia coli* from genome data. In particular, we were interested in seeing whether the feasibility constraint that was enforced in the reconstruction would cause the method to drop metabolic reactions which were known to be present in the metabolism.

We downloaded the *E. coli* K12 protein sequences from NCBI with the accession number U00096 [4,14]. There were 4331 coding subsequences (CDS) in total. As the enzyme database, we used UniProt version 9.3 [8] which contained 250296 protein sequences. We looked for the substring "(EC x.x.x.x)" in the textual description given for each sequence to find out which enzyme each sequence coded for. In this way, we obtained 101137 sequences with an EC number[1].

As the reaction database $\mathcal{R}$, we used MetaCyc version 10.6 [15] with 6241 reactions of which 5255 had associated EC numbers. In what follows, this set of reactions is referred to as the universal metabolic network.

[1] Each Uniprot entry specifies the enzymatic reaction only at EC number level. EC number can specify more than one concrete reaction, such as EC 2.4.1.1 which corresponds to phosphorylases operating on various sugar molecules. When the EC number of an enzyme corresponds to multiple reactions in the reaction database, each matching reaction receives the sequence homology score from that enzyme.

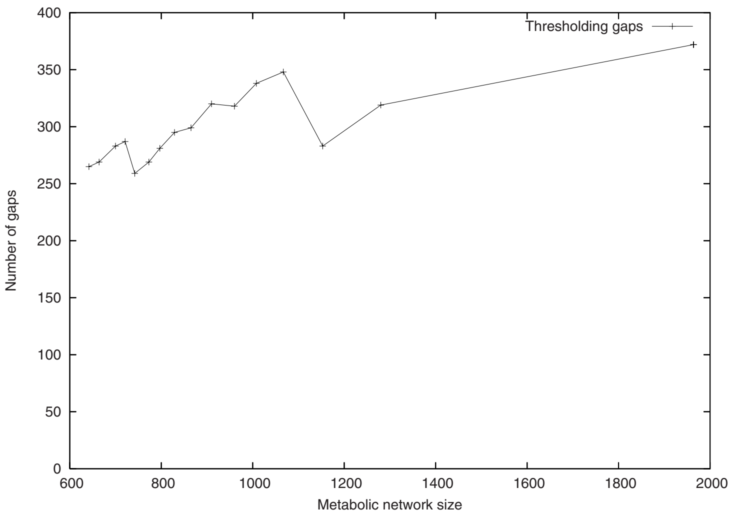


Fig. 1. Number of gaps, or reactions not connected to sources with a feasible pathway, in metabolic networks obtained with the thresholding method

We compared the *E. coli* protein sequences against all protein sequences from UniProt with an EC number using BLAST [2]. E-value cutoff of BLAST was set to 10 to detect remote homologs. From the BLAST scores obtained this way for each sequence pair, we derived the reaction scores for MetaCyc reactions using the equation (8).

Our method was compared against the baseline reconstruction method which is based only on sequence homology scoring. In this *thresholding method*, a reaction is chosen into the reconstruction only if the reaction score is higher than some specified threshold. Naturally, the resulting metabolic network contains gaps because the feasibility is not enforced in any way.

We reconstructed a metabolic network for *E. coli* using our method on the data discussed above. We experimented with different threshold value parameters, varying the value from 0 to 400 in increments of 25. Different values of threshold parameter model the degree of confidence we want the reconstructed model to obtain from the sequence homology: with high values of the parameter, smaller subnetworks with higher sequence homology evidence are produced, while with lower parameter values larger networks with less sequence based evidence are reconstructed. As the set of source metabolites A, we used 111 metabolites including glucose, which is the main carbon source of *E. coli*, and cofactor molecules that are known to be present in the organism in abundance, such as ATP, NAD and CO_2.

The number of gaps in metabolic networks obtained with the thresholding method is shown in Figure 1. This value can be seen as a measure of the manual work needed to curate the initial reconstruction by hand.

Table 1. Summary of results for our reconstruction method (**S**CAR) and the thresholding method (**T**HRE) for the different threshold parameter values showing the numbers of reactions in results, feasibility gaps in THRE result (#Gaps), reactions with negative scores filling the gaps in the SCAR result (#Fillers), average reaction score (Score) and average score for gap filling reactions in the SCAR result (Fill score). The scores reported are $f_b(r)$ scores with $b = 0$. Only the increments of 50 are shown in the threshold value.

Threshold	#Reactions (S / T)	#Gaps	#Fillers	Score (S/T)	Fill score
0	1354/1964	372	338	292.7/336.7	0
50	938/1280	319	208	410.4/498.3	7.1
100	809/1067	348	174	469.1/583.4	13.8
150	755/960	318	175	487.3/634.5	18.0
200	649/865	299	132	548.0/685.1	28.0
250	597/796	281	126	580.7/724.4	34.4
300	551/742	259	117	597.7/757.5	41.9
350	500/700	283	101	637.1/783.5	57.4
400	469/642	265	97	659.1/820.6	97.8

A summary of the reconstruction results for both our method and the thresholding method is shown in Table 1. Our method produces smaller networks than the thresholding method as some reactions with a high score are not included in the result. This happens when the addition of such a reaction would also require that negative-scoring reactions were added to the network due to the feasibility constraint and the score of the resulting pathway would be negative. In this case, the pathway is not added to the network. On the average, 20–25% of reactions in our method's result had a negative score meaning that they were used to fill the gaps in the network. Similarly, the average reaction score is smaller compared to

Table 2. Reactions included in the EcoCyc database that were not found by the thresholding method (threshold value 200) because of low reaction scores, but which were included in the SCAR reconstruction. MetaCyc reaction identifiers without the trailing _RXN shown. A dash signifies a missing EC number or name in MetaCyc.

Reaction	EC	Name	Score
1.1.1.283	1.1.1.283	Methylglyoxal reductase (NADPH-dependent)	37
1.13.11.16	1.13.11.16	3-carboxyethylcatechol 2,3-dioxygenase	0
BETA-PHOSPHOGLUCOMUTASE	5.4.2.6	β-phosphoglucomutase	186
CHORPYRLY	4.-.-.-	-	40
GALACTITOLPDEHYD	1.1.1.-	-	0
GLUCONOLACT	3.1.1.17	Gluconolactonase	0
NAG6PDEACET	3.5.1.25	N-acetylglucosamine-6-phosphate deacetylase	0
OXALODECARB	4.1.1.3	Oxaloacetate decarboxylase	52
PDXJ	2.6.99.2	-	0
RXN-821	-	-	0
RXN0-313	4.-.-.-	-	0
RXN0-5116	2.7.1.16	-	0
TAGAALDOL	4.1.2.40	Tagatose-bisphosphate aldolase	69
TAGAKIN	2.7.1.144	Tagatose-6-phosphate kinase	124

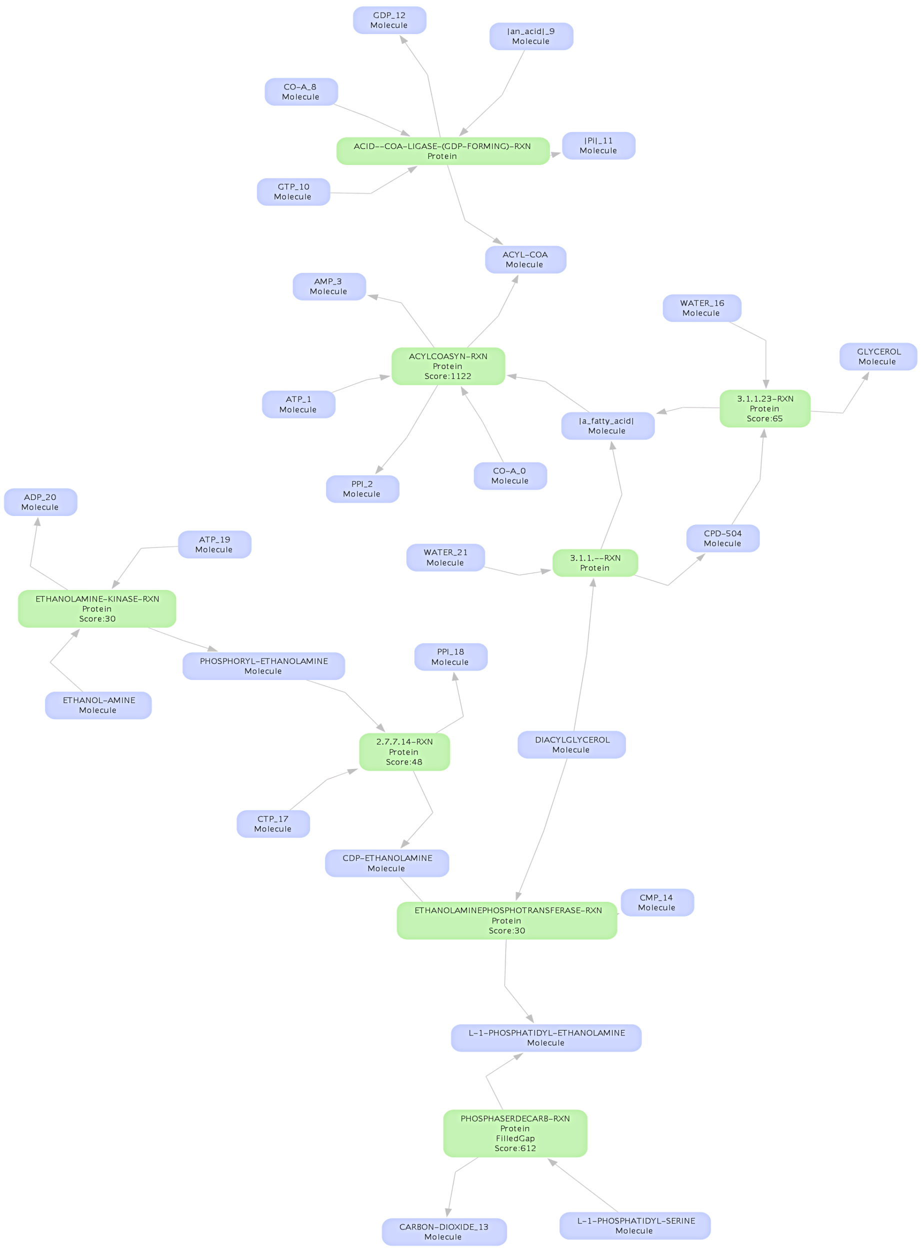

GDP_12 Molecule
|an_acid|_9 Molecule
CO-A_8 Molecule
ACID--COA-LIGASE-(GDP-FORMING)-RXN Protein
|Pi|_11 Molecule
GTP_10 Molecule
ACYL-COA Molecule
AMP_3 Molecule
WATER_16 Molecule
GLYCEROL Molecule
ACYLCOASYN-RXN Protein Score:1122
ATP_1 Molecule
3.1.1.23-RXN Protein Score:65
|a_fatty_acid| Molecule
PPI_2 Molecule
CO-A_0 Molecule
ADP_20 Molecule
ATP_19 Molecule
CPD-504 Molecule
WATER_21 Molecule
3.1.1.--RXN Protein
ETHANOLAMINE-KINASE-RXN Protein Score:30
PHOSPHORYL-ETHANOLAMINE Molecule
PPI_18 Molecule
ETHANOL-AMINE Molecule
2.7.7.14-RXN Protein Score:48
DIACYLGLYCEROL Molecule
CTP_17 Molecule
CDP-ETHANOLAMINE Molecule
CMP_14 Molecule
ETHANOLAMINEPHOSPHOTRANSFERASE-RXN Protein Score:30
L-1-PHOSPHATIDYL-ETHANOLAMINE Molecule
PHOSPHASERDECARB-RXN Protein FilledGap Score:612
CARBON-DIOXIDE_13 Molecule
L-1-PHOSPHATIDYL-SERINE Molecule

Fig. 2. Phosphatidylserine decarboxylase reaction (EC 4.1.1.65) and the network in a SCAR reconstruction that fills the gap present in the thresholding reconstruction with threshold value 200. Note that each reaction is considered to be bidirectional, thus the directionality of arrows carries no other meaning than separating substrates and products of the same reaction. The figure was generated with the `BMVis` visualisation tool [1].

the thresholding method as expected, because the method repairs the gaps with low-scoring reactions.

Our method took about 150 seconds on the average to run per reconstruction on a 1.6 GHz Pentium M processor. The running time can be tuned by setting the maximum number of initial paths generated. In this experiment, we used a maximum of 6000 paths which was experimentally verified to be enough for the network score $F_b(R)$ to converge.

At threshold parameter value $b = 200$, our method was able to recover 14 reactions missed by the thresholding method but found in the EcoCyc database. EcoCyc is a comprehensive, manually curated database of *E. coli* metabolism [24]. These reactions are shown in Table 2. As the score of each reaction is below the threshold value, the reason why these reactions were added to the result was to satisfy the feasibility constraint.

As an example of a gap that exists in the thresholded reconstruction but which has been filled in a SCAR reconstruction, consider the reaction phosphatidylserine decarboxylase (EC 4.1.1.65) which received a score $f_b(r) = 412$ from the BLAST alignment of *E.coli* sequence GI 1790604 and UniProt sequence P0A8K4, with $b = 200$. Our method succeeded in repairing the gap; the reaction and a feasible subnetwork that repairs the gap is shown in Figure 2.

4 Discussion

In this article we introduce a computational method to assist the reconstruction of a metabolic network of an organism based on its genome information. The method is based on optimization techniques and graph traversal algorithms. As a distinctive feature, the presented method combines the search of reactions that are most likely catalyzed by the genes of the target organism and the filling the gaps in the reconstructed metabolic network to a single computational step. It is easy to incorporate experimental evidence, such as information about experimentally observed metabolites or reactions with the method to improve the quality of the reconstructed metabolic network models.

As the present method constructs gapless metabolic networks, the reconstructed models can contain also reactions that have no known catalyzing enzyme, as long as these reactions improve the feasibility of the model. The advantages of this feature are twofold. First, we can augment the reaction database utilized by the reconstruction algorithm with computer-generated, or hypothetical, reactions [20] without the knowledge of enzymes possibly catalyzing these reactions. Second, the identification of an unannotated but structurally necessary reaction can serve as a hint for finding the gene responsible for that particular function. This capability is beyond most current computational methods for metabolic reconstruction.

The present method shares some properties with recently introduced method called GapFill [28]. The main difference between GapFill and the present method is found from the weaker definition of the reaction gap applied in the GapFill.

In GapFill, reaction gaps are thought to be removed as soon as the network produces substrate metabolites for each of its internal reactions – even if the network is unable to produce these metabolites from its external substrates. This formulation easily leads to situations where the gaps in a draft network are filled by small cycles where reactions produce substrates for each others, but that are disconnected from the rest of the network. In the present method, on the other hand, we require the substrates of each reaction in the network have to be produced from the external sources. We argue that our stronger definition of the reaction gaps is biologically more relevant: the network model where some metabolites cannot be produced from the external sources is clearly incomplete. For example in the most tasks of constraint based modelling of metabolism, the reactions whose substrates cannot be produced from the external substrates are irrelevant, and can be removed from the model as a preprocessing step. Furthermore, in GapFill the gaps in the network are filled locally, and only the number of reactions needed to fill the gaps is considered in the optimization. The present method, on the other hand, looks for globally optimal modifications to simultaneously fill all the gaps, taking also the available genomic evidence for the existence of gap-filling reactions in the target organism into account.

To conclude, we believe that the present method is able to produce useful suggestions about the structure of a metabolic network to guide a domain expert in a very time-consuming task of metabolic reconstruction. To further improve the accuracy of the reconstructed metabolic network models, we will investigate alternative ways of scoring the reactions in a database. These alternatives contain more advanced methods for detecting homologous sequences between the enzymes in a database and the genome of the target organism that might share a function, as well as the inclusion of the other type of data, such as whole-cell metabolome measurements, into the computation of reaction scores.

Acknowledgments

We would like to thank Antti Tani for the initial computational experiments, Taneli Mielikäinen and Paula Jouhten for valuable discussions, the BioMine project at the Department of Computer Science, University of Helsinki for the `BMVis` network visualisation tool, and the reviewers for their comments that helped to improve the manuscript.

References

1. BMVis graph visualisation tool. Department of Computer Science, University of Helsinki, `http://www.cs.helsinki.fi/group/biomine`
2. Altschul, S.F., Madden, T.L., Schaffer, A.A., Zhang, J., Zhang, Z., Miller, W., Lipman, D.J.: Gapped blast and psi-blast: a new generation of protein database search programs. Nucleic Acids Research 25, 3389–3402 (1997)

3. Berkelaar, M., Eikland, K., Notebaert, P.: lp_solve: Open source (mixed-integer) linear programming system, Multi-platform, pure ANSI C / POSIX source code, Lex/Yacc based parsing. Version 5.1.0.0 dated 1 May 2004. GNU LGPL (Lesser General Public Licence) (2005), http://groups.yahoo.com/group/lp_solve
4. Blattner, F.R., Plunkett 3rd, G., Bloch, C.A., Perna, N.T., Burland, V., Riley, M., Collado-Vides, J., Glasner, J.D., Rode, C.K., Mayhew, G.F., Gregor, J., Davis, N.W., Kirkpatrick, H.A., Goeden, M.A., Rose, D.J., Mau, B., Shao, Y.: The complete genome sequence of escherichia coli k-12. Science 277, 1453–1474 (1997)
5. Burgard, A.P., Nikolaev, E.V., Schilling, C.H., Maranas, C.D.: Flux coupling analysis of genome-scale metabolic network reconstructions. Genome Research 14(2), 301–312 (2004)
6. Caspi, R., Foerster, H., Fulcher, C.A., Hopkinson, R., Ingraham, J., Kaipa, P., Krummenacker, M., Paley, S., Pick, J., Rhee, S.Y., Tissier, C., Zhang, P., Karp, P.D.: Metacyc: A multiorganism database of metabolic pathways and enzymes. Nucleic Acids Research 516, D511–D516 (2006)
7. Chen, L., Vitkup, D.: Predicting genes for orphan metabolic activities using phylogenetic profiles. Genome Biology 7(R17), 1777–1782 (2006)
8. The UniProt Consortium. The universal protein resource (uniprot). Nucleic Acids Research 35, D193–D197 (2007)
9. Cordwell, S.J.: Microbial genomes and "missing" enzymes: redefining biochemical pathways. Arch Microbiol 172, 269–279 (1999)
10. Csete, M., Doyle, J.: Bow ties, metabolism and disease. TRENDS in Biotechnology 22(9), 446–450 (2004)
11. Duarte, N., Becker, S., Jamshidi, N., Thiele, I., Mo, M., Vo, T., Srivas, R., Palsson, B.: Global reconstruction of the human metabolic network based on genomic and bibliomic data. Proceedings of National Academy of Science, USA 104(6), 1777–1782 (2007)
12. Duarte, N., Herrgård, M., Palsson, B.: Reconstruction and validation of *Saccharomyces cerevisiae* iND750, a fully compartmentalized genome-scale metabolic model. Genome Research 14(7), 1298–1309 (2004)
13. Edwards, J., Palsson, B.: The Escherichia coli MG1655 in silico metabolic genotype: its definition, characteristics, and capabilities. Proceedings of National Academy of Science, USA 97(10), 5528–5533 (2000)
14. Riley, M., et al.: Escherichia coli k-12: a cooperatively developed annotation snapshot. Nucleic Acids Res. 34(1), 1–9 (2006)
15. Caspi, R., et al.: The metacyc database of metabolic pathways and enzymes and the biocyc collection of pathway/genome databases. Nucleic Acids Research 36(Database issue), 623–631 (2008)
16. Förster, J., Famili, I., Fu, P., Palsson, B., Nielsen, J.: Genome-scale reconstruction of the Saccharomyces cerevisiae metabolic network. Genome Research 13(2), 244–253 (2003)
17. Francke, C., Siezen, R.J., Teusink, B.: Reconstructing the metabolic network of a bacterium from its genome. Trends Microbiol 13(11), 550–558 (2005)
18. Galperin, M., Koonin, E.: Who's your neighbor? new computational approaches for functional genomics. Nat. Biotechnol. 18, 609–613 (2000)
19. Green, M., Karp, P.D.: A Bayesian method for identifying missing enzymes in predicted metabolic pathway databases. BMC Bioinformatics 5(76) (2004)
20. Hatzimanikatis, V., Li, C., Ionita, J.A., Henry, C.S., Jankowski, M.D., Broadbelt, L.J.: Exploring the diversity of complex metabolic networks. Bioinformatics 21(8) (2005)

21. Herrgård, M., Fong, S., Palsson, B.: Identification of genome-scale metabolic network models using experimentally measured flux profiles. PLoS Computational Biology 2(7), 72 (2006)
22. IUBMB. Enzyme Nomenclature. Academic Press (1992)
23. Karp, P., Paley, S., Romero, P.: The pathway tools software. Bioinformatics 232, S225–S232 (2002)
24. Keseler, I.M., Collado-Vides, J., Gama-Castro, S., Ingraham, J., Paley, S., Paulsen, I.T., Peralta-Gil, M., Karp, P.D.: EcoCyc: A comprehensive database resource for Escherichia coli. Nucleic Acids Research 337, D334–D337 (2005)
25. Kharchenko, P., Chen, L., Freund, Y., Vitkup, D., Church, G.M.: Identifying metabolic enzymes with multiple types of association evidence. BMC Bioinformatics 7(177) (2006)
26. Kharchenko, P., Church, G.M., Vitkup, D.: Filling gaps in a metabolic network using expression information. Bioinformatics 185, I178–I185 (2004)
27. Kim, P.-J., Lee, D.-Y., Kim, T., Lee, K., Jeong, H., Lee, S., Park, S.: Metabolite essentiality elucidates robustness of escherichia coli metabolism. Proceedings of National Academy of Science, USA 104(34), 13638–13642 (2007)
28. Kumar, V., Dasika, M., Maranas, C.: Optimization based automated curation of metabolic reconstructions. BMC Bioinformatics 8(212), 13638–13642 (2007)
29. Marcotte, E.: Computational genetics: finding protein function by nonhomology methods. Curr. Opin. Struct. Biol. 10, 359–365 (2000)
30. McGuire, A., Hughes, J., Church, G.: Conservation of DNA regulatory motifs and discovery of new motifs in microbial genomes. Genome Res. 10, 744–757 (2000)
31. Osterman, A., Overbeek, R.: Missing genes in metabolic pathways: a comparative genomics approach. Current Opinion in Chemical Biology 7, 238–251 (2003)
32. Pharkya, P., Maranas, C.D.: An optimization framework for identifying reaction activation/inhibition or elimination candidates for overproduction in microbial systems. Metabolic Engineering 8(1), 1–13 (2006)
33. Pitkänen, E., Rantanen, A., Rousu, J., Ukkonen, E.: Finding feasible pathways in metabolic networks. In: Bozanis, P., Houstis, E.N. (eds.) PCI 2005. LNCS, vol. 3746, pp. 123–133. Springer, Heidelberg (2005)
34. Resendis-Antonio, O., Reed, J., Encarnación, S., Collado-Vides, J., Palsson, B.: Metabolic reconstruction and modeling of nitrogen fixation in Rhizobium etli. PLoS Computational Biology 3(10), 13638–13642 (2007)
35. Schilling, C.H., Letscher, D., Palsson, B.: Theory for the systemic definition of metabolic pathways and their use in interpreting metabolic function from a pathway-oriented perspective. Journal of Theoretical Biology 203(3), 228–248 (2003)
36. Schuster, S., Fell, D.A., Dandekar, T.: A general definition of metabolic pathways useful for systematic organization and analysis of complex metabolic network. Nature Biotechnology 18, 326–332 (2000)
37. Stelling, J., Klamt, S., Bettenbrock, K., Schuster, S., Gilles, E.D.: Metabolic network structure determines key aspects of functionality and regulation. Nature 420, 190–193 (2002)
38. van Rossum, G., Drake Jr., F.L.: An Introduction to Python. Network Theory Ltd. (2006)
39. Wolf, Y., Rogozin, I., Kondrashov, A., Koonin, E.: Genome alignment, evolution of prokaryotic genome organization, and prediction of gene function using genomic context. Genome Res. 11, 356–372 (2001)

Finding Frequent Subgraphs in Biological Networks Via Maximal Item Sets

Hans Zantema[1,3], Stefan Wagemans[1], and Dragan Bošnački[2]

[1] Department of Computer Science, TU Eindhoven, P.O. Box 513, 5600 MB Eindhoven, The Netherlands
H.Zantema@tue.nl

[2] Department of Biomedical Engineering, TU Eindhoven, P.O. Box 513, 5600 MB Eindhoven, The Netherlands
D.Bosnacki@tue.nl

[3] Institute for Computing and Information Sciences, Radboud University Nijmegen, P.O. Box 9010, 6500 GL Nijmegen, The Netherlands

Abstract. We describe an improvement of an algorithm for detecting frequently occurring patterns and modules in biological networks. The improvement is based on the observation that the problem of finding frequent network parts can be reduced to the problem of finding maximal frequent item sets (MFI). The MFI problem is a classical problem in the data mining community and there exist numerous efficient tools for it, most of them publicly available. We apply MFI tools to find frequent subgraphs in metabolic pathways from the KEGG database. Our experimental results show that, compared to the existing specialized tools for frequent subgraphs detection, the MFI tools coupled with an adequate postprocessing are much more efficient with regard to the running time and the size of the frequent graphs.

1 Introduction

Understanding the structure and dynamics of biological networks is one of the main challenges of bioinformatics and systems biology. Insight into the network organization and behavior helps us to understand the biological processes inside the cell. With the increase of network related experimental data effective methods and models to analyze this data are needed.

A natural way to model biological networks is by means of graphs. The problem of finding frequent subgraphs in a collection of graphs representing biological networks was introduced by Koyutürk, Grama and Szpankowski [8]. In a sense, the problem is analogous to finding common subsequences in a collection of biological sequences that usually represent genes. Specialized tools for such sequence alignment, like CLUSTAL and BLAST, have been used routinely by researchers. The similarity between gene sequences detected in this way can be used to derive different kinds of structural, evolutionary, and functional information.

Similarly, detecting common parts in biological networks within one organism or across different organisms can provide insight into common motifs of cellular interactions, organization of functional modules, relationships and interactions between sequences, and patterns of gene regulation.

The problem of finding all maximal frequent subgraphs can be formulated as follows: given a collection of graphs and a natural number n, called threshold, find all subgraphs that are contained in at least n graphs of the collection. An additional condition is that the subgraphs are connected, i.e., between each two nodes a, b of such a subgraph there is a path either from a to b or vice versa. Moreover, we require that the graphs are maximal in the sense that they are not contained in another frequent subgraph.

One can consider the graphs of the given collection as sets of edges. In this way the problem of finding all maximal frequent subgraphs is reduced to the problem of finding maximal frequent subsets that correspond to graphs consisting of one connected component. Like the subgraphs above, a subset is frequent and maximal if it is contained in more than a threshold number of sets of the collection and it is not a subset of any other maximal frequent set. Thus, in our approach we remove the connectedness requirement and we simply look for unrestricted maximal frequent subsets. (The latter correspond to graphs that are not necessarily connected.)

Our improvement is based on the fact that one can recover all maximal frequent *connected* subgraphs if one knows all maximal frequent (unrestricted) subgraphs. Finding the latter boils down to the problem of finding so called maximal frequent item sets (MFI), which is a classical problem in the area of data mining. Thus, in the first step our method finds all maximal frequent subsets. In the second step the collection of maximal frequent subsets (item sets) is processed to find all maximal frequent subgraphs.

The algorithm by Koyutürk et al. in [8] is also based on the maximal frequent item sets. This algorithm seeks to improve the MFI algorithm by taking into account the additional information that the item sets that we are looking for are connected graphs. In contrast, we use the MFI algorithm(s) as a black box and obtain the final collection of subgraphs by a postprocessing of the output of the MFI algorithm. In this way, one can take a full advantage of the highly efficient algorithms and tools that are developed for data mining [2]. It turns out that these fast algorithms fully compensate the additional overhead incurred by the postprocessing.

Our experiments with biological networks extracted from the KEGG database [12] show that one can achieve a significant speed-up compared to the algorithm in [8]. More concretely, we applied our algorithm to graphs extracted from metabolic networks as it was done in [8]. These graphs have enzymes as nodes, while the edges represent functional relations between the enzymes. In particular, there is an edge between two enzymes if the corresponding reactions that they catalyze are related. This means that the output of one reaction is an input for the other. Although some information is inevitably lost by such a representation, the benefit is that the graph mining problem is simplified significantly. At the same time, in practice, the enzyme graphs can be related back to the metabolic pathways in a straightforward way.

The choice to use enzyme graphs derived from KEGG metabolic pathways is mainly motivated by the fact that we want to compare our approach with the existing state of the art method from [8]. This being said, our approach is much more general and it can be applied to other kinds of biological networks as well.

To summarize, the main contributions of this paper are:

– Reduction of the problem of finding maximal frequent subgraphs to finding maximal frequent item sets.
– As a result we are able to find frequent subgraphs for smaller thresholds and much faster than the existing state of the art algorithm/tool by Koyutürk et al. [8].
– We apply the algorithms to the new data set extracted from the latest version of KEGG. The latter is significantly larger than the version of the KEGG data used in [8].

Related work. There exists extensive literature on mining frequent patterns in networks and in particular on mining frequent subgraphs which is beyond the scope of this paper. Here we do not discuss also the class of approaches that defines the mining problem as one of finding isomorphic subgraphs (e.g., [11,16]), since they turn out to be too computationally expensive for the kind of biological applications that we are targeting.

As we already emphasized, the main difference between our approach and the one in [8] is that they adjust the MFI algorithm trying to exploit the connectedness of the item sets/graphs. In contrast, we are using the MFI algorithm unchanged and apply postprocessing to extract the maximal frequent subgraph from its output. In [9] a new algorithm is presented that combines the algorithm in [8] with a graph simplification technique, called ortholog graph contraction. This algorithm was used for detecting conserved patterns in biological networks. The MFS algorithm in [8] is a modification of the MFI algorithm in [3]. The latter also gives a nice overview of the state of the art MFI algorithms and tools. An MFS algorithm which is extension of an a priori mining technique for MFI was presented also in [6]. This algorithm works with the adjacency matrices of the graphs. As a drawback it tends to be expensive for large sparse graphs that occur in biological applications. Another algorithm for finding MFS was presented in [1]. Because of the relatively straightforward enumerative approach, this algorithm cannot handle larger graph collections. Other algorithms for MFS were given in [4,5,15]. The comparison of these works with our approach remains for future work.

Paper layout. In Section 2 we introduce graph models of metabolic pathways that are used in the experimental evaluation of our approach. Section 3 gives the theoretical background and results on maximal frequent items and maximal frequent subgraphs. Section 4 describes the main contribution of the paper – the algorithm for finding maximal subgraphs. In Section 5 we report on some experimental results with a prototype implementation of our algorithm. The final section concludes the paper and points to some future work.

2 Graph Models of Metabolic Pathways

In this section we define the graphs corresponding to metabolic pathways along the lines of [8].

Each pathway can be seen as a set of chemical reactions that are catalyzed by enzymes. Each reaction involves a set of compounds called metabolites. Metabolites that are an input of the reaction are called substrates. The output metabolites are called products. This intuition behind the metabolic pathways is captured formally in the following definition.

Definition 1. *A metabolic pathway* (M, Z, R) *is a collection of metabolites* M*, enzymes* Z *and reactions* R*. Each reaction* $r \in R$ *is associated with a set of enzymes* $Z(r) \subseteq Z$*, a set of substrates* $S(r) \subseteq M$*, and a set of products* $T(r) \subseteq M$*.*

The reactions may be connected with each other in the sense that products of one reaction can be substrates for another. In the applications described in this paper, we focus on capturing interconnections between enzymes. To this end, we define directed graphs that have enzymes as nodes and where the edges represent interactions between the enzymes. As the enzymes are closely related to the reactions, we consider that there is a relation (edge in our graph) if there exists a connection between the reactions, i.e., if a reaction corresponding to the target enzyme consumes a product of a reaction corresponding to the source enzyme. More formally,

Definition 2. *With a given metabolic pathway* (M, Z, R)*, we associate a directed* enzyme graph (V, E)*, where* V *and* E *are the sets of vertices and edges, respectively, defined as follows:*

- $V = Z$,
- $E \subseteq V \times V$ *and* $(z, z') \in E$ *if and only if for some* $r, r' \in R$ *it holds that*
  - $z \in Z(r), z' \in Z(r')$, *and*
  - $T(r) \cap S(r') \neq \emptyset$.

By representing the version of a given metabolic pathway for each organism of some database [7,10],e.g. KEGG, one obtains a collection of enzyme graphs. All graphs in such a collection have the same set of vertices V and they differ only in the edge set E.

An interesting application of the enzyme graphs is in finding frequently occurring patterns in metabolic pathways. Identifying those patterns can provide insight into the evolution, structure and function of the metabolic pathways. (See [8] for a more extensive discussion on the motivation.) The problem of finding frequently occurring patterns can be translated into a problem of finding maximal frequent subgraphs of the graphs in the collection. We also require that these graphs are connected, since we are interested in related parts of the metabolic pathway. The maximality requirement ensures that we obtain a compact representation of the frequent patterns in the sense that no pattern is contained in another pattern. We formalize these concepts in the next section.

The above described conversion of metabolic pathways into enzyme graphs simplifies the algorithmic analysis significantly. Nevertheless, the obtained results are still biologically relevant. With the conversion some information is inevitably lost. For instance, two different metabolic pathways that feature the same enzymes and differ only in their reactions can be associated with the same enzyme graph. Thus, strictly speaking the conversion back from an enzyme graph to a metabolic pathway is not unique. (See [8] for an example of metabolic pathways and their corresponding graphs.) However, in practice usually it is easy to recover the interesting biological information from the enzyme graphs, i.e. to revert back to the original metabolic pathway graphs.

3 Maximal Frequent Subgraphs and Maximal Frequent Item Sets

This section gives the basic definitions and concepts about maximal frequent subgraphs and maximal frequent item sets. It also presents some original results that are used later on in the design of the algorithm for finding maximal frequent subgraphs.

A graph (V, E) is given, where V is the set of nodes (vertices) and $E \subseteq V \times V$ is the set of edges. Also a set of subgraphs (V, E_i) is given for $i = 1, \ldots, n$, satisfying $E_i \subseteq E$ for all $i = 1, \ldots, n$. In all our observations on graphs the set V of nodes is fixed. The only difference in the graphs is in the set of edges, always being a subset of E. In the sequel we identify a graph (V, E') with its set of edges E'.

Although the graphs are directed, the direction of the arrows does not play a role and the graphs can be considered to be undirected without affecting the algorithms that we give in the sequel. Consequently, two edges (v, v') and (w, w') are said to be *connected* if $\{v, v'\} \cap \{w, w'\} \neq \emptyset$. A graph E' is said to be *connected* if for every two edges $e, e' \in E'$ there exist edges $e_1, \ldots, e_k \in E'$ for some k such that $e = e_1$, $e_k = e'$ and the two edges e_i and e_{i+1} are connected for all $i = 1, \ldots, k-1$. It is easy to see that this coincides with the usual definition that requires that between any two nodes of the graph there is a path.

Given a threshold t, a graph E' is said to be a *frequent item set* if

$$\#\{i \in \{1, \ldots, n\} \mid E' \subseteq E_i\} \geq t.$$

where t is a natural number and $\#S$ denotes the number of elements in the set S. Given a threshold t, a graph E' is said to be a *frequent subgraph* if it is a connected frequent item set.

A graph E' is said to be a *maximal frequent item set* if it is a frequent item set and there is no frequent item set E'' distinct from E' such that $E' \subseteq E'' \subseteq E$. Likewise, a graph E' is said to be a *maximal frequent subgraph* if it is a frequent subgraph and there is no frequent subgraph E'' distinct from E' such that $E' \subseteq E'' \subseteq E$. Finally, a graph E'' is said to be a *component* of a (possibly not connected) graph E' if $E'' \subseteq E'$, E'' is connected and there is no connected graph $\tilde{E}$ distinct from E'' such that $E'' \subseteq \tilde{E} \subseteq E'$.

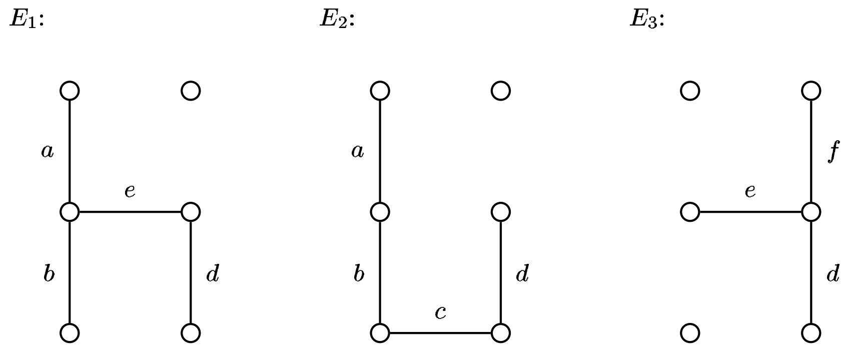


Fig. 1. A graph collection example

We illustrate these notions by the following example. Let us consider the graphs in Fig. 1.

We choose the threshold to be 2. Then there are two maximal frequent item sets: $\{a, b, d\}$ and $\{d, e\}$. Note that the maximal frequent item sets $\{a, b, d\}$ is not connected. There are also two maximal frequent subgraphs: $\{a, b\}$ and $\{d, e\}$.

Our algorithm and its runtime improvement upon earlier algorithms is based on the following observations:

– Components of a graph are easily found for any given graph by a standard algorithm.
– Maximal frequent item sets can be found much faster than maximal frequent subgraphs.
– Maximal frequent subgraphs can be obtained by decomposing maximal frequent item sets into components, as is stated by the following theorem.

Theorem 1. *Every maximal frequent subgraph is a component of a maximal frequent item set.*

Proof. Take an arbitrary maximal frequent subgraph E'; we have to prove that it is a component of a maximal frequent item set. By definition E' is frequent. Next obtain E'' from E' by adding edges outside E' one by one as long as the new set is frequent. So E'' is frequent, and no other edge can be added to E'' by which it remains frequent, so E'' is a maximal frequent item set. Now it remains to prove that E' is a component of E''. By definition E' is connected; we have to prove that no connected graph $\tilde{E}$ distinct from E' exists such that $E' \subseteq \tilde{E} \subseteq E''$. Assume it exists, then it is frequent since $\tilde{E} \subseteq E''$ and E'' is frequent. Moreover it is connected, so $\tilde{E}$ is a frequent subgraph. This contradicts the assumption that E' is a maximal frequent subgraph. $\square$

The following theorem is useful to establish which of the components of a maximal frequent item set are maximal frequent subgraphs.

Theorem 2. *Let U be a set of frequent subgraphs such that every maximal frequent subgraph is in U. Then the set of maximal frequent subgraphs is equal to*

$$\{E' \in U \mid \text{ there is no } \tilde{E} \in U \text{ distinct from } E' \text{ satisfying } E' \subseteq \tilde{E}\}.$$

Proof. Assume E' is a maximal frequent subgraph. Then it is in U. Due to maximality there is no strictly greater frequent subgraph $\tilde{E}$, by which E' is in the given set.

Conversely let E' be in the given set. Since it is in U it is a frequent subgraph. Now construct $\tilde{E}$ from E' by adding edges as long as $\tilde{E}$ is a frequent subgraph. Now by construction $\tilde{E}$ is a maximal frequent subgraph. So $\tilde{E} \in U$. Since E' is in the given set, this is only possible if $\tilde{E} = E'$. Hence E' is a maximal frequent subgraph. □

4 An Algorithm for Finding Frequent Subgraphs Via Frequent Item Sets

This section describes the algorithms that are used in the sequel supplemented by some illustrative examples.

For our small example in Fig. 1 we can construct all maximal frequent subgraphs out of the two maximal frequent item sets $\{a, b, d\}$ and $\{d, e\}$ as follows:

– We decompose the frequent item sets into their components $\{a, b\}$, $\{d\}$ and $\{d, e\}$.
– By Theorem 1 we know that every maximal frequent subgraph coincides with one of these components, so in Theorem 2 we may choose U to be the set of these three components.
– We observe that $\{d\} \subseteq \{d, e\}$ and that the other pairs of components are incomparable.
– By Theorem 2 we conclude that there are exactly two maximal frequent subgraphs: $\{a, b\}$ and $\{d, e\}$.

Now we extend the idea of this example to a general applicable algorithm.

In order to compute the maximal frequent subgraphs from the set of components of maximal frequent item sets, we have to remove all components which are contained in another component. One way to do this is to first make the set of all components of maximal frequent item sets and then for every pair of components to check whether one is contained in the other, and if so, remove the smaller. Slightly more efficiently, the same can be done on-the-fly, as is presented in the algorithm *FindMFS* given in Fig. 2.

This algorithm has been designed in such a way that according to Theorems 1 and 2 at the end MFS consists exactly of the set maximal frequent subgraphs: all components of all maximal frequent item sets are computed, and during the execution of the algorithm, for any two of them for which the smaller is strictly contained in the larger, the smaller is removed. As ingredients of the algorithm we need:

```
$MFI$ := FindMFI($\{E_1, \ldots, E_n\}$);
$MFS := \emptyset$;
for all $E \in MFI$ do
    begin
    $C$ := FindComponents($E$);
    $MFS' := MFS$;
    for all $X \in C$ do
        begin
        $MFS := MFS \cup \{X\}$;
        for all $Y \in MFS'$ do
            begin
            if $X \neq Y \wedge X \subseteq Y$ then $MFS := MFS \setminus \{X\}$;
            if $X \neq Y \wedge Y \subseteq X$ then $MFS := MFS \setminus \{Y\}$;
            end
        end
    end
```

Fig. 2. Algorithm *FindMFS* for finding MFS via MFI

– The computation `FindMFI` of the set of all maximal frequent item sets. For this we use the the tool `Afopt` [13,2].
– The computation `FindComponents` of all components of a possibly non-connected graph. For this we use the standard algorithm (c.f., [14]) starting by a single edge and repeatedly adding all connected edges.

In our graph collection example the algorithm starts by finding the set of maximal frequent item sets to be $MFI = \{\{a, b, d\}, \{d, e\}\}$. Assume that $E = \{a, b, d\}$ is first chosen in the outer loop. Then the two components $\{a, b\}$ and $\{d\}$ are computed. Since MFS' is still empty, after the first execution of the body of the outer loop, the set MFS consists of these two components. In the second execution of the body of the outer loop the remaining set $\{d, e\}$ is chosen and decomposed into components. It turns out that this set is one component itself, and is added to MFS. Next in the inner loop it is checked whether this new component $\{d, e\}$ is either a subset or a superset of a component we found before. It is: $\{d\} \subseteq \{d, e\}$, by which $\{d\}$ is removed from MFS, yielding the final set $MFS = \{\{a, b\}, \{d, e\}\}$, indeed being the set of maximal frequent subgraphs.

5 Experimental Results and Discussion

In this section we give some experimental results with our approach and we compare them with the results obtained with the pathway miner tool `pwmine` described in [8]. We applied our algorithm to metabolic pathways from the KEGG database. Because of the comparison with above mentioned tool `pwmine`, we chose the same three metabolic pathways as in [8]: Glutamate, Pyrimidine and Alanine-Aspartate. In the sequel we discuss the results for each pathway separately.

We did two sets of experiments. In the first set we applied `pwmine` and our approach to the same data that was used in [8]. In the second set the tools were applied to new data from the KEGG database (as it was on January 1, 2008). The enzyme graphs were automatically extracted from the corresponding XML files. In KEGG each pathway has one general XML file. There is also one specific XML file per organism for that pathway in which the enzyme graph is given. Using the information in those files, we extracted those graphs and translated them into the corresponding input formats for the tools `pwmine` and `Afopt` using Java programs.

All experiments were performed on a PC with four 1.99 GHz Dual Core AMD Opteron 270 processors and 16GB memory, running under Linux. All time measurements were done using the `time` command. We obtain the total time used by adding the user time (u) and the system time (s). (We avoid using the total time as it includes the time consumed by other processes of the system.)

The columns in the tables indicate the following:

– T (%) gives the threshold value as percentages of organisms/graphs in the dataset.
– T (# org.) indicates the absolute threshold value in number of organisms/grafs for that dataset.
– #fr. edg. indicates the number of distinct frequent edges.
– #MFS gives the number of maximal frequent subgraphs.
– Max E indicates the number of edges in the largest maximal frequent subgraph.
– The "pwmine" column contains the runtime in seconds of the tool `pwmine`.
– The "FindMFS" column gives the total time (in seconds) we needed to calculate the maximal frequent item sets (MFI) *and* to extract all maximal frequent subgraphs from MFI. This was done by a Java implementation of the algorithm *FindMFS* given in Fig. 2 (Section 3). All programs can be downloaded from `http://www.findmfs.tk/`. (As it was previously mentioned, we used the tool `Afopt` to find the MFIs that we downloaded from [2].)

The time it took to produce the input files for `pwmine` and the files for `Afopt` is not taken into account. This is required only once per dataset and as such it causes only a negligible overhead.

5.1 Glutamate

Table 1 contains the experimental results for the Glutamate pathway obtained with the KEGG data used in [8]. At that time (2003) the KEGG database contained data for this pathway for 155 organisms. Therefore we consider a collection of the corresponding 155 enzyme graphs.

With this version of the KEGG database, for thresholds greater than 10% pathway mine is still faster than `FindMFS`. For thresholds of 5% and 7.5% `FindMFS` wins. Actually, for 5% the difference is already quite significant: `pwmine` cannot produce a result within 2000 seconds while `FindMFS` needs less than 2 seconds to find all maximal frequent subgraphs. For practical reasons this detail

Table 1. Experimental results for the Glutamate pathway with data from [8]

T (%)	T (# org.)	#fr.edg.	#MFS	Max E	pwmine	FindMFS
5.0	8	190	132	30	>2000	1.880
7.5	12	103	56	15	1.488	1.088
10.0	16	70	34	15	0.384	0.931
12.5	19	58	39	13	0.127	0.949
15.0	23	36	21	11	0.028	0.830
17.5	27	26	13	10	0.016	0.904
20.0	31	23	12	9	0.010	0.801
30.0	47	10	7	3	0.008	0.757
40.0	62	4	2	3	0.008	0.859
50.0	78	1	1	1	0.007	0.807
60.0	93	0	0	0	0.009	0.744
70.0	109	0	0	0	0.008	0.923
80.0	124	0	0	0	0.008	0.778

Table 2. Experimental results for the Glutamate pathway with new KEGG data

T (%)	T (# org.)	#fr.edg.	#MFS	Max E	pwmine	FindMFS
5.0	35	115	1260	76	>2000	11.602
7.5	52	113	959	70	>2000	6.361
10.0	70	107	658	62	>2000	5.158
12.5	87	100	498	49	>2000	3.716
15.0	105	90	376	42	>2000	3.197
17.5	122	87	275	38	>2000	2.196
20.0	140	82	222	37	>2000	2.249
30.0	209	57	135	29	>2000	1.794
40.0	279	42	36	25	>2000	1.084
50.0	349	28	29	18	10.88	1.132
60.0	419	18	14	11	0.124	1.022
70.0	489	13	13	3	0.028	0.89
80.0	558	0	0	0	0.027	0.661

may be only slightly beneficial, but it indicates what could happen when dealing with a larger amount of data, as we show in the sequel.

In Table 2 the analogous measurements for both tools are given using new data from the KEGG database (as of 01-01-2008). The Glutamate pathway in that version of the database contained 698 organisms.

With the new data `FindMFS` is a clear winner for thresholds below 60%. Already for a threshold of 40% `FindMFS` is more than 1,000 times faster than `pwmine`. Moreover, `pwmine` is not able to produce results within 2000 seconds for thresholds below 40%.

Using the old KEGG data, we were able to find the examples of maximal frequent subgraphs for Glutamate that were reported in [8] and that are given in Fig. 3. (Of course, this comes as no surprise, since we used the same data.) Using different thresholds one can obtain different versions of the graph in Fig. 3. By

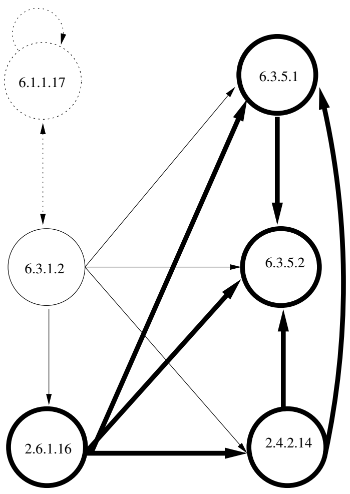


Fig. 3. Glutamate maximal frequent subgraphs (old data)

setting the threshold to 29%, which amounts to 45 of 155 organisms, one obtains the graph of 4 nodes and 6 edges which are shown in bold. All the enzymes in this subgraph are related through the use of the metabolite L-glutamine (see [8]) for more details).

By lowering the threshold to 19.3% (30 organisms) we get a graph of 5 nodes and 10 edges which is a supergraph of the above described graph. In Fig. 3 the nodes and the edges of the new graph are drawn with solid lines. The new graph is a supergraph of the old one and the new enzyme is also added because of a reaction involving L-glutamine. Finally, a threshold of 14.2% (22 organisms) gives another maximal graph that comprises 6 nodes and 13 edges and that contains the previous two. The new nodes and edges are denoted with dotted line. In this case too, the new enzyme is related to the L-glutamate.

The graphs in Fig. 3 are found with the new data as well by applying the same thresholds. Only this time they are embedded as subgraphs of larger maximal frequent subgraphs. For instance, taking the lowest threshold of 29% (202 organisms out of 698 in the new data) the corresponding graph from [8], i.e., the bold line graph in Fig. 3 obtained with the same threshold, can be found back as a subgraph of 9 larger graphs that have sizes of 12, 16, 18, 21, 22, 23, and 29 edges. The smallest of those graphs (12 edges) is given in Fig. 4.

The corresponding subgraph in Fig. 3 is drawn in bold in Fig. 4 too. The new enzymes EC 1.4.1.13, EC 1.5.1.12, and EC 2.6.1.1 are again related to the nodes of the old graph via L-glutamate.

Alternatively, the graph obtained with threshold 29% with the old data is found as a maximal frequent subgraph also with the new data. Only the threshold should be raised to 58% (405 out of 698 organisms).

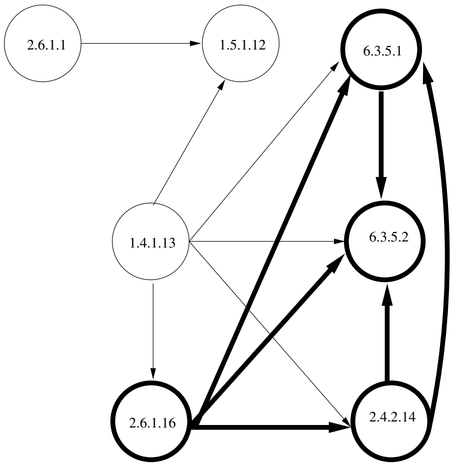


Fig. 4. Glutamate maximal frequent subgraphs (new data)

5.2 Pyrimidine Pathway

Tables 3 and 4 contain the results with the old (156 organisms) and new KEGG data (697 organisms), respectively, for the Pyrimidine pathway. Similarly as for the Glutamate pathway above, on the new larger data set, `FindMFS` outperforms `pwmine` for thresholds under 60%. The latter is not able to produce a result within a reasonable time for thresholds below 30%. Also for this pathway we were able to recover the maximal subgraphs reported in [8]. Using the same thresholds they are part of larger maximal subgraphs.

Table 3. Experimental results for the Pyrimidine pathway with data from [8]

T (%)	T (# org.)	#fr.edg.	#MFS	Max E	pwmine	FindMFS
5.0	8	172	191	37	>2000	2.827
7.5	12	126	185	19	10.333	2.597
10.0	16	91	120	15	0.306	1.690
12.5	20	71	67	15	0.090	1.001
15.0	23	57	49	12	0.026	1.048
17.5	27	42	35	9	0.013	0.832
20.0	31	36	23	7	0.008	0.674
30.0	47	13	8	4	0.007	0.734
40.0	62	5	4	2	0.007	0.628
50.0	78	0	0	0	0.007	0.553
60.0	94	0	0	0	0.007	0.531
70.0	110	0	0	0	0.006	0.678
80.0	125	0	0	0	0.007	0.531

Table 4. Experimental results for the Pyrimidine pathway with the new KEGG data

T (%)	T (# org.)	#fr.edg.	#MFS	Max E	pwmine	FindMFS
5.0	35	135	2684	70	>2000	49.151
7.5	52	108	3126	57	>2000	61.632
10.0	70	96	3595	50	>2000	63.989
12.5	87	90	3263	48	>2000	43.869
15.0	105	88	2396	46	>2000	27.46
17.5	122	83	1800	42	>2000	17.241
20.0	139	75	1390	38	>2000	12.103
30.0	209	64	546	30	>2000	4.401
40.0	279	52	310	26	1202.748	2.769
50.0	349	38	148	17	12.224	2.151
60.0	418	32	43	12	0.348	1.468
70.0	488	21	13	8	0.039	0.998
80.0	558	9	6	3	0.032	0.86

Table 5. Experimental results for the Alanine-Aspartate pathway with data from [8]

T (%)	T (# org.)	#fr.edg.	#MFS	Max E	pwmine	FindMFS
5.0	8	178	100	31	>2000	1.331
7.5	12	114	72	18	11.464	1.107
10.0	16	61	34	16	2.494	0.818
12.5	20	52	30	15	0.715	0.923
15.0	23	39	21	12	0.115	0.791
17.5	27	31	15	12	0.061	0.816
20.0	31	21	13	11	0.024	0.821
30.0	47	11	3	8	0.008	0.763
40.0	62	8	3	4	0.007	0.606
50.0	78	1	1	1	0.006	0.742
60.0	94	0	0	0	0.007	0.595
70.0	110	0	0	0	0.006	0.689
80.0	125	0	0	0	0.012	0.730

5.3 Alanine-Aspartate Pathway

The results with the Alanine-Aspartate pathway with the old 156 organisms (Tab. 5) and new 698 organisms (Tab. 6) data show the same trend like in the previous two cases. For lower thresholds `FindMFS` is much faster than `pwmine`.

6 Conclusions

We presented a new approach for mining frequently occurring patterns in biological networks. In particular we showed how state-of-the-art algorithms for mining maximal frequent item sets can be employed for finding maximal frequent subgraphs. We verified our approach experimentally on collections of metabolic pathways from the KEGG database.

Table 6. Experimental results for the Alanine-Aspartate pathway with the new KEGG data

T (%)	T (# org.)	#fr.edg.	#MFS	Max E	pwmine	FindMFS
5.0	35	106	172	54	>2000	2.905
7.5	52	95	177	48	>2000	2.292
10.0	70	86	158	39	>2000	1.869
12.5	87	77	93	37	>2000	1.688
15.0	105	72	84	30	>2000	1.965
17.5	122	63	71	30	>2000	1.12
20.0	140	61	55	29	>2000	1.433
30.0	209	43	36	23	498.561	1.061
40.0	279	32	19	18	13.348	0.921
50.0	349	30	18	13	0.491	1.031
60.0	419	17	13	7	0.033	0.985
70.0	489	11	10	3	0.027	0.838
80.0	558	0	0	0	0.015	0.704

With our approach we obtain a real-time response for nearly all thresholds for the current state of the KEGG database. For practical applications, analogous to BLAST and CLUSTAL queries, such a fast feedback to the user is of utmost importance. In contrast, the present MFS algorithms are not able to produce results within a reasonable time for thresholds below 30%. Moreover, for thresholds for which they do give results, our approach is up to 1,000 times faster.

As our approach is applicable to any type of biological networks, it would be interesting to apply it on other case studies. We intend to investigate the efficiency of our approach in case of a smaller number (less than 30) of large graphs (more than 10,000 edges). This is complementary to the application described in this paper in which we deal with a large collection of relatively small graphs.

Back to the KEGG metabolic pathways application, instead of applying our approach indiscriminately to the whole database, one can narrow the graph collection to a given class of organisms. It would be interesting to see if there is a correlation between the maximal graphs that are found and the evolutionary distance between different organisms.

In this paper we focused mainly on the algorithmic side of mining frequent patterns in biological networks. The interpretation of the obtained results from a biological point of view is an important avenue of future research.

Acknowledgments. D. Bošnački worked on this paper in the context of the BioRange SP.1.2.4 project funded by NWO (Dutch Organization for Scientific Research). We would like to thank Mehmet Koyutürk for providing the data used in [8], Huub ten Eikelder, Peter Hilbers, and Perry Moerland for the valuable discussions about [8], and Erik de Vink for his comments on a draft version of this paper.

References

1. Cook, D.J., Holder, L.B.: Graph Based Data Mining. IEEE Intell. Syst. 15, 32–41 (2000)
2. Frequent Itemset Mining Implementations Repository, http://fimi.cs.helsinki.fi/
3. Gouda, K., Zaki, M.J.: Efficiently Mining Maximal Frequent Itemsets. In: IEEE International Conference on Data Mining ICDM 2001, pp. 163–170 (2001)
4. Hu, J., Shen, X., Shao, Y., Bysstoff, C., Zaki, M.J.: Mining Protein Contact Maps. In: BIOKDD, pp. 3–10 (2002)
5. Huan, J., Wang, W., Prins, J., Yang, J.: SPIN: Mining Maximal Frequent Subgraphs from Graph Databases. In: KDD 2004, pp. 581–586 (2004)
6. Inokuchi, A., Wahio, T., Okada, T., Motoda, H.: Applying a priori-based graph mining method to mutagenesis data analysis. J. Comput. Aided Chem. 2, 87–92 (2001)
7. Karp, P.D., Mavrovouniotis, M.L.: Representing, Analyzing, and Synthesizing Biochemical Pathways. IEEE Expert, 11–21 (1994)
8. Koyutürk, M., Grama, A., Szpankowski, W.: An efficient algorithm for detecting frequent subgraphs in biological networks. Bioinformatics 20(suppl. 1), i200–i207 (2004)
9. Koyutürk, M., Kim, Y., Subramaniam, S., Szpankowski, W., Grama, A.: Detecting Conserved Interaction Patterns in Biological Networks. Journal of Computational Biology 13(7), 1299–1322 (2006)
10. Krishnamurthy, L., Nadeau, J., Özosyoğlu, G., Özosyoğlu, M., Schaeffer, G., Taşan, M., Xu, W.: em Pathways Database System: An Integrated System for Biological Pathways. Bioinformatics 19, 930–937 (2003)
11. Kuramochi, M., Karypis, G.: Frequent subgraph discovery. In: IEEE International Conference on Data Mining ICDM 2001, pp. 313–320 (2001)
12. http://www.kegg.com
13. Liu, G., Lu, H., Yu, J.X., Wang, W., Xiao, X.: AFOPT: An Efficient Implementation of Pattern Growth Approach. In: Proc. of the ICDM 2003 Workshop on Frequent Itemset Mining Implementations, FIMI 2003, Melbourne, Florida, USA, December 19 (2003)
14. Rivest, R.L., Leiserson, C.E.: Introduction to Algorithms. McGraw-Hill, New York (1990)
15. Thomas, L.T., Valluri, S.R., Karalaplem, K.: MARGIN: Maximal Frequent Subgraph Mining. In: Proc. of the 6th International Conference on Data Mining ICDM 2006, IEEE, Los Alamitos (2006)
16. Yan, X., Han, J.: gSpan: graph-based substructure pattern mining. In: IEEE International Conference on Data Mining ICDM 2002, pp. 721–724 (2002)

Multi-functional Protein Clustering in PPI Networks

Clara Pizzuti[1] and Simona E. Rombo[2]

[1] ICAR-CNR, Via P. Bucci 41C, 87036 Rende (CS), Italy
pizzuti@icar.cnr.it
[2] DEIS - Università della Calabria, Via P. Bucci 41C, 87036 Rende (CS), Italy
simona.rombo@deis.unical.it

Abstract. Protein-Protein Interaction (PPI) networks contain valuable information for the isolation of groups of proteins that participate in the same biological function. Many proteins play different roles in the cell by taking part in several processes, but isolating the different processes in which a protein is involved is often a difficult task. In this paper we present a method based on a greedy local search technique to detect functional modules in PPI graphs. The approach is conceived as a generalization of the algorithm PINCoC to generate overlapping clusters of the interaction graph in input. Due to this peculiarity, multi-facets proteins are allowed to belong to different groups corresponding to different biological processes. A comparison of the results obtained by our method with those of other well known clustering algorithms shows the capability of our approach to detect different and meaningful functional modules.

1 Introduction

Proteins are the building blocks of all organisms and play a fundamental role in executing and regulating most biological processes. Recently, it has been noted that, to fully understand cell activity, proteins cannot be analyzed independently from the other proteins because they seldom act in isolation to perform their tasks [25]. Advances in technology have allowed researches to derive, through experimental and in-silico methods, the collection of all interactions between proteins of an organism. The availability of protein-protein interaction (PPI) networks has thus stimulated the search for automated and accurate tools to analyze pair-wise protein interactions with the aim of extracting relevant functional modules. A functional module is a group of proteins participating to the same biological function. Their detection provides important knowledge to better understand the behavior of organisms.

PPI networks are naturally modelled as graphs where nodes represent proteins and edges represent pairwise interactions. Dense regions of a given PPI network correspond to highly interacting proteins that could be involved in common biological processes. One of the main difficulty in analyzing PPI graphs is their scale-free topology. A scale-free graph is characterized by the property that the degrees k of vertices are distributed according to a power law function, as $P(k) \propto k^{-\alpha}$, where $\alpha > 0$. This implies that most proteins interact with only a few other proteins, while a small number of proteins, known as $hubs$, have many interactions. Hubs proteins have been investigated [13] and recognized to have an important role for the life of organisms. Typically, because of their characteristic of being connected to a high number of proteins, they participate in multiple biological processes. Traditional clustering methods, however, assign a protein to only one group, which is unlikely for biological systems. In such a way these methods hamper the possibility of proteins to be clustered in several groups, on the basis of the different functions they have in the cell. This represents a significant inability of these approaches to describe the complexity of biological systems. To overcome such a problem, recent proposals have suggested different strategies [19,24,3].

In this paper, we present a partitioning technique of protein-protein interaction networks to produce overlapping clustering of the interaction graph. The algorithm, named Multi-Functional *PINCoC* (MF-*PINCoC*), is an extension of the method *PINCoC*, a *PPI network Co-Clustering* based algorithm, presented in [20], suitably modified to allow the participation of proteins to multiple functional groups. Co-clustering methods [16], differently from clustering approaches, aim at simultaneously grouping both the dimensions of a data set.

The PPI network is represented through the binary adjacency matrix A of the associated graph, where rows and columns correspond to proteins and a 1 entry at the position (i,j) means that the proteins i and j interact. The algorithm searches for, eventually overlapping, dense sub-matrices containing the maximum number of ones by using a greedy local search technique. It starts with an initial random solution constituted by a single protein and finds a locally optimal solution by adding/removing connected proteins that best contribute to improve a *quality* function. In order to enable participation of a protein to more groups, its degree k, i.e. the number of other proteins with which it is connected, is computed. A protein can be added to the current cluster if the number of clusters to which it has already been assigned is less than its degree. The method is enriched with one step of backtracking, to limit the effects of the initial random choice of a protein to build a cluster, and a remove strategy of proteins, to escape poor local optima. When the algorithm cannot improve any more the solution found so far, the computed cluster is returned. At this point a new random protein is chosen, and the process is repeated until all the proteins are assigned to a group.

MF-*PINCoC* has two fundamental advantages with respect to other approaches presented in the literature. The first, inherited from *PINCoC*, is that the number of clusters is automatically determined by the algorithm. The second, which is its main characteristic, is that for each protein interacting with other proteins, MF-*PINCoC* is able to identify the different groups in which the protein is involved, each group being distinguished by a different biological property. Note that, differently from other techniques [24], MF-*PINCoC* allows the participation to different clusters not only to the highly connected proteins recognized as hubs[1], but also to all the other proteins. Such a peculiarity is automatically incorporated in the approach without any lack in efficiency, and it avoids leaving possible candidates to be multi-facets proteins out from the analysis.

In the experimental result section we show that MF-*PINCoC* is able *(i)* to efficiently isolate groups of proteins corresponding to the most compact sets of interactions, and *(ii)* to assign proteins to more than one cluster, each characterized by a different biological function. A comparison with other well known protein clustering methods points out the very good results of our approach with respect to them.

[1] In [24] the authors recognized as *hubs* those proteins involved in a number of interactions between 40 and 283.

The paper is organized as follows. The next section describes the MF-*PINCoC* algorithm and the variations introduced w.r.t. *PINCoC* to allow overlapping clusterings. Section 3 reports the related work on protein clustering. Section 4 illustrates the experiments carried out on the Saccaromyces cerevisiae protein data set and compares the obtained results with those of [4,14,24]. Finally, in Section 5 we draw our conclusions.

2 Approach Description

In this section we recall the notation adopted by both the MF-*PINCoC* and *PINCoC* algorithms, and describe the extensions realized to allow for multiple group participation of proteins.

A PPI network $\mathcal{P}$ is modelled as an undirected graph $G = (V, E)$ where the nodes V correspond to the proteins and the edges E correspond to the pairwise interactions. If the network is constituted by N proteins, the associated graph can be represented with its $N \times N$ adjacency matrix A, where the entry at position (i, j) is 1 if there is an edge between nodes i and j, 0 otherwise. The problem of finding dense regions of a PPI network $\mathcal{P}$ can be transformed in that of finding dense subgraphs of the graph G associated with $\mathcal{P}$, and consequently, dense sub-matrices of the adjacency matrix A corresponding to G. Searching for dense sub-matrices of such a matrix A can be viewed as a special case of co-clustering a binary data matrix where the set of rows and the set of columns represent the same concept. In order to better explain the idea, first a definition of co-clustering is given, and then the formalization of the problem of clustering proteins as a co-clustering problem is provided. Co-clustering [16,7], also known as bi-clustering, differently from clustering, tries to simultaneously group both the dimensions of a data set. A *co-cluster* of a matrix A is defined as a sub-matrix $B = (I, J)$ of A, where I is a subset of the rows $X = \{I_1, \ldots, I_N\}$ of A, and J is a subset of the columns $Y = \{J_1, \ldots, J_M\}$ of A.

Then, the problem of co-clustering may be formulated as follows: given a data matrix A, find row and column maximal groups which divide the matrix into regions that satisfy some homogeneity characteristics. The kind of homogeneity a co-cluster has to fulfil depends on the application domain. In our case we would like to find as many proteins as possible having the highest number of interactions. This corresponds to identify highly dense squared sub-matrices, i.e., containing as many values equal to 1 as possible. Higher the number of ones, more likely those proteins are to be functionally related.

Let a_{iJ} denote the *mean value* of the ith row of the co-cluster $B = (I, J)$, and a_{Ij} the mean of the jth column of B. More formally,

$$a_{iJ} = \frac{1}{|J|}\sum_{j\in J} a_{ij}, \text{ and } a_{Ij} = \frac{1}{|I|}\sum_{i\in I} a_{ij}$$

The *volume* v_B of a co-cluster $B = (I, J)$ is the number of 1 entries a_{ij} such that $i \in I$ and $j \in J$, that is $v_B = \sum_{i\in I, j\in J} a_{ij}$.

Given a co-cluster $B = (I, J)$, the *power mean of B of order r*, denoted by $\mathbf{M}_r(B)$ is defined as

$$\mathbf{M}_r(B) = \frac{\sum_{i\in I}(a_{iJ})^r + \sum_{j\in J}(a_{Ij})^r}{|I| + |J|}$$

A measure based on volume and row/column mean, that allows the detection of maximal and dense sub-matrices, can be defined as follows.

Given a co-cluster $B = (I, J)$, let $\mathbf{M}_r(B)$ be the power mean of B of order r. The *quality* of B is defined as

$$Q(B) = \mathbf{M}_r(B) \times v_B$$

The problem of protein clustering can be formulated as follows: given a data matrix A, find row and column maximal groups that partition the matrix into sub-matrices $\{B_1, \ldots B_h\}$, each having maximal $Q(B_i)$ values.

It is worth to note that high values of the exponent r bias the *quality* function towards matrices containing a low number of zeroes. In fact, it amplifies the weight of the densely interconnected nodes, while reducing those of less connected in the computation of the *quality* function. In the following the terms co-cluster, cluster, and sub-matrix are used to express the same concept.

MF-$PINCoC$ starts with an initial random cluster $B_i = (I_i, J_i)$ constituted by a single row and a single column such that $I_i = \{l\}$ and $J_i = \{l\}$, where $1 \leq l \leq N$ is a random row/column index. Then it evolves the initial cluster by successive transformations of B_i, until the *quality* function is improved. The transformations consist in the change of membership (called $flip$ or $move$) of the row/column that leads to the largest increase of the *quality* function. If a bit is set from 0 to 1 it means that the corresponding protein, which was not included in the cluster B_i, is added to B_i. Vice versa, if a bit is set from 1 to 0 it means that the corresponding protein is removed from the cluster. During its execution, in order to avoid getting trapped into poor local maxima, instead of performing the flip maximizing the $quality$, with a user-provided probability p the algorithm selects the row/column of B_i scoring the minimum decrease of the *quality* function, and removes it from B_i. This kind of flip is called REMOVE-MIN. The flips are repeated until either a preset of maximum number of flips is reached, or the solution cannot ulteriorly be improved (get trapped into a local maximum). Until the number of flips is below a fixed maximum value and the quality function increases, MF-*PINCoC* executes a REMOVE-MIN move with probability p, and a greedy move with probability $(1-p)$; otherwise, the cluster $B_i = (I_i, J_i)$ is returned. At this point the algorithm performs one step of backtracking, i.e., for each $h \in I_i$ it temporary removes h from I_i and tries to find a node l such that $I_i - \{h\} \cup \{l\}$ improves the *quality* of B_i. In such a case h is removed and l is added. If more than one l node exists, the one generating the better improvement of $Q(B_i)$ is chosen. Finally, B_i is added to B, its rows/columns are removed from A, a new random cluster is generated, and the process is repeated until all the rows/columns have been assigned.

As previously pointed out, many proteins may be involved in several biological functions by interacting with different groups of proteins. In order to allow these multi-facets proteins to be assigned to more than one cluster, we relax the constraint adopted in *PINCoC* to exclude a protein to be considered for inclusion in another cluster, once it has already been put into a group. To this end, for each protein we compute its degree k, i.e. the number of other proteins with which it is connected. When building a new cluster, a protein can be added to the current cluster if the number of clusters to which it has already been assigned is less than its degree. In such a way each protein, not only hubs, can belong to multiple clusters, provided that its contribution to the *quality* function is effective, i.e. it is the choice that produces the best improvement. In the next section we report the main proposals to protein clustering recently presented in the literature.

3 Related Work

Clustering approaches to PPI networks can be broadly categorized as distance-based and graph-based [15] ones. Distance-based clustering approaches apply traditional clustering techniques by employing the concept of distance between two proteins [2,18]. Graph-based clustering approaches consider the network topology and partition the graph trying to optimize a cost function [12,5,10,4,23,22,14,19,24]. In the following some of the main proposals are described.

Molecular complex detection (MCODE) [4] detects dense and connected regions by weighting nodes on the basis of their local neighborhood density. To this end, the k-core concept is applied. A k-core is a graph in which each vertex has degree at least k. The highest k-core of a graph is the most densely connected subgraph. The core-clustering coefficient of a node, i.e. the density of the highest k-core of the vertices directly connected to it, is then used to give a weight to each vertex. MCODE performs three steps: vertex weighting, complex prediction, and optional postprocessing to add or remove proteins. In the first step nodes are weighted according to the density of the highest k-core. In the second step the vertex with the highest weight is selected as seed cluster, and new nodes are included in the cluster if their weight is above a fixed threshold. This process is repeated for the next-highest unexamined node. In such a way the densest regions of the graph are identified. Postprocessing is finally optionally executed to filter proteins according to certain connectivity criteria.

The Restricted Neighborhood Search Clustering (RNSC), proposed by King et al. [14], is a cost-based local search algorithm that explores the solution space of all the possible clusterings to minimize a cost function that reflects the number of inter-cluster and intra-cluster edges. The idea resembles our approach, however, RNSC uses two cost functions. The first, called the naive cost function, for each node v, computes the number of bad connections incident with v, i.e. one that exists between v and a node not belonging to the same cluster of v, or one that does not exist between v and another node in the same cluster as v. The second one, called the scaled cost function, measures the size of the area that v effects in the clustering. The algorithm begins with a random clustering, and attempts to find a clustering with low naive cost by moving a vertex from a cluster to another one. Then it tries to improve the solution by searching for a clustering with low scaled cost. Differently from the approach presented here, neither MCODE nor RNSC allow the participation of a protein to multiple clusters.

Pereira et al. [19] transform the interaction graph into the corresponding line graph, in which edges represent nodes and nodes represent edges. Then they apply the graph clustering algorithm TribeMCL of [10] to group the interaction network corresponding to the line graph, and transform back the obtained clusters. The approach of clustering the line graph produces an overlapping graph partitioning of the original protein-protein interaction graph, thus allowing proteins to be present in multiple functional modules.

In [1] CFinder, a program for detecting and visualizing densely interconnected and overlapped groups of nodes, is presented. CFinder uses the Clique Percolation Method

[9] to find k-clique percolation clusters, i.e. groups of nodes that can be reached via chains of k-cliques and the link in these cliques. The parameter k has to be provided in input. Approaches such as [1] may be viewed as general approaches to study the structure of networks, suitably represented as graphs (e.g., genetic or social networks and microarray data), rather than a specialized technique to cluster PPI networks.

In [8] Cho et al. propose a flow-based modularization approach to identify overlapping functional modules in a PPI network. The modularization process consists of three phases: informative protein selection, flow simulation to detect preliminary modules and a post-process to merge similar preliminary modules. Differently from such an approach, MF-*PINCoC* does not need any post-processing step to produce the final overlapping clusterings.

Ucar et al. [24] propose an approach to reduce the scale-free topology of PPI networks by duplicating the hub nodes. After this refinement, the resulting graph is clustered by using three known graph partitioning methods. Because of the duplication process, hub proteins can be placed in multiple groups. Of course this multiple participation, differently by our approach, is not possible for the other proteins.

A different method, based on an ensemble framework, is described in [3]. The authors use three traditional graph partitioning algorithms with two metrics to obtain six basis clusterings. Then apply different consensus methods to decide each protein to which cluster should belong. A soft consensus clustering variant has also been developed to allow proteins having high propensity towards multiple memberships, to be assigned to different clusters. Though amenability to multiple membership is computed for all the nodes, the authors note that hub proteins have the highest probability to participate in more than one cluster. Both these last two methods need as input parameter the number of clusters to find. Our approach, on the contrary, searches for all the possible clusters it can find in the network.

In the next section we report the results obtained by our approach and compare them with those obtained by MCODE and RNSC, two of the most known methods in the literature [6]. Such a comparison further confirms the importance of allowing for multiple-cluster participation; in fact, constraining each protein to belong to only one module causes clusterings that are often less significant. Moreover, a discussion regarding the participation of proteins to multiple clusters, with respect to the hub proteins identified in [24], will be reported in the last sub-section.

4 Experimental Validation

In this section we present the results obtained by running MF-*PINCoC* on the PPI network of budding yeast *Saccaromyces cerevisiae*. The data set has been extracted from the *DIP* database [21] (*http://dip.doe-mbi.ucla.edu/*). At the time of download (May 2007) it consisted of 5,027 proteins and 22,223 interactions.

4.1 Validation Metrics

Before presenting the experiments, we describe the validation metrics used to asses the quality of the results. We used two metrics, a topological measure (*clustering coefficient*) and a domain based measure (*p-value*).

Clustering Coefficient: the concept of clustering coefficient has been defined by Watt in [26] and takes into account only the nodes of a network and how they are linked together. Given a node i, let n_i be the number of links connecting the k_i neighbors of i to each other. The clustering coefficient of i is defined as $C_i = 2n_i/k_i(k_i-1)$. Note that n_i represents the number of triangles passing through i, and $k_i(k_i-1)/2$ the number of possible triangles that could pass through node i. The clustering coefficient C_{B_j} of a cluster B_j is the average of the clustering coefficients of the proteins belonging to B. Analogously, the clustering coefficient C_B of a clustering $B = \{B_1, \ldots, B_h\}$ is $C_B = \sum C_{B_j}/h$.

p-value: in the PPI networks it is important to verify if the clusters obtained correspond to a function meaningful from a biological point of view. This validation can be done by using the known biological associations from the *Gene Ontology Consortium Online DataBase* [11] . The Gene Ontology database provides three vocabularies of known associations: Molecular Function, Cellular Component, and Biological Process. We used the process vocabulary for validation by querying the GO Term-Finder tool (*http://db.yeastgenome.org/cgi-bin/GO/goTermFinder*) and the p-values returned to obtain a statistical and biological meaningfulness of a group of proteins. The p-value is a commonly used measure of the functional homogeneity of a cluster. It gives the probability that a given set of proteins occurs by chance. In particular, given a cluster of size n with m proteins sharing a particular biological annotation, then the probability of observing m or more proteins that are annotated with the same GO term out of those n proteins, according to the Hypergeometric Distribution, is:

$$p - value = \sum_{i=m}^{n} \frac{\binom{M}{i}\binom{N-M}{n-i}}{\binom{N}{n}}$$

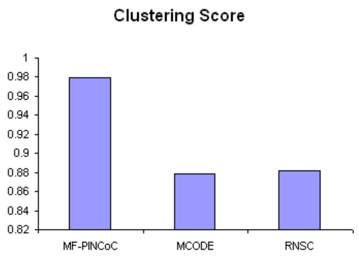
where N is the number of proteins in the database with M of them known to have that same annotation. Thus, the closer the p-value to zero, the more significant the associated GO term. The biological significance of a group is settled by using a cut-off value to distinguish significant from insignificant groups. If a cluster has a p-value below the cut-off, it is considered insignificant. In our experiments we used a cut-off of 0.05. As observed in [24], it is interesting to have a global measure of an obtained clustering, instead of the p-value of a single group. The p-value score of a clustering is then defined as

$$clustering\ score = 1 - \frac{\sum_i^{n_S} min(p_i) + (n_I \times cutoff)}{(n_I + n_S) \times cutoff}$$

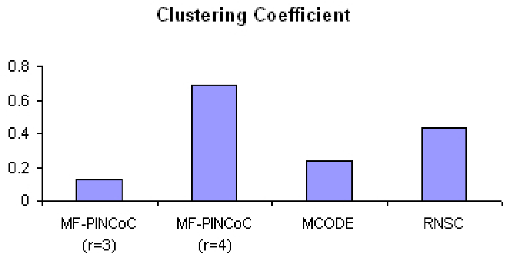
where $min(p_i)$ is the smallest p-value of the partition i, n_S is the number of significant partitions, and n_I is the number of insignificant partitions.

4.2 Comparison of MF-PINCoC, MCODE, and RNSC

In this section we compare the results obtained by running MF-PINCoC, MCODE, and RNSC on the S. Cerevisiae network. In particular, such a comparison has been carried out not only to investigate the ability of our method to discover significant functional modules w.r.t. other well consolidated techniques, but also to analyze how allowing



(a)



(b)

Fig. 1. Comparison among the three methods, showing:(a) Clustering Score; (b) Clustering Coefficient

proteins to participate in different clusterings may be useful to obtain more significant groups.

MF-*PINCoC* needs as input parameters the probability p of a REMOVE-MIN move, the number of maximum moves allowed, and the order r of the *quality* function. We set the former to 0.1, the second to 1,000, and the latter to 3. It is worth to note that *(i)* a low value of probability p is preferable to avoid the disruption of the greedy steps; *(ii)* the number of maximum flips has never been reached, in fact on average not more than 50 flips were executed before reaching a local optimum; *(iii)* the order value used is a compromise between the compactness of clusters and their size. As regards MCODE and RNSC, we run the two methods with the default parameters set by the authors. MF-*PINCoC* returned 6,108 clusters, 5,189 were couples of proteins, 145 cliques constituted by triples, 588 of size between 4 and 6, the remaining 186 with a number of proteins between 7 and 40. MCODE obtained only 57 clusters, 17 of which were triples. The cluster size is between 3 and 59. The clusters covered only 789 proteins out of the 5,027 present. RNSC obtained 2,524 clusters, 1,017 were singletons, 972 couples of proteins, 375 triples, 134 clusters of size between 4 and 7, and the remaining 26 of size between 8 and 21. Because of the different number of clusters obtained, we chose 50 random clusters returned by each method with maximum size and queried the GO Term-Finder tool. MF-*PINCoC* gave back one insignificant cluster, while MCODE and RNSC gave 6 and 5 insignificant groups, respectively.

Figure 1 graphically illustrates the behavior of MF-PINCoC, MCODE and RNSC in terms of both domain-based and topological measures. In particular, Figure 1 (a) shows the clustering scores, computed on the 50 chosen clusters, for the three methods. The figure points out that the clustering score of MF-PINCoC (0.980) is greater than those of the other two methods, which is 0.879 for MCODE and 0.882 for RNSC respectively. This means that the biological meaning of the clusters obtained by MF-PINCoC is, on average, better than the clusters generated by the other two methods. Figure 1 (b) shows the clustering coefficients computed on all the obtained clusters. The clustering coefficient of MF-PINCoC has been computed for two different values of the parameter r ($r = 3, 4$). As already observed in section 2, higher values of r bias our method towards denser but smaller clusters. In fact, for $r = 4$ we obtained 6,322 clusters, 5,332 were couples, 138 cliques constituted by triples, 724 of size between 4 and 6, the

Table 1. Some significant clusters obtained by the three methods MF-PINCoC, MCODE and RNSC

	Cluster	p-value	Associated process
MF-PINCoC	PFS2,PTI1,MPE1,REF2,YTH1,FIP1,CFT1, CFT2,PTA1,YSH1,HCA4,PAP1,RNA14,GLC7	2.17E-26	
MCODE	CFT1,CFT2,FIP1,GLC7,MPE1,PAP1,PFS2,PTA1,PTI1, MPE1,PAP1,PFS2,PTA1,PTI1,REF2,RNA14,YSH1,YTH1	4.67E-27	mRNA Polyadenylation
RNSC	REF2,PCF11,GLC7,RNA14,YTH1,FIP1,PAP1,CFT1, CFT2,PTA1,YSH1,PTI1,PFS2,MPE1,HCA4,SSU72	1.08E-28	
MF-PINCoC	NUP84,NUP60,CRM1,PAB1,MSN5,NUP57,NUP42,NUP49, GSP1,NUP145,SRP1,NUP2,NUP100,KAP123,KAP95,PSE1,NUP116	6.61E-26	Nuclear Transport
MCODE	MSN5, NTF2, NIC96, NUP145, NSP1, GSP1	1.66E-09	
MF-PINCoC	HAS1,MAK21,CIC1,SDA1,NOP6,NUG1,NOP7,CKA1,NOP2,SSF1, NOP4,BUD20,RPF2,YTM1,RLP7,NOP15,MAK5,NSA2,ERB1,TIF6,NOG1	1.68E-22	Ribosome Biogenesis and Assembly
RNSC	URB1, NOP4, MAK21, HAS1, NOC2, BRX1, CIC1, NOP12, PUF6, DBP10, NOP2, SSF1, RPF2, DRS1, MAK5	2.90E-15	
MF-PINCoC	NUP84,CRM1,NUP120,MSN5,NUP42,NUP145,NUP57, NUP49,SRP1,NUP2,NUP100,KAP95,PSE1,NUP116	2.03E-23	
MCODE	SPP381, MSL1, LEA1, SMX3	1.19E-06	Nuclear mRNA splicing, via spliceosome
RNSC	CDC6, ORC1, ORC2, ORC3, ORC4, ORC5, ORC6	6.16E-16	

remaining 128 of size between 7 and 33. Thus, with respect to the previous experiment, with $r = 3$, clusters have a lower number of proteins. However, the clustering coefficient is 0.69 with $r = 4$ and 0.13 with $r = 3$. On the other hand, MCODE scored a clustering coefficient 0.23, and RNSC 0.43. This points out that, in order to obtain a better value also in terms of topological connectivity, the input parameter r has to be properly tuned.

Table 1 shows some of the clusters obtained by the three methods, for which the GO validation returned the same associated process. The table points out the good capability of MF-*PINCoC* to isolate functional modules.

4.3 Multi-functional Proteins

We now show, with some examples, how MF-*PINCoC* is able to cluster multi-facets proteins into different functional modules, each characterized by a particular function. In table 2 we report the protein name, the number of proteins with which it is connected (denoted degree), the list of proteins participating to the same cluster, and the associated biological process. We consider three proteins KAP95, LSM8, and CKA1 that have been discussed by Ucar et al. in [24], and compare their results with ours. As reported in [24], KAP95 is an essential protein known to take part in *nucleocytoplasmatic transport*. MF-*PINCoC* groups KAP95 with other 5 proteins (NTF2, GSP1, PSE1, SRP1, NUP1) participating to this same biological process. Ucar et al. point out that one the partitions they found (NTF2, SSA1, YRB1, RNA1, GSP1, SRM1, MTR10, KAP122, KAP142, KAP124, NUP1, NUP2, NUP42, NUP60, NUP82, NUP145, NUP157, NUP-170) contained 8 NUPs proteins and 3 KAPs proteins, known as nucleoporins and karyorephins respectively, with p-value 1.07E-27. We obtained an analogous result, in two different clusters. The former contains 9 NUPs proteins (NUP2, NUP84, NUP60, NUP57, NUP42, NUP49, NUP145, NUP100, NUP116) and two KAPs proteins (KAP-95 and KAP123), sharing the *Nuclear Transport* biological process with p-value 6.61 E-26, the second one contains 4 NUPs proteins (NUP116, NUP57, NUP60, NUP100, NUP145) and 3 KAPs proteins (KAP95, KAP104, KAP123), sharing the *cellular localization* process, with p-value 2.06E-09.

The hub protein LSM8 has been found by Ucar et al. with other 10 proteins (LSM2, LSM3, LSM5, PRP3, PRP4, PRP6, PRP21, PRP31, SMB1, SPP381) with biological process *mRNA splicing* and p-value 1.2E-12. We found the same protein in several groups, in particular, as reported in the table 2, LSM8, for this same process, has been grouped with 12 proteins (LSM3, PRP3, PRP4, PRP6, PRP8, PRP31, SMB1, SPP381, SMD3, SMX2, SNU114, SNU66) having p-value 1.46E-23. The two sets of proteins are almost the same, the difference is that the cluster found by MF-*PINCoC* does not contain LSM2, but has four new proteins, SMD3, SMX2, SNU114, SNU66, and a p-value much higher, thus a better biological meaning. However, LSM8 has been grouped with other proteins forming other functional modules, like reported in the table. For

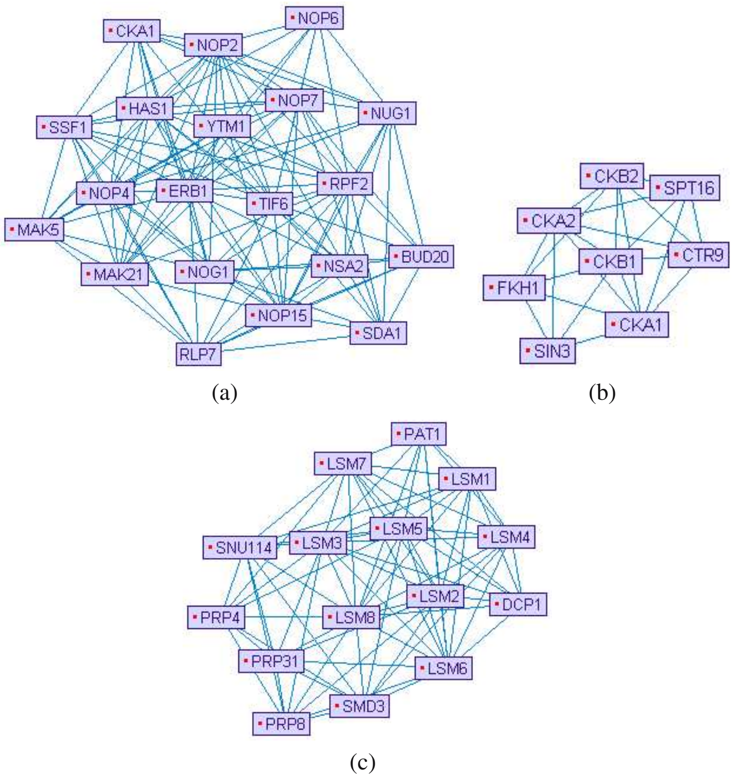


Fig. 2. PPI networks of clusters obtained showing:(a) first cluster reported in table 2 relative to the CKA1 protein; (b) third cluster reported in table 2 relative to the CKA1 protein; (c) second cluster of table 2 relative to the LSM8 protein

Table 2. Some examples of hub proteins and the clusters they participate

Hub-Protein	degree	Clusters	p-value	Associated process
KAP95	58	KAP95,KAP123,NUP2,NUP84,NUP60, NUP42,NUP49,NUP145,NUP100,NUP116, SRP1,CRM1,PAB1,PSE1,GSP1	6.61E-26	Nuclear Transport
		MSN5,NUP57,KAP95,KAP104,KAP123, NUP116, NUP57,NUP60,NUP100,GSP1,SRP1,PSE1	2.06E-09	Cellular localization
		KAP95, NTF2,GSP1,PSE1,SRP1,NUP1	1.71E-09	Nucleocytoplasmatic transport
LSM8	71	LSM3, LSM8, PRP3, PRP31, PRP4, PRP6, PRP8, SMB1, SMD3, SMX2, SNU114, SNU66, SPP381	1.46E-23	Nuclear mRNA splicing, via spliceosome
		LSM8, LSM1, LSM2, LSM3, LSM4, LSM5, LSM6, LSM7, DCP1,PAT1,PRP31, PRP4, PRP8, SMD3, SNU114	3.02E-22	mRNA metabolic process
		LSM1, LSM8,LSM2,EDC3,KEM1,DCP2, LSM4	1.44E-06	Biopolymer catabolic process
CKA1	66	HAS1,MAK21,CIC1,SDA1,NOP6,NUG1,NOP7,CKA1, NOP2,SSF1,NOP4,BUD20,RPF2,YTM1, RLP7,NOP15,MAK5,NSA2,ERB1,TIF6,NOG1	1.68E-22	Ribosome biogenesis and assembly
		RPF2,YTM1,NOG1,ERB1,MAK5,HAS1,TIF6,CKA1, MAK21,NOP2,NOP4,NOP6,NOP7,NOP15,CIC1,SSF1	9.96E-07	Cellular component organization and biogenesis
		CKA1,CKB1,CKA2,CKB2,SPT16,CTR9,SIN3,FKH1	1.03E-07	Transcription, DNA-dependent

example, it is clustered with 7 proteins of the LSM family, which are known to interact each other in the *mRNA metabolic* process, with a very low p-value (3.02E-22).

CKA1 is a protein involved in several cellular events. Ucar et al. located CKA1 in three different partitions. One is annotated with the biological process *transcription, DNA-dependent* and p-value 2.3e-19, the second one with *protein amino acid phosphorylation* and p-value 1.2E-05, the third group is annotated with *organelle organization and biogenesis* and p-value 3.2E-12. MF-*PINCoC* found, among the others, a group with p-value 1.68E-22 and annotation *ribosome biogenesis and assembly*, another one with p-value 9.96E-07 and process *cellular component organization and biogenesis*, the third one with p-value 1.03E-07 and biological process *transcription, DNA-dependent*. Finally, figure 2 draws three clusters of proteins in which CKA1 and LSM8 are involved. In particular, figures 2(a) and 2(b) show the first and third clusters reported in table 2 relative to the CKA1 protein. Figure 2(c) displays the second cluster of table 2 relative to the LSM8 protein. The graphs have been drawn by using the PIVOT software [17]. These results point out that the strategy of allowing proteins to belong to different clusters seems to be effective in grouping multi-functional proteins into multiple functional groups, to individuate biologically significant modules, each corresponding to a different function in which these proteins are involved.

5 Conclusions

We proposed the algorithm *MF-PINCoC*, an extension of the algorithm *PINCoC*, aiming at individuating clusters of multi-facets proteins in PPI networks. One of the main feature of the method consists in allowing proteins to be placed in multiple clusters. This is a distinguished advantage since it enables a more accurate representation of the complexity of biological systems and the detection of different functional modules in which proteins are involved. As proved by tests carried out on the *Saccaromyces cerevisiae* proteins data set, the presented method returns partitions that are biologically relevant, correctly clustering proteins which are known to participate in different biological processes. A comparison with other existing approaches shows that MF-*PINCoC* is competitive with respect to these methods according to validation techniques commonly adopted in the literature.

References

1. Adamcsek, B., Palla, G., Farkas, I.J., Dernyi, I., Vicsek, T.: Cfinder: locating cliques and overlapping modules in biological networks. Bioinformatics 22(8), 1021–1023 (2006)
2. Arnau, V., Mars, S., Marìn, I.: Iterative cluster analysis of protein interaction data. Bioinformatics 21(3), 364–378 (2004)
3. Asur, S., Ucar, D., Parthasarathy, S.: An ensemble framework for clustering protein-protein interaction networks. Bioinformatics 40, i29–i40 (2007)
4. Bader, G., Hogue, H.: An automated method for finding molecular complexes in large protein-protein interaction networks. BMC Bioinformatics 4(2) (2003)
5. Blatt, M., Wiseman, S., Domany, E.: Superparamagnetic clustering of data. Phisical Review Letters 76(18), 3251–3254 (1996)
6. Brohèe, S., van Helden, J.: Evaluation of clustering algorithms for protein-protein interaction networks. BMC Bioinformatics 7, 488 (2006)
7. Cheng, Y., Church, G.M.: Biclustering of expression data. In: Proceedings of the 8th International Conference On Intelligent Systems for Molecular Biology (ISMB 2000), pp. 93–103 (2000)
8. Cho, Y.-R., Hwang, W., Ramanathan, M., Zhang, A.: Semantic integration to identify overlapping functional modules in protein interaction networks. BMC Bioinformatics 8, 265 (2007)
9. Derenyi, I., et al.: Clique percolation in random networks. Physical Review Letters 94, 160–202 (2005)
10. Enright, A.J., Dongen, S.V., Ouzounis, C.A.: An efficient algorithm for large-scale detection of protein families. Nucleid Acids Res 30(7), 1575–1584 (2002)
11. Asburner, S., et al.: Gene ontology: tool for the unification of biology. the gene ontology consortium 25, 25–29 (2000)
12. Hartuv, E., Shamir, R.: Clustering algorithm based graph connectivity. Information Processing Letters 76, 175–181 (2000)
13. Jeong, H., Barabasi, A.L., Oltvai, Z.N.: Lethality and centrality in protein networks. Nature 411, 41–42 (2001)
14. King, A.D., Przulj, N., Jurisica, I.: Protein complex prediction via cost-based clustering. Bioinformatics 20(17), 3013–3020 (2004)
15. Lin, C., Cho, Y., Hwang, W., Pei, P., Zhang, A.: Clustering methods in protein-protein interaction network. In: Knowledge Discovery in Bioinformatics: Techniques, Methods and Application, John Wiley & Sons,Inc., Chichester (2006)
16. Madeira, S.C., Oliveira, A.L.: Biclustering algorithms for biological data analysis: A survey. IEEE Transactions on Computational Biology and Bioinformatics 1(1), 24–45 (2004)
17. Orlev, N., Shamir, R., Shiloh, Y.: Pivot: Protein interaction visualization tool. Bioinformatics 20(3), 424–425 (2004)
18. Pei, P., Zhang, A.: A two-step approach for clustering proteins based on protein interaction profiles. In: IEEE Int. Symposium on Bioinformatics and Bioengeneering (BIBE 2005), pp. 201–209 (2005)

19. Pereira, J.B., Enright, A.J., Ouzounis, C.A.: Detection of functional modules from protein interaction networks. Proteins: Structure, Fuctions, and Bioinformatics (20), 49–57 (2004)
20. Pizzuti, C., Rombo, S.: Pincoc: a co-clustering based approach to analyze protein-protein interaction networks. In: Proceedings of the 8th International Conference on Intelligent Data Engineering and Automated Learning (IDEAL 2007) (2007)
21. Salwinski, L., Miller, C.S., Smith, A.J., Pettit, F.K., Bowie, J.U., Eisenberg, D.: The database of interacting proteins: 2004 update. Nucleic Acids Research 32(Database issue), D449–D451 (2004)
22. Samantha, M.P., Liang, S.: Redundancies in large-scale protein interaction networks. In: Proceedings of the National Academy of Science, USA, 100, pp. 12579–12583 (2003)
23. Spirin, V., Mirny, L.A.: Protein complexes and and functional modules in molecular networks. In: Proceedings of the National Academy of Science, USA, 100, pp. 12123–12128 (2003)
24. Ucar, D., Asur, S., Çatalyürek, Ü.V., Parthasarathy, S.: Improving functional modularity in protein-protein interactions graphs using hub-induced subgraphs. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) PKDD 2006. LNCS (LNAI), vol. 4213, pp. 371–382. Springer, Heidelberg (2006)
25. von Mering, D., Krause, C., et al.: Comparative assessment of a large-scale data sets of protein-protein interactions. Nature 31, 399–403 (2002)
26. Watt, D.J.: Small worlds. Princeton University Press, Princeton (1999)

Protein-Protein Interaction Network Querying by a "Focus and Zoom" Approach

Valeria Fionda[1], Luigi Palopoli[2], Simona Panni[3], and Simona E. Rombo[2]

[1] Dept. of Mathematics, Università della Calabria
[2] DEIS, Università della Calabria
[3] Dept. of Cellular Biology, Università della Calabria

Abstract. We propose an approach to network querying in protein-protein interaction networks based on bipartite graph weighted matching. An algorithm is presented that first "focuses" the potentially relevant portion of the target graph by performing a global alignment of this one with the query graph, and then "zooms" on the actual matching nodes by considering their topological arrangement, hereby obtaining a (possibly) approximated occurrence of the query graph within the target graph. Approximation is related to node insertions, node deletions and edge deletions possibly intervening in the query graph. The technique manages networks of arbitrary topology. Moreover, edge labels are used to represent and manage the reliability of involved interactions. Some preliminary experimental analysis is also accounted for in the paper.

1 Introduction

Last years have witnessed the collection of enormous amounts of biological information that need to be interpreted for the purposes of molecular biology applications. Such information include mainly string-shaped data (e.g., DNA and protein sequences) but also complex structure ones, notably, protein-protein interaction (PPI) networks and metabolic pathways. Through the last few years, the research in bioinformatics has been mainly directed towards the analysis of string-shaped data and many important results have been obtained (see, e.g., [2] for a survey). Nevertheless, computer-based methods devoted to analyzing complex structure biological data are relevant as well, and strongly needed.

In this respect, as stated by Sharan and Ideker in [11], one of the main modes to compare biological networks is *network querying*, "in which a given network is searched for subnetworks that are similar to a subnetwork query of interest". Such a problem, that parallels that of string pattern matching, "is aimed at transferring biological knowledge within and across species" [11]. In fact, PPI subnetworks may correspond to functional modules made of proteins involved in the same biological processes. Unfortunately, as subgraph isomorphism checking is involved, the applicability of exact approaches to solve network querying is rather limited due to the NP-completeness of the problem [7]. Thus, approaches have been proposed where the search is constrained to simple structures, such as paths and trees [9,10,12], some heuristic methods have been presented to deal with true subgraph queries [15], whereas only a few techniques have been proposed based on exact algorithms, so that their practical applicability is limited to queries that are sparse graphs or containing a small number of nodes [16].

This paper provides a contribution in this setting, by proposing a new technique to network querying, which we briefly summarize below, whose main characteristics are as follows: *(i)* it allows to manage arbitrary topology networks, *(ii)* it allows to take into account reliability values associated with interactions and *(iii)* it is capable of singling out also approximated answers to the query graph, as corresponding to evolution determined variations in the sets of nodes and edges. To the best of our knowledge, this is the first technique that comprises all those three characteristics, as also pointed out in the following Section 3.

To illustrate the main ideas underlying our approach, assume we are given a target protein-protein interaction network $\mathcal{G}^{\mathrm{T}}$ and a (typically much smaller) query network $\mathcal{G}^{\mathrm{Q}}$, and that we are interested in finding a (possibly approximated) occurrence of $\mathcal{G}^{\mathrm{Q}}$ in $\mathcal{G}^{\mathrm{T}}$. To this end, our technique first "focuses" a portion of the target network being relevant to the query one as resulting by aligning the two networks. To do that, a minimum bipartite graph weighted matching [6] is used working on the basis of protein sequence similarities. This initial "global" alignment produces a preliminary solution, whose topology may, however, significantly disagree with that of the query network. Therefore, our algorithm "zooms" towards a suitable solution, that matches to a sufficiently large extent the query topology. This is obtained by *refining* similarity values associated with pairs of proteins in $\mathcal{G}^{\mathrm{Q}}$ and $\mathcal{G}^{\mathrm{T}}$ taking into account topology constraints, and then looking for a new alignment of the networks. The process is iterated, going through a number of alignments until one is obtained that satisfies both protein similarities and topological constraints.

Note that repeatedly computing such global alignments provides some guarantees that the resulting solution remains close to the globally optimum match. Furthermore, differently from other network querying techniques we are aware of, which are typically based on a oil-stain visiting strategy, our global alignment strategy permits to naturally deal with missing edges (possibly corresponding to information missing in the database): this case corresponds to producing an alignment of the query network with a generally unconnected subgraph of the target one.

The rest of the paper is organized as follows. The next section illustrates in detail the proposed approach. Section 3 surveys some related literature. Section 4 discusses some preliminary experimental results we obtained with our technique and, finally, in Section 5 some conclusions are drawn.

2 The Proposed Approach

Before explaining the technique in detail, we give two preliminary definitions useful to formulate the problem under consideration.

Definition 1. (Protein Interaction Graph) A *Protein Interaction Graph* is a weighted (undirected) graph $\mathcal{G} = \langle P, I \rangle$, such that:

– $P = \{p_1, p_2, \ldots, p_n\}$ is the set of nodes, each of which represents a protein;
– $I = \{\langle\{p_i, p_j\}, c_{i,j}\rangle\}$ is the set of weighted edges, each denoting an interaction between proteins, and the label $c_{i,j}$ is the *reliability factor* associated to that interaction.

Definition 2. (Distance Dictionary) Given a query protein interaction graph $\mathcal{G}^{\mathrm{Q}}$ and a target protein interaction graph $\mathcal{G}^{\mathrm{T}}$, the *Distance Dictionary DD* is a set of triplets $\langle p_i^{\mathrm{Q}}, p_j^{\mathrm{T}}, d_{i,j}\rangle$, where p_i^{Q} belongs to $\mathcal{G}^{\mathrm{Q}}$, p_j^{T} belongs to $\mathcal{G}^{\mathrm{T}}$ and $d_{i,j}$ is the distance value associated to the pair p_i^{Q} and p_j^{T}.

Thus, let $\mathcal{G}^{\mathrm{Q}} = \langle P^{\mathrm{Q}}, I^{\mathrm{Q}}\rangle$ and $\mathcal{G}^{\mathrm{T}} = \langle P^{\mathrm{T}}, I^{\mathrm{T}}\rangle$ be two protein interaction graphs. In particular, $\mathcal{G}^{\mathrm{Q}}$ denotes the query network to search for in the target network $\mathcal{G}^{\mathrm{T}}$. Assume that a distance dictionary $DD^{(0)}$ is available that stores information about protein sequence similarities of $\mathcal{G}^{\mathrm{Q}}$ and $\mathcal{G}^{\mathrm{T}}$ (details about the computation of $DD^{(0)}$ will be given in Section 2.1).

At step 0, the algorithm first aligns $\mathcal{G}^{\mathrm{Q}}$ and $\mathcal{G}^{\mathrm{T}}$ by exploiting a minimum bipartite graph weighted matching procedure [6] applied to the bipartite weighted graph $\mathcal{G}^{\mathrm{QT}} = \langle P^{\mathrm{QT}}, I^{\mathrm{QT}}\rangle$ such that:

– $P^{\mathrm{QT}} = P^{\mathrm{Q}} \cup P^{\mathrm{T}}$,
– $I^{\mathrm{QT}} = \{\langle\{p_i^{\mathrm{Q}}, p_j^{\mathrm{T}}\}, d_{i,j}^{(0)}\rangle\}$ is the set of weighted edges, where the label $d_{i,j}^{(0)}$ is the distance score between p_i^{Q} and p_j^{T} as stored in the distance dictionary $DD^{(0)}$.

The result of running the weighted matching algorithm on $\mathcal{G}^{\mathrm{QT}}$ is returned in a dictionary $DD^{\mathrm{S}(0)} \subset DD^{(0)}$ storing the triplets $\langle p_i^{\mathrm{Q}}, p_j^{\mathrm{T}}, d_{i,j}^{(0)}\rangle$ corresponding to computed node pairings.

Before going on with illustrating our algorithm, we need to introduce some further concepts. Thus, define $unmatch^{\mathrm{T}}(DD^{\mathrm{S}(0)})$ the set of nodes $p_j^{\mathrm{T}} \in P^{\mathrm{T}}$ such that *(i)* p_j^{T} is on the shortest path connecting two nodes p_{j1}^{T} and p_{j2}^{T} in P^{T}, *(ii)* the triplets $\langle p_{i1}^{\mathrm{Q}}, p_{j1}^{\mathrm{T}}, d_{i1,j1}^{(0)}\rangle$ and $\langle p_{i2}^{\mathrm{Q}}, p_{j2}^{\mathrm{T}}, d_{i2,j2}^{(0)}\rangle$ belong to $DD^{\mathrm{S}(0)}$, and *(iii)* p_{i1}^{Q} and p_{i2}^{Q} are directly linked by an edge in $\mathcal{G}^{\mathrm{Q}}$. Moreover, define $unmatch^{\mathrm{Q}}(DD^{\mathrm{S}(0)})$ the set of nodes $p_i^{\mathrm{Q}} \in P^{\mathrm{Q}}$ that have not been paired with any node of $\mathcal{G}^{\mathrm{T}}$ in $DD^{\mathrm{S}(0)}$.

Define the *extended dictionary* $DD^{\mathrm{S}(0)}+ = DD^{\mathrm{S}(0)} \cup DD_{\mathrm{in}}^{\mathrm{S}(0)} \cup DD_{\mathrm{del}}^{\mathrm{S}(0)}$, where $DD_{\mathrm{in}}^{\mathrm{S}(0)} = \{\langle \bullet, p_j^{\mathrm{T}}, -\rangle\}$ for p_j^{T} a node in $unmatch^{\mathrm{T}}(DD^{\mathrm{S}(0)})$, and $DD_{\mathrm{del}}^{\mathrm{S}(0)} = \{\langle p_i^{\mathrm{Q}}, \bullet, -\rangle\}$ for p_i^{Q} a node in $unmatch^{\mathrm{Q}}(DD^{\mathrm{S}(0)})$. Let $\mathcal{G}^{\mathrm{S}} = \langle P^{\mathrm{S}}, I^{\mathrm{S}}\rangle$ be the subgraph of $\mathcal{G}^{\mathrm{T}}$ such that P^{S} is the set of nodes of $\mathcal{G}^{\mathrm{T}}$ occurring in $DD^{\mathrm{S}(0)}+$, and the set of edges I^{S} is as follows. An edge is added in $\mathcal{G}^{\mathrm{S}}$ between proteins p_h^{T} and p_k^{T} if *(i)* there is an edge $\langle p_h^{\mathrm{T}}, p_k^{\mathrm{T}}, c_{h,k}\rangle$ in $\mathcal{G}^{\mathrm{T}}$, *(ii)* there is an edge $\langle p_i^{\mathrm{Q}}, p_j^{\mathrm{Q}}, c_{i,j}\rangle$ in $\mathcal{G}^{\mathrm{Q}}$, and *(iii)* the triplets $\langle p_i^{\mathrm{Q}}, p_h^{\mathrm{T}}, d_{i,h}^{(0)}\rangle$ and $\langle p_j^{\mathrm{Q}}, p_k^{\mathrm{T}}, d_{j,k}^{(0)}\rangle$ belong to $DD^{\mathrm{S}(0)}$. Moreover, for those pairs of proteins p_h^{T} and p_k^{T} for which conditions *(ii)* and *(iii)* above hold, but condition *(i)* does not, all the edges in the shortest path connecting p_h^{T} and p_k^{T} in $\mathcal{G}^{\mathrm{T}}$ are added to $\mathcal{G}^{\mathrm{S}}$. The edge labels of $\mathcal{G}^{\mathrm{S}}$ are those of $\mathcal{G}^{\mathrm{T}}$. We refer to $\sigma^{(0)} = \langle \mathcal{G}^{\mathrm{S}}, DD^{\mathrm{S}(0)}+\rangle$ as an *approximate occurrence* of $\mathcal{G}^{\mathrm{Q}}$ in $\mathcal{G}^{\mathrm{T}}$.

Note that $\sigma^{(0)}$ may well encode a suitable matching for $\mathcal{G}^{\mathrm{Q}}$ in $\mathcal{G}^{\mathrm{T}}$ or, otherwise, some relevant topological differences might be there significantly distinguishing $\mathcal{G}^{\mathrm{Q}}$ and $\mathcal{G}^{\mathrm{S}}$. In order to evaluate the “quality” of $\sigma^{(0)}$, we introduce a measure

Algorithm for protein interaction graph querying

```
Input:
        a basic distance dictionary $DD^{(0)}$;
        a query protein interaction graph $\mathcal{G}^{\mathrm{Q}} = \langle P^{\mathrm{Q}}, I^{\mathrm{Q}} \rangle$;
        a target protein interaction graph $\mathcal{G}^{\mathrm{T}} = \langle P^{\mathrm{T}}, I^{\mathrm{T}} \rangle$;
        real values $\pi_{\mathrm{ins}}$, $\pi_{\mathrm{del}}$, $\pi_{\mathrm{egd}}$, $\pi_{\mathrm{cm}}$, $I_{\mathrm{MAX}}$, $\alpha$, $\beta$, $\gamma$;
        an integer value MaxIteration
        a threshold value $D_{th}$;
Ouput:
        an approximate occurrence $\sigma^*$ of $\mathcal{G}^{\mathrm{Q}}$ on $\mathcal{G}^{\mathrm{T}}$ s.t. $D^{\sigma^*} \leq D_{th}$;

 1: $h = 0$;
 2: for $k = 1$ to MaxIteration do
 3:     compute $\sigma^{(\mathrm{h})} = \langle \mathcal{G}^{\mathrm{S}}, DD^{\mathrm{S(h)}}+ \rangle$ solving minimum bipartite weighted
        matching problem on $\mathcal{G}^{\mathrm{QT}} = \langle P^{\mathrm{QT}}, I^{\mathrm{QT}} \rangle$ s.t.
            $P^{\mathrm{QT}} = P^{\mathrm{Q}} \cup P^{\mathrm{T}}$,
            $I = \{\langle p_i^{\mathrm{Q}}, p_j^{\mathrm{T}}, d_{i,j}^{(\mathrm{h})} \rangle\}$ if $\langle p_i^{\mathrm{Q}}, p_j^{\mathrm{T}}, d_{i,j}^{(\mathrm{h})} \rangle \in DD^{(\mathrm{h})}$;
 4:     compute $D^{\sigma^{(\mathrm{h})}}$;
 5:     if $(D^{\sigma^{(\mathrm{h})}} > D_{th})$
 6:         $h = h + 1$;
 7:         for each $\langle p_i^{\mathrm{Q}}, p_j^{\mathrm{T}}, d_{i,j}^{(\mathrm{h\text{-}1})} \rangle \in DD^{\mathrm{S(h\text{-}1)}}$
 8:             refine $DD^{(\mathrm{h\text{-}1})}$ to obtain $DD^{(\mathrm{h})}$ using:
 9:                 $d_{i,j}^{(\mathrm{h})} = d_{i,j}^{(\mathrm{h\text{-}1})} + \alpha \cdot \mu'_{\mathrm{ins}} \cdot d_{\mathrm{ins}} + \beta \cdot \mu'_{\mathrm{del}} \cdot d_{\mathrm{del}} + \gamma \cdot \mu'_{\mathrm{egd}} \cdot d_{\mathrm{egd}} +$
                    $-\mu'_{\mathrm{cm}} \cdot d_{\mathrm{cm}}$;
10:     else stop and return $\sigma^* = \sigma^{(\mathrm{h})}$;
11: return "No solution found."
```

Fig. 1. The proposed algorithm

of the "distance" between subgraphs, which is encoded in a *distance score* $D^{\sigma^{(0)}}$ that, for the sake of the readability, will be detailed below in Section 2.1. For the moment being, let us just state that the larger $D^{\sigma^{(0)}}$ is, the more $\mathcal{G}^{\mathrm{S}}$ differs from $\mathcal{G}^{\mathrm{Q}}$. Thus, we are going to consider $\sigma^{(0)}$ an acceptable solution if the corresponding $D^{\sigma^{(0)}}$ is less than a given fixed quality threshold D_{th}.

We can thus resume to illustrating our algorithm. Its next step is to evaluate $D^{\sigma^{(0)}}$ for $\sigma^{(0)}$ and compare it to D_{th}. If $D^{\sigma^{(0)}} \leq D_{th}$, then $\sigma^{(0)}$ is returned as the output. Otherwise, a further minimum bipartite graph weighted matching step is performed as explained below. Let $\sigma^{(\mathrm{h})} = \langle \mathcal{G}^{\mathrm{S}}, DD^{\mathrm{S(h)}}+ \rangle$ be the approximate occurrence computed at the generic step h of the algorithm using the dictionary $DD^{(\mathrm{h})}$ such that $D^{\sigma^{(\mathrm{h})}} > D_{th}$. The next run of the bipartite weighted matching algorithm uses an updated dictionary $DD^{(\mathrm{h+1})}$ obtained from $DD^{(\mathrm{h})}$ and $DD^{\mathrm{S(h)}}+$ as explained next. Initially, $DD^{(\mathrm{h+1})}$ is set equal to $DD^{(\mathrm{h})}$, then some of its entries are *refined*, using $DD^{\mathrm{S(h)}}+$ as follows. Let:

$$d_{i,j}^{(\mathrm{h+1})} = d_{i,j}^{(\mathrm{h})} + \alpha \cdot \mu_{\mathrm{ins}} \cdot \left(\frac{1 - d_{i,j}^{(\mathrm{h})}}{C_i \cdot I_{\mathrm{MAX}}}\right) + \beta \cdot \mu_{\mathrm{del}} \cdot (1 - d_{i,j}^{(\mathrm{h})}) + \gamma \cdot \mu_{\mathrm{egd}} \cdot \left(\frac{1 - d_{i,j}^{(\mathrm{h})}}{C_i}\right) + \qquad (1)$$

$$-\mu_{\mathrm{cm}} \cdot \left(\frac{d_{i,j}^{(\mathrm{h})}}{C_i}\right)$$

where:

- the triplet $\langle p_i^{\mathrm{Q}}, p_j^{\mathrm{T}}, d_{i,j}^{(\mathrm{h})} \rangle$ belongs to $DD^{\mathrm{S(h)}}$,
- the term C_i is defined as the sum of the reliability factors of the edges incident on p_i^{Q},
- I_{MAX} serves the purpose of bounding from above the number of insertions per single edge that we use in the computation,
- μ_{ins} is the penalty score for node insertions, μ_{del} that for node deletions, μ_{egd} that for edge deletions,
- μ_{cm} is a bonus score that rewards correct matches of edges,
- α, β, γ are real values used to weigh the penalty factors μ_{ins}, μ_{del} and μ_{egd} so that $\alpha + \beta + \gamma = 1$.

The rationale of the formula, whose terms will be detailed in the following Section 2.1, is that of modifying the original values of protein similarity in such a way as to take into account information about the topology mismatches of the current solution. By the virtue of this update, the following run of the bipartite weighted matching produces a new solution $\sigma^{(\mathrm{h+1})}$.

Iterations proceed until to either a good approximate solution σ^* is found (that is, $D^{\sigma*} \leq D_{th}$) or, otherwise, a maximum number of iterations (MaxIteration) is reached, in which case no solution is returned.

A snapshot of the algorithm is shown in Figure 1.
The following result holds.

Proposition 1. *Let n and m be the number of nodes in the target and query networks, respectively. In the worst case the algorithm runs in $O(\text{MaxIteration} \cdot n^3)$ time.*

Proof. The shortest path between each pair of nodes in the target network can be pre-computed by the Floyd-Warshall algorithm in $O(n^3)$. During each iteration, two steps are performed. The first one is the computation of a potential solution, obtained by solving a bipartite graph maximum weight matching. The second step is the refinement of the similarity values associated with matching nodes. The time required to compute the maximum weight matching of a bipartite graph made of $\overline{n}$ nodes is $O(\overline{n}^3)$ [6]. Since n is always larger than m, the maximum number of nodes in the bipartite graph is $O(n)$, thus the first step costs $O(n^3)$. The refinement step costs $O(m^2)$ because the number of the edges in the query graph is at most m^2 and all the edges (interactions) have to be explored once to refine the similarities of corresponding nodes. The maximum number of iterations is MaxIteration, thus, overall, the algorithm runs in $O(\text{MaxIteration} \cdot n^3)$ time.

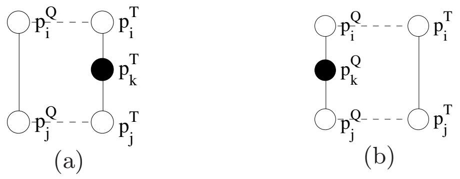


Fig. 2. (a) Node insertion; (b) node deletion

2.1 Technical Details

This section is devoted to illustrate some technical details regarding the parameters and other concepts we have used above.

Basic distance dictionary. As already pointed out, a preprocessing of the protein interaction graphs $\mathcal{G}^{\mathrm{Q}}$ and $\mathcal{G}^{\mathrm{T}}$ in input is necessary in order to evaluate the sequence similarity of pairs of proteins $(p^{\mathrm{Q}}, p^{\mathrm{T}})$ such that p^{Q} belongs to $\mathcal{G}^{\mathrm{Q}}$ and p^{T} belongs to $\mathcal{G}^{\mathrm{T}}$. All information obtained during the preprocessing stage are stored in the basic distance dictionary $DD^{(0)}$, that is computed as follows. The Blast 2 sequences algorithm [14] is executed to align the amino acid sequences of pairs of proteins from $\mathcal{G}^{\mathrm{Q}}$ and $\mathcal{G}^{\mathrm{T}}$, respectively. The resulting BLAST E-values are used to compute a distance score $d_{i,j}^{(0)}$ for each pair of nodes p_i^{Q} of $\mathcal{G}^{\mathrm{Q}}$ and p_j^{T} of $\mathcal{G}^{\mathrm{T}}$, according to the following formula:

$$d_{i,j}^{(0)} = \begin{cases} 1, & \text{if } E \geq 10^{-2} \\ 1 - 2^{\frac{20}{\log E}}, & \text{if } E < 10^{-2} \end{cases}$$

where E is the BLAST E-value as returned by Blast 2 on input p_i^{Q} and p_j^{T}.

Note that the E-value can assume, in general, values greater than 1, and the lower it is, the more similar the protein sequences are. The formula reported above serves the purpose of both normalizing the distance score thus that it varies between 0 and 1 and obtaining more significant variations when the E-value reaches very small values (corresponding to very similar sequences).

Node insertion/deletion. As pointed out in the Introduction, given a query graph in input, our approach aims at searching for its *approximate* occurrences in the target network. In fact, as pointed out in [1], during the evolution of an organism, some events may occur that modify the associated network structure. Those events are *gene duplication*, that causes the addition of new nodes, and *link dynamics*, corresponding to gain and loss of interactions through mutations in existing proteins. In its turn, a gene duplication may be associate to both *node insertions* and *node deletions* [3,12].

Thus, a node insertion event may be associated to the presence of one or more surplus nodes in the path connecting two nodes p_i^{T} and p_j^{T} in the target network, when they are recognized to correspond to two nodes p_i^{Q} and p_j^{Q} in the query network, connected by just one edge. Figure 2(a) clarifies the issue, where the case of a single node insertion is represented.

To take into account node insertions, we define the *number of node insertions* between each pair of nodes p_i^{T} and p_j^{T} belonging to a connected subgraph of $\mathcal{G}^{\mathrm{T}}$ w.r.t. the query network $\mathcal{G}^{\mathrm{Q}}$ as the number of nodes in the shortest path linking p_i^{T} and p_j^{T} in $\mathcal{G}^{\mathrm{T}}$.

A node deletion event occurs when there is some node in the query graph that does not correspond to any node in the target network (see Figure 2(b)). This is taken care of using scores, as will be detailed below.

Distance Score. The distance score D^{σ} for an approximate occurrence σ is obtained by evaluating: *(i)* protein sequence similarity, *(ii)* network topology, *(iii)* number of node insertions, *(iv)* number of node deletions and *(v)* number of edge deletions, where edge deletions are intended in terms of edges that occur in the query but not in the target graph, and that are interpreted as lack or incorrectness of information.

Thus, let $\mathcal{G}^{\mathrm{Q}}$ be the query protein interaction graph, $\mathcal{G}^{\mathrm{T}}$ be the target protein interaction graph and $\sigma^{(\mathrm{h})} = \langle \mathcal{G}^{\mathrm{S}}, DD^{\mathrm{S(h)}}+\rangle$ ($DD^{\mathrm{S(h)}}+ = DD^{\mathrm{S(h)}} \cup DD_{\mathrm{in}}^{\mathrm{S(h)}} \cup DD_{\mathrm{del}}^{\mathrm{S(h)}}$) be an approximate occurrence of $\mathcal{G}^{\mathrm{Q}}$ on $\mathcal{G}^{\mathrm{T}}$. The distance score $D^{\sigma^{(\mathrm{h})}}$ associated to $\sigma^{(\mathrm{h})}$ is computed according to the following formula:

$$D^{\sigma^{(\mathrm{h})}} = \sum_{\langle p_i^{\mathrm{Q}}, p_j^{\mathrm{T}}, d_{i,j}^{(\mathrm{h})}\rangle \in DD^{\mathrm{S(h)}}} d_{i,j}^{(\mathrm{h})} + \mu_{\mathrm{ins}}^{\mathrm{S}} + \mu_{\mathrm{del}}^{\mathrm{S}} + \mu_{\mathrm{egd}}^{\mathrm{S}} - \mu_{\mathrm{cm}}^{\mathrm{S}} \qquad (2)$$

where $d_{i,j}^{(\mathrm{h})}$ is the distance score of nodes p_i^{Q} and p_j^{T} as stored in $DD^{\mathrm{S(h)}}$ (if are such a triplet exists), $\mu_{\mathrm{ins}}^{\mathrm{S}}$ is the penalty score for node insertions, $\mu_{\mathrm{del}}^{\mathrm{S}}$ is the penalty score associated to node deletions, $\mu_{\mathrm{egd}}^{\mathrm{S}}$ is the penalty score associated to edge deletions and $\mu_{\mathrm{cm}}^{\mathrm{S}}$ is a bonus score to reward presumably correct matches. In particular, the three penalty scores are computed as follows:

– Let $E = \{\langle\{p_i^{\mathrm{Q}}, p_l^{\mathrm{Q}}\}, c_{i,l}\rangle\}$ be the set of edges in $\mathcal{G}^{\mathrm{Q}}$, each of which corresponding to a pair of triplets $\langle p_i^{\mathrm{Q}}, p_j^{\mathrm{T}}, d_{i,j}^{(\mathrm{h})}\rangle$ and $\langle p_l^{\mathrm{Q}}, p_k^{\mathrm{T}}, d_{l,k}^{(\mathrm{h})}\rangle$ in $DD^{\mathrm{S(h)}}$. Then:

$$\mu_{\mathrm{ins}}^{\mathrm{S}} = \sum_{\langle\{p_i^{\mathrm{Q}}, p_l^{\mathrm{Q}}\}, c_{i,l}\rangle \in E} \pi_{\mathrm{ins}} \cdot n_{\mathrm{ins}} \cdot c_{i,l}$$

where π_{ins} is a fixed given penalty associated to a single node insertion and n_{ins} is the number of nodes on the shortest path between p_j^{T} and p_k^{T} (if any).

– $\mu_{\mathrm{del}}^{\mathrm{S}} = |DD_{\mathrm{del}}^{\mathrm{S(h)}}| \cdot \pi_{\mathrm{del}}$ where π_{del} is the penalty associated to a single node deletion.

– Let $F = \{\langle\{p_i^{\mathrm{Q}}, p_l^{\mathrm{Q}}\}, c_{i,l}\rangle\}$ be the set of edges in $\mathcal{G}^{\mathrm{Q}}$, each of which corresponding to a pair of triplets $\langle p_i^{\mathrm{Q}}, p_j^{\mathrm{T}}, d_{i,j}^{(\mathrm{h})}\rangle$ and $\langle p_l^{\mathrm{Q}}, p_k^{\mathrm{T}}, d_{l,k}^{(\mathrm{h})}\rangle$ in $DD^{\mathrm{S(h)}}$ such that p_j^{T} and p_k^{T} are non connected in $\mathcal{G}^{\mathrm{T}}$. Then:

$$\mu_{\mathrm{egd}}^{\mathrm{S}} = \sum_{\langle\{p_i^{\mathrm{Q}}, p_l^{\mathrm{Q}}\}, c_{i,l}\rangle \in F} \pi_{\mathrm{egd}} \cdot c_{i,l}$$

where π_{egd} is a fixed given penalty associated to a single edge deletion w.r.t. $\mathcal{G}^{\mathrm{Q}}$.

The bonus score $\mu^{\mathrm{S}}_{\mathrm{cm}}$, is computed as follows:

– Let $G = \{\langle\{p^{\mathrm{Q}}_i, p^{\mathrm{Q}}_l\}, c_{i,h}\rangle\}$ be the set of edges in $\mathcal{G}^{\mathrm{Q}}$, each of which corresponds to a pair of triplets $\langle p^{\mathrm{Q}}_i, p^{\mathrm{T}}_j, d^{(\mathrm{h})}_{i,j}\rangle$ and $\langle p^{\mathrm{Q}}_l, p^{\mathrm{T}}_k, d^{(\mathrm{h})}_{l,k}\rangle$ in $DD^{\mathrm{S(h)}}$, such that the edge $\langle\{p^{\mathrm{T}}_j, p^{\mathrm{T}}_k\}, c_{j,k}\rangle$ is in $\mathcal{G}^{\mathrm{T}}$. Then:

$$\mu^{\mathrm{S}}_{\mathrm{cm}} = \sum_{\langle\{p^{\mathrm{Q}}_i, p^{\mathrm{Q}}_l\}, c_{i,l}\rangle \in G} \pi_{\mathrm{cm}} \cdot \frac{c_{i,l} + c_{j,k}}{2}$$

where π_{cm} is a fixed given score associated to the correct match between the two edges in $\mathcal{G}^{\mathrm{Q}}$ and $\mathcal{G}^{\mathrm{T}}$.

Note that, in the formulae above, reliability factors c_{il} are exploited in order to weigh penalty and bonus scores between proteins by the probabilities that the corresponding interactions actually hold.

Refined similarity scores. Let $\mathcal{G}^{\mathrm{Q}}$ be the query protein interaction graph, $\mathcal{G}^{\mathrm{T}}$ be the target protein interaction graph, $DD^{(\mathrm{h})}$ be a distance dictionary involving all the pairs of proteins of $\mathcal{G}^{\mathrm{Q}}$ and $\mathcal{G}^{\mathrm{T}}$. Furthermore, let $\sigma^{(\mathrm{h})} = \langle \mathcal{G}^{\mathrm{S}}, DD^{\mathrm{S(h)}}+\rangle$, s.t. $DD^{\mathrm{S(h)}}+ = DD^{\mathrm{S(h)}} \cup DD^{\mathrm{S(h)}}_{\mathrm{in}} \cup DD^{\mathrm{S(h)}}_{\mathrm{del}}$ and $DD^{\mathrm{S(h)}} \subset DD^{(\mathrm{h})}$, be an approximate occurrence of $\mathcal{G}^{\mathrm{Q}}$ in $\mathcal{G}^{\mathrm{T}}$. The penalty scores μ_{ins}, μ_{del} and μ_{egd}, necessary to compute the refined similarities according to formula (1), are evaluated as follows:

– Let $E_i = \{\langle\{p^{\mathrm{Q}}_i, p^{\mathrm{Q}}_l\}, c_{i,l}\rangle\}$ be the set of edges incident onto p^{Q}_i in $\mathcal{G}^{\mathrm{Q}}$, each of which corresponding to a pair of triplets $\langle p^{\mathrm{Q}}_i, p^{\mathrm{T}}_j, d^{(\mathrm{h})}_{i,j}\rangle$ and $\langle p^{\mathrm{Q}}_l, p^{\mathrm{T}}_k, d^{(\mathrm{h})}_{l,k}\rangle$ in $DD^{\mathrm{S(h)}}$. Then:

$$\mu_{\mathrm{ins}} = \sum_{\langle\{p^{\mathrm{Q}}_i, p^{\mathrm{Q}}_l\}, c_{i,l}\rangle \in E_i} \min\{n_{\mathrm{ins}}, I_{\mathrm{MAX}}\} \cdot c_{i,l}.$$

where I_{MAX}, n_{ins} and I_{MAX} are as explained in the previous section.

– Let $DD_{\mathrm{del},i}$ be a subset of $DD^{\mathrm{S(h)}}+$ that contains the triplets $\langle p^{\mathrm{Q}}_l, \bullet, -\rangle$ such that the nodes p^{Q}_l are connected by an edge to p^{Q}_i in $\mathcal{G}^{\mathrm{Q}}$, and $n_{\mathrm{adj},i}$ be the number of nodes directly linked by an edge to p^{Q}_i in $\mathcal{G}^{\mathrm{Q}}$. Then:

$$\mu_{\mathrm{del}} = \frac{|DD_{\mathrm{del},i}|}{n_{\mathrm{adj},i}}$$

– Let $F_i = \{\langle\{p^{\mathrm{Q}}_i, p^{\mathrm{Q}}_l\}, c_{i,l}\rangle\}$ be the set of edges incident on p^{Q}_i in $\mathcal{G}^{\mathrm{Q}}$, each corresponding to the triplets $\langle p^{\mathrm{Q}}_i, p^{\mathrm{T}}_j, d^{(\mathrm{h})}_{i,j}\rangle$ and $\langle p^{\mathrm{Q}}_l, p^{\mathrm{T}}_k, d^{(\mathrm{h})}_{l,k}\rangle$ in $DD^{\mathrm{S(h)}}$ such that there does not exist any path in $\mathcal{G}^{\mathrm{T}}$ connecting p^{T}_j and p^{T}_k. Then:

$$\mu_{\mathrm{egd}} = \sum_{\langle\{p^{\mathrm{Q}}_i, p^{\mathrm{Q}}_l\}, c_{i,l}\rangle \in F_i} c_{i,l}$$

The bonus score μ_{cm} of formula (1) is computed as follows:

– let $G_i = \{\langle\{p_i^{\mathrm{Q}}, p_l^{\mathrm{Q}}\}, c_{i,l}\rangle\}$ be the set of edges incident on p_i^{Q} in $\mathcal{G}^{\mathrm{Q}}$, each corresponding to the triplets $\langle p_i^{\mathrm{Q}}, p_j^{\mathrm{T}}, d_{i,j}^{(\mathrm{h})}\rangle$ and $\langle p_l^{\mathrm{Q}}, p_k^{\mathrm{T}}, d_{l,k}^{(\mathrm{h})}\rangle$ in $DD^{\mathrm{S(h)}}$ such that there exists the edge $\langle\{p_j^{\mathrm{T}}, p_k^{\mathrm{T}}\}, c_{j,k}\rangle$ in $\mathcal{G}^{\mathrm{T}}$. Then:

$$\mu_{\mathrm{cm}} = \sum_{\langle\{p_i^{\mathrm{Q}}, p_l^{\mathrm{Q}}\}, c_{i,l}\rangle \in G} \frac{c_{i,l} + c_{j,k}}{2}$$

3 Related Work

Network querying techniques as applied to biological networks, which are briefly surveyed below, can be divided in two main categories: those ones searching for efficient solutions under particular topological constraints imposed on the query graph, e.g., the query is required to be a path, and those more general ones that, like ours, manage arbitrary query topologies.

Specific query topologies. Kelley et al. developed *PathBLAST* [9], a procedure to align two PPI networks in order to identify conserved interaction pathways and complexes. It searches for high scoring alignments involving two paths, one for each network, in which proteins of the first path are paired with putative homologs occurring in the same order in the second path.

The algorithm *MetaPathwayHunter*, presented in [10] solves the problem of querying metabolic networks, where the queries are multi-source trees. MetaPathwayHunter searches the networks for approximated matching, allowing node insertions (limited to one node), whereas deletions are not dealt with.

The references [12] and [3] illustrate two techniques for network querying, called *QPath* and *QNet*. In particular, QPath queries a PPI network by a query pathway consisting of a linear chain of interacting proteins. The algorithm works similarly to sequence alignment, so that proteins in analogous positions have similar sequences. PPI networks interactions reliability scores are considered, and insertions and deletions are allowed. QNet is an extension of QPath in which queries can take the form of trees or graphs with limited tree-width.

As already stated, differently from the approaches surveyed above, our technique deals with arbitrary topologies in both the query and the target networks. In that, it is more closely related to the works described below.

General query topologies. The system *GenoLink* presented in [4] is able to integrate data from different sources (e.g., databases of proteins, genes, organisms, chromosomes) and to query the resulting data graph by graph patterns with constraints attached to both vertices and edges; a query result is the set of all subgraphs of the target graph that are similar to the query pattern and satisfy the constraints. The goals of [4] are clearly different from our own, since the aim here is that of comparing heterogeneous graphs via constrained network querying.

Ferro et al. in [5] present a tecnique called *NetMatch*, a Cytoscape plug-in for network querying allowing for approximated querying. A query in NetMatch is a graph in which some nodes are specified and others are wildcards (which can match an unspecified number of elements). Although dealing, as we do, with approximate network querying, the technique in [5] mainly focuses on topological similarity, whereas our results are deeply influenced by information about *node similarities* as well which, we argue, are to be considered essential for the analysis of PPI networks.

In [15], a tool for querying large graph data-sets, called *SAGA*, is described. The tool allows for searching for all the subgraphs contained in a graph data-set that are similar to a query graph. The authors define a concept of similarity between subgraphs based on the structural distances of the match, the number of mismatches and the number of gaps. An index-based heuristic is exploited for the purposes of the query processing. SAGA has been successfully exploited to query biological pathways and literature data-sets, although it shows some limitations in dealing with dense and with large query graphs.

In [16] the problems of path matching and graph matching are considered. An exact algorithm is presented to search for subgraphs of arbitrary structure in a large graph, grouping related vertices in the target network for each vertex in the query. Being an exact matcher, it is only practical for queries having a number of nodes as large as 20 in the query network, though its performances improve if the query is a sparse graph. However, for the same reason of being an exact matcher, this is the reference technique we chose in our comparative experiments (see, below, Section 4.1).

The techniques presented in [4,5,15,16], are closely related to our own, but with the following differences: (*i*) none of them exploits edge labels to manage interaction reliability factors which, considered the diverse trustability of methods used to establish the various protein interactions to hold, are practically very relevant to correctly single out, in the target network, highly-probable matchers of the query network; (*ii*) our technique does not imply any constraint on the number of involved nodes or the density of the query subgraph; (*iii*) as far as we know, our technique is the first naturally dealing with *edge deletions*, thus that possible incompleteness in the available information about interactions is dealt with.

4 Experimental Results

In this section, we illustrate some preliminary results we obtained by running our algorithm on the four PPI networks of *S. cerevisiae*, *D. melanogaster*, *C. elegans* and *H. sapiens*. In particular, as described in detail in Sections 4.1, we exploited *S. cerevisiae*, *D. melanogaster* and *C. elegans* networks to compare our results with those presented in [16] for the same organisms. Section 4.2 illustrates how the *S. cerevisiae* and *H. sapiens* networks have been queried to further assess the ability of our technique in recognizing conservations across species. To this end, we considered some well characterized groups of both yeast and human proteins for which the biological processes they are involved in are understood.

For the experiments, we chose to fix the parameter values as follows. Factors π_{ins}, π_{del}, π_{egd} and π_{cm} have been all set to one, α has been set to 0.3, β to 0.1, γ to 0.6 and I_{MAX} to 5. Note that, within the preliminary test experiments accounted for below, we did not perform a fine tuning on such parameters, which is deferred to further experimental work. The algorithm was implemented on a 3.4 GHz Pentium *IV* with 4 GB RAM. The resulting running times of the experiments we conducted vary from a minimum of 53 seconds to a maximum of about 17 minutes.

4.1 Querying *D. melanogaster* and *C. elegans* by *S. cerevisiae*

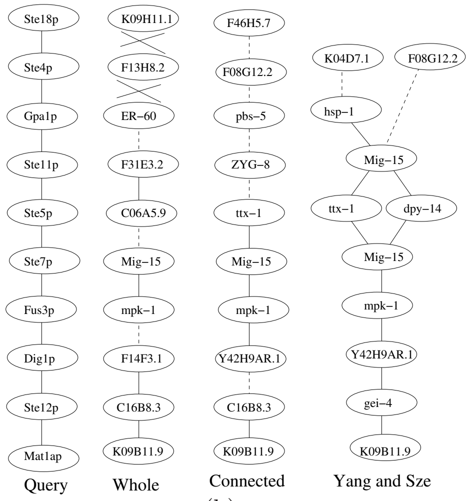
We compared our method with the one presented in [16]. In particular, we focused first on a path of the *S. cerevisiae* network to query the *C. elegans* network. This path, denoted by "Query" in Figure 3(b), corresponds to the longer mating-pheromone response pathway from the protein interaction network of *S. cerevisiae* [8]. The same figure also shows the output returned by the approach of Yang and Sze [16], and the outputs returned by our algorithm when run on "whole" the *C. elegans* network and on a "connected" part of it, resp. In the figures, graph nodes are labeled by protein names, dashed edges correspond to node insertions, whereas cross edges represent edge deletions. In particular, we obtained at most two node insertions per edge on this example, whereas Yang and Sze fixed a priori the maximum number of node insertions per edge to be equal to one (our algorithm does not require any such a limitation about the maximum number of node insertions to be fixed). The table in Figure 3(a) reports the E-values corresponding to the solutions returned by the considered approaches.

Looking at the results shown in Figure 3(b), the first important observation is that our algorithm is able to associate the MAP kinases *Fus3p* of *S. cerevisiae* and *mpk-1* of *C. elegans*, whose associated E-value equals zero, on that agreeing with the results reported in [16]. The results of both our executions agree with Yang and Sze also for the *S. cerevisiae* and *C. elegans* proteins *Ste7p/Mig-15* and *Mat1ap/K09B11.9*. Furthermore, the result on the connected subnetwork of *C. elegans* is the same of Yang and Sze also for *Ste4p/F08G12.2*, *Ste5p/ttx-1* and *Dig1p/Y42H9AR.1*. On the contrary, both executions of our algorithm returned a different result for the protein *Ste11p* of *S. cerevisiae* that, in [16], is paired again with *Mig-15* (which was paired with *Ste7p* as well). This incongruence might be caused for Yang and Sze admit multiple pairings of proteins; on the contrary, our approach search for one-to-one pairings. In any case, the result our approach returns is significant from the biological standpoint, since proteins *Ste11p* and *F31E3.2* both belong to putative serine/threonine-protein kinase family (as well as *Ste7p* and *Mig-15*). Results returned in both executions of our algorithm are slightly better, in terms of E-values, than those reported in [16] for the two proteins *Ste12p* and *Gpa1p*. Furthermore, we are able to pair also protein *Ste18p*, that Yang and Sze do not associate to any protein of *C. elegans*.

The second example is one where the query is a yeast graph with general topology representing a related functional module from Spirin and Mirny [13]. Figure 4(a) illustrates the yeast query, Figure 4(b) shows a table containing the E-values corresponding to the results returned by our algorithm (applied on connected subnetworks) and by Yang and Sze, resp., when applied on both *C. elegans* and *D. melanogaster*. Figure 4(c) and Figure 4(d) illustrate the corresponding result subgraphs. In this experiment, the bait used to query *C. elegans* and

p_i^Q	E-value Yang and Sze	E-value Whole net	E-value subnet
Mat1ap	$3 \cdot 10^{-3}$	$3 \cdot 10^{-3}$	$3 \cdot 10^{-3}$
Ste12p	$1 \cdot 10^{-3}$	$2 \cdot 10^{-4}$	$2 \cdot 10^{-4}$
Dig1p	$2 \cdot 10^{-5}$	$3 \cdot 10^{-4}$	$2 \cdot 10^{-5}$
Fus3p	0	0	0
Ste7p	$7 \cdot 10^{-34}$	$7 \cdot 10^{-34}$	$7 \cdot 10^{-34}$
Ste5p	$5 \cdot 10^{-3}$/ $6 \cdot 10^{-3}$	$2 \cdot 10^{-3}$	$5 \cdot 10^{-3}$
Ste11p	$2 \cdot 10^{-35}$	$2 \cdot 10^{-17}$	$3 \cdot 10^{-14}$
Gpa1p	$1 \cdot 10^{-2}$/–	$4 \cdot 10^{-3}$	$2 \cdot 10^{-3}$
Ste4p	$2 \cdot 10^{-14}$/ $4 \cdot 10^{-15}$	$7 \cdot 10^{-10}$	$4 \cdot 10^{-15}$
Ste18p	–	$1 \cdot 10^{-3}$	$2 \cdot 10^{-3}$

(a)



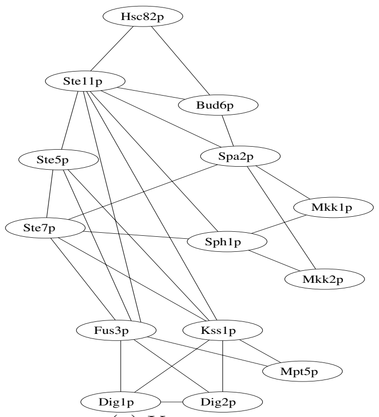
(b)

Fig. 3. Comparison on the longer mating-pheromone response pathway

D. melanogaster networks is a well characterized yeast signalling cascade. This yeast pathway controls peculiar yeast processes that are pheromone response (via $Fus3p$) and pseudohyphal invasive groth pathway (via $Kss1p$) through a so-called $MAPK$ pathway (Mitoge activated protein kinase). The $MAPK$ signalling cascades are likely to be found in all eukaryotic organism although the substrates phosphorylated by these kinases and the final response can be different in different organisms. Thus, in response to the query network, our technique retrives two *C. elegans* and *D. melanogaster* $MAPK$ cascades (Figure 4(c)-left an Figure 4(d)-left), as suggested by the presence of several $MAPK$ (i.e. proteins $mkk-4$, $pmk-1$, $mpk-1$, $jnk-1$ in *C. elegans*, and proteins $ERKA$, $CG7717$ in *D. melanogaster*, resp.) and other S/T kinases (i.e. proteins $mig-15$, $gsk-3$ in *C. elegans* and $CG7001$, *cdc2c*, $CG17161$ in *D. melanogaster*, resp.). This example illustrates well a peculiarity of our approach, that is, trying to find a good compromise between node similarity and network topology. In fact, the solution of [16] presents, in some cases, lower E-values than the correspondent ones in our solution, but our algorithm is able to pair all the proteins of the query network, which the technique in [16] does not, where three node deletions in *C. elegans* and four ones in *D. melanogaster*, respectively, can be observed.

4.2 Querying *H. sapiens* by *S. cerevisiae*

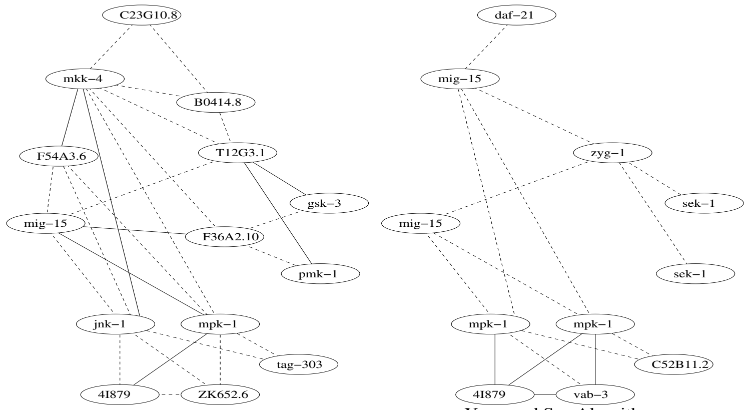
In Figure 5 and Figure 6 two yeast queries are shown that are matched to the target human network. In particular, the query network in Figure 5 concerns proteins that control cell-cycle transitions. The progression through the cell-division cycle in eukaryotes is driven by particular protein kinases (CDK) which trigger the transition to the following phase of the cycle. These enzymes are serine/threonine kinases that require for their activation to be associated with



(a) Yeast query

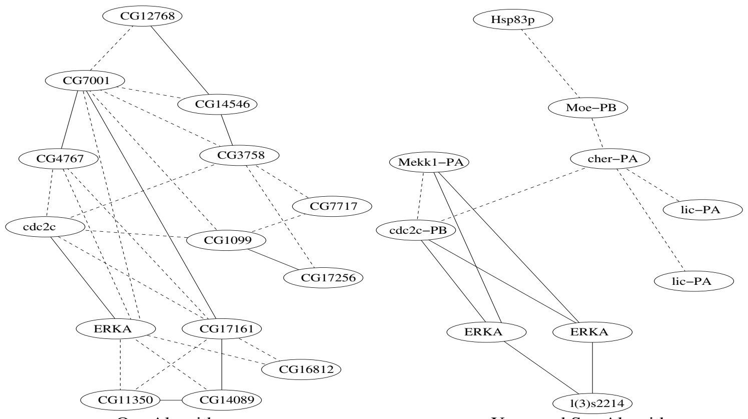
p_i^Q	C. Elegans		D. Melanogaster	
	E-val Our Alg.	E-val Yang and Sze	E-val Our Alg.	E-val Yang and Sze
Hsc82p	$1 \cdot 10^{-3}$	0	$2 \cdot 10^{-3}$	0
Ste11p	$4 \cdot 10^{-20}$	$2 \cdot 10^{-35}$	$2 \cdot 10^{-18}$	-
Bud6p	$1 \cdot 10^{-3}$	-	$3 \cdot 10^{-3}$	$5 \cdot 10^{-4}$
Ste5p	$8 \cdot 10^{-4}$	—	$6 \cdot 10^{-3}$	$6 \cdot 10^{-3}$
Spa2p	$1 \cdot 10^{-4}$	$3 \cdot 10^{-5}$	$7 \cdot 10^{-4}$	$6 \cdot 10^{-3}$
Ste7p	$7 \cdot 10^{-34}$	$7 \cdot 10^{-34}$	$6 \cdot 10^{-21}$	$6 \cdot 10^{-21}$
Sph1p	$4 \cdot 10^{-3}$	-	$1 \cdot 10^{-3}$	-
Mkk1	$8 \cdot 10^{-19}$	$2 \cdot 10^{-44}$	$7 \cdot 10^{-25}$	$5 \cdot 10^{-49}$
Mkk2	$8 \cdot 10^{-24}$	$1 \cdot 10^{-43}$	$3 \cdot 10^{-31}$	$4 \cdot 10^{-48}$
Fus3p	$6 \cdot 10^{-61}$	0	0	0
Kss1p	$4 \cdot 10^{-97}$	$4 \cdot 10^{-97}$	$3 \cdot 10^{-19}$	$7 \cdot 10^{-94}$
Dig1p	$2 \cdot 10^{-5}$	$2 \cdot 10^{-5}$	$8 \cdot 10^{-4}$	—
Dig2p	$1 \cdot 10^{-3}$	$2 \cdot 10^{-5}$	$4 \cdot 10^{-4}$	$1 \cdot 10^{-3}$
Mpt5p	$8 \cdot 10^{-3}$	$8 \cdot 10^{-3}$	$3 \cdot 10^{-3}$	—

(b)



Our Algorithm Yang and Sze Algorithm

(c) C. Elegans



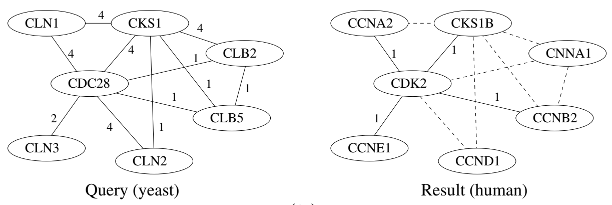
Our Algorithm Yang and Sze Algorithm

(d) D. Melanogaster

Fig. 4. Comparison on the functional module from Spirin and Mirny [13]

p_i^Q	p_j^T	E-value
CDC28	CDK2	0
CLN1	CCNA2	$2 \cdot 10^{-10}$
CKS1	CKS1B	$1 \cdot 10^{-29}$
CLB2	CCNA1	$3 \cdot 10^{-39}$
CLB5	CCNB2	$7 \cdot 10^{-43}$
CLN2	CCND1	$7 \cdot 10^{-6}$
CLN3	CCNE1	$2 \cdot 10^{-4}$

(a)

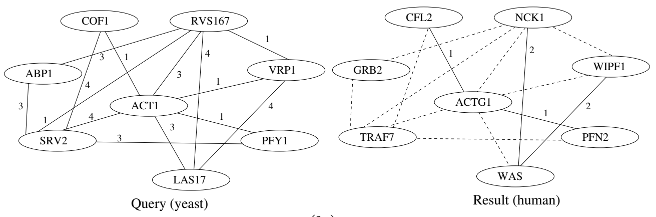


(b)

Fig. 5. Querying *H. sapiens* by *S. cerevisiae*: example 1

p_i^Q	p_j^T	E-value
ACT1	ACTG1	0
COF1	CFL2	$6 \cdot 10^{-27}$
RVS167	NCK1	$9 \cdot 10^{-10}$
VRP1	WIPF1	$4 \cdot 10^{-9}$
PFY1	PFN2	$2 \cdot 10^{-7}$
LAS17	WAS	$2 \cdot 10^{-14}$
SRV2	TRAF7	$4 \cdot 10^{-3}$
ABP1	GRB2	$4 \cdot 10^{-5}$

(a)



(b)

Fig. 6. Querying *H. sapiens* by *S. cerevisiae*: example 2

regulative subunits known as cyclins. The query is composed of the budding yeast *S. cerevisiae* cyclin dependent kinase ($CDC28$) which associates with all different cyclins ($CLN1$, $CLB2$, $CLB5$, $CLN2$, $CLN3$). In yeast, different cyclins work in different phases of the cell cycle binding the same CDK. Mammalian cells, instead, have evolved multiple CDKs, each one working only with some cyclins. Consequently, in the human CDK network retrieved by applying our algorithm, some yeast interactions correspond to multiple-edge interactions in the human. For example, human cyclin D ($CCND1$) does not interact directly with $CDK2$ ($CDK2$) because it binds the homologs $CDK4$ and $CDK6$, but they have as a common partner the inhibitory protein $p21$ ($CDKN1A$) that is found as a node insertion in our approach (not explicitly shown in Figure 5). Instead cyclin $A2$ ($CCNA2$) and cyclin E ($CCNE1$) are directly connected to $CDK2$.

In the second experiment, we queried the human network with the yeast actin-related-proteins graph. Results are illustrated in Figure 6. Actin is well conserved among eukaryotes being a main component of the cytoskeleton. In yeast, it binds several proteins which regulate its polymerization/depolymerization and which are presented in the graph. Human homologs of the yeast proteins have been correctly paired (i.e., $ACT1/ACTG1$, $COF1/CFL2$, $VRP1/WIPF1$, $PFY1/PFN2$, $LAS17/WAS$ in yeast and human, respectively). Furthermore, as in the previous example, the network has increased its complexity moving from yeast to human. Thus, while $PFN2$ and $CFL2$ are still directly linked to actin, an insertion node, not shown in Figure 6, divides the regulators $WIPF1$ and WAS from it.

This latter set of experiments has preliminarily confirmed that our technique is indeed able of retrieving biologically meaningful subgraphs matching the query network in the target one.

5 Concluding Remarks

In this paper we presented an approach to search for approximate occurrences in protein-protein interaction networks based on bipartite graph weighted matching. To summarize, the technique presents the following characteristics: *(i)* it manages networks of arbitrary topology, both for query and for target ones, *(ii)* edge labels are used to represent and manage the reliability of involved interactions and *(iii)* node insertions, node deletions and edge deletions are dealt with. The preliminary experimental results we obtained with it are quite encouraging.

Possible extensions of the technique may regard the possibility of "fixing" some nodes of the query network whose homolog in the target one is known; this may help the biologist to *guide* the algorithm towards solutions where known correspondences between proteins are imposed.

References

1. Berg, J., Lässig, M., Wagner, A.: Structure and evolution of protein interaction networks: a statistical model for link dynamics and gene duplications. BMC Evolutionary Biology 4(1) (2004)
2. Brazma, A., Jonassen, I., Eidhammer, I., Gilbert, D.: Approaches to the automatic discovery of patterns in biosequences. Journal of Computational Biology 5, 279–305 (1995)
3. Dost, B., Shlomi, T., Gupta, N., Ruppin, E., Bafna, V., Sharan, R.: Qnet: A tool for querying protein interaction networks. In: Speed, T., Huang, H. (eds.) RECOMB 2007. LNCS (LNBI), vol. 4453, pp. 1–15. Springer, Heidelberg (2007)
4. Durand, P., Labarre, L., Meil, A., Divol, J.L., Vandenbrouck, Y., Viari, A., Wojcik, J.: Genolink: a graph-based querying and browsing system for investigating the function of genes and proteins. BMC Bioinformatics 21(7) (2006)
5. Ferro, A., Giugno, R., Pigola, G., Pulvirenti, A., Skripin, D., Bader, G.D., Shasha, D.: Netmatch: a cytoscape plugin for searching biological networks. Bioinformatics (2007)
6. Galil, Z.: Efficient algorithms for finding maximum matching in graphs. ACM Comput. Surv. 18(1), 23–38 (1986)
7. Garey, M., Johnson, D.: Computers and intractability: A guide to the theory of NP-completeness. Freeman, New York (1979)
8. Gustin, M.C., Albertyn, J., Alexander, M., et al.: Map kinase pathways in the yeast saccharomyces cerevisiae. Microbiol. Mol. Biol. Rev. 62, 1264–1300 (1998)
9. Kelley, B.P., Sharan, R., Karp, R.M., Sittler, T., Root, D.E., Stockwell, B.R., Ideker, T.: Conserved pathways within bacteria and yeast as revealed by global protein network alignment. Proceedings of the National Academy of Science USA 100(20), 11394–11399 (2003)
10. Pinter, R.Y., Rokhlenko, O., Yeger-Lotem, E., Ziv-Ukelson, M.: Alignment of metabolic pathways. Bioinformatics 21(16), 3401–3408 (2005)
11. Sharan, R., Ideker, T.: Modeling cellular machinery through biological network comparison. Nature Biotechnology 24(4), 427–433 (2006)
12. Shlomi, T., Segal, D., Ruppin, E., Sharan, R.: Qpath: a method for querying pathways in a protein-protein interaction network. BMC Bioinformatics 7 (2006)

13. Spirin, V., Mirny, L.A.: Protein complexes and functional modules in molecular networks. Proceedings of the National Academy of Sciences of the USA 100, 12123–12128 (2003)
14. Tatusova, T.A., Madden, T.L.: Blast 2 sequences, a new tool for comparing protein and nucleotide sequences. FEMS Microbiology Letters 174(2), 247–250 (1999)
15. Tian, Y., Mceachin, R.C., Santos, C., States, D.J., Patel, J.M.: Saga: a subgraph matching tool for biological graphs. Bioinformatics 23(2), 232–239 (2007)
16. Yang, Q., Sze, S.-H.: Saga: a subgraph matching tool for biological graphs. Journal of Computational Biology 14(1), 56–67 (2007)

Matching Spatial Regions with Combinations of Interacting Gene Expression Patterns

Jano van Hemert[1] and Richard Baldock[2]

[1] National e-Science Centre, School of Informatics, University of Edinburgh, UK
J.vanHemert@ed.ac.uk
[2] Human Genetics Unit, Medical Research Council, Edinburgh, UK
Richard.Baldock@hgu.mrc.ac.uk

Abstract. The Edinburgh Mouse Atlas aims to capture *in-situ* gene expression patterns in a common spatial framework. In this study, we construct a grammar to define spatial regions by combinations of these patterns. Combinations are formed by applying operators to curated gene expression patterns from the atlas, thereby resembling gene interactions in a spatial context. The space of combinations is searched using an evolutionary algorithm with the objective of finding the best match to a given target pattern. We evaluate the method by testing its robustness and the statistical significance of the results it finds.

Keywords: gene expression patterns, in situ hybridization, spatio-temporal atlases; evolutionary algorithms.

1 Introduction

The location of expressing genes can be revealed by performing *in-situ hybridisation* on either embryo sections or wholemount embryos. This process uses labelled RNA that binds to mRNA in the cell, which is a good indication the corresponding gene in the cell is active. Essentially, the result is a stained embryo or part thereof, where the stain indicates where the gene is expressing. As these patterns exhibit gradients, and as the process is quite sensitive, the resulting data needs careful examination before inferring the location of a gene expression pattern. The Edinburgh Mouse Atlas Project (EMAP) [1] has a curators office, which performs these examinations and translates these patterns into the common spatio-temporal framework for the developing *Mus Musculus*.

EMAP is a unique resource that captures data in one common spatio-temporal framework, thereby opening the possibilities to perform queries and analyses in both embryo space and embryo development time to explore how genes interact on an inter-cellular level. Currently the spatial data can be queried by defining a pattern in the context of the embryo. The database will then return all gene expression patterns that intersect with the query domain, sorted by similarity with that domain. In addition the spatial patterns can be clustered in terms of spatial similarity to reveal putative syn-expression groups. More sophisticated analysis involving pattern combinations is not however possible. Recently the push is towards gene marker studies that try to redefine the spatial context of embryos in terms of where genes are expressing.

We have developed a methodology to define a given target pattern by the combination of multiple gene expression patterns via several gene interactions operations. The target pattern can be the expression pattern of a particular gene, a pattern defined by a human, a pattern defined by anatomical components or by any other means of defining spatial area within the context of the model mouse embryo. The method searches for a set of genes and combines their expression patterns using predefined operations to closely match the target pattern, thereby attempting to define this pattern spatially. The objective of this study is to measure the robustness of the methodology and validate the significance of the resulting gene interactions. This is important as much noise exists in the acquisition of the patterns as well as much inaccuracy may exist in the target pattern.

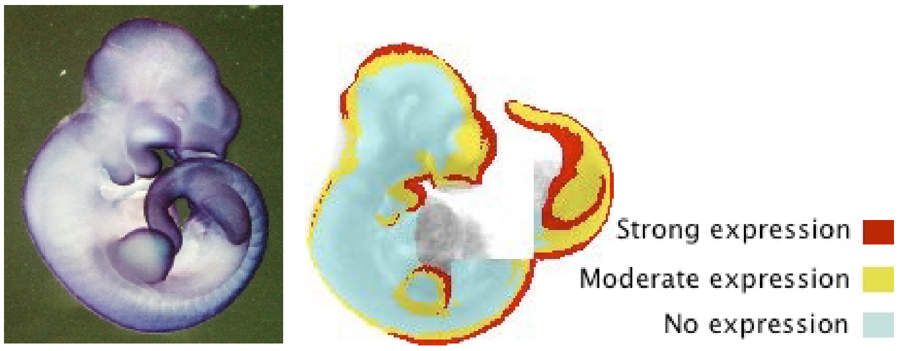
In the next section we describe the Edinburgh Mouse Atlas Project, a spatio-temporal framework for capturing anatomy and gene expression patterns in developing stages of the mouse. Then, in Section 3, we describe the methodology for constructing and searching gene interaction trees. Experiments and results are provided in Section 4. Last, we provide a discussion in Section 5.

2 Edinburgh Mouse Atlas Project

EMAGE (http://genex.hgu.mrc.ac.uk/) is a freely available, curated database of gene expression patterns generated by *in situ* techniques in the developing mouse embryo [1]. It is unique in that it contains standardized spatial representations of the regions of gene expression for each gene, denoted against a set of virtual reference embryo models. As such, the data can be interrogated in a novel and abstract manner by using space to define a query. Accompanying the spatial representations of gene expression patterns are text descriptions of the sites of expression, which also allows searching of the data by more conventional text-based methods terms.

Data is entered into the database by curators that determine the level of expression in each *in situ* hybridization experiment considered and then map those levels on to a standard embryo model. An example of such a mapping is given in Figure 1. The strength of gene expression patterns are classified either as no expression, weak expression, moderate expression, strong expression, or possible detection. Possible detection means the curator is uncertain whether the region exhibits gene expression, hence we exclude these regions from our analyses.

In this study we restrict to a subset of the data contained in the database. This subset of data originates from one study [2] and contains 1970 images of *in situ* gene expression patterns in a wholemount developing mouse embryo model of Theiler Stages 15–19 [3]. The study includes 1131 genes; a subset of genes were screened two or three times. By mapping the strong and moderate expression



(a) Original image shows expression of Hmgb1 in a mouse embryo at Theiler Stage 17

(b) Standard embryo with mapped levels of expression

Fig. 1. An example of curating data; the gene expression in the original image on the left is mapped on to the standard embryo model of equal developmental stage on the right (entry EMAGE:3052 in the online database)

patterns of these images on to the two-dimensional model for Theiler Stage 17 shown in Figure 1(b), we can work with all these patterns at the same time.

3 Combining Gene Expression Patterns

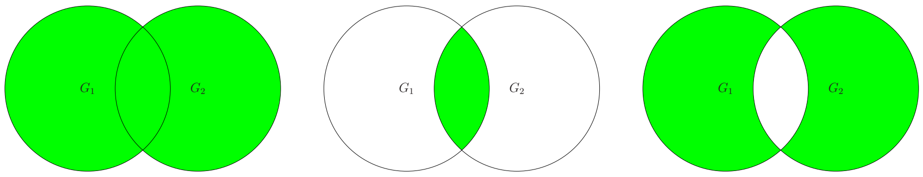
To make possible a directed search for a given target pattern we need to define how patterns can interact, to define how interactions are structured, to define a function that allows to measure the quality of matching two patterns and to have a mechanism whereby we can search the space of pattern interactions. The following sections will discuss each of these in detail.

3.1 Defining the Interaction Patterns

The interaction patterns are shown in Figure 2. Each interaction pattern is an operation on a spatial pattern where the operation, either OR, AND or XOR, is performed over each pixel[1] in the space as defined by the pattern. The AND operation is also referred to as the conjunction of patterns, whole the OR operation is referred to as the disjunction of patterns. This definition on sets takes every pixel location as an item and then a pattern is defined as a a set of pixel locations.

In terms of gene interactions, the AND operation represents two genes that require co-location in time and space in order to express. The XOR operation represents two genes cancelling each other's expression out when co-located. The OR operation merely takes the conjunction and no visible interaction occurs.

[1] Alternatively in a dimension higher than two it will be performed over a voxel.



(a) OR: conjunction (b) AND: disjunction (c) XOR: conjunction except for the disjunction

Fig. 2. Types of interactions of two gene expression patterns G_1 and G_2

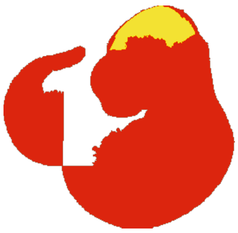
3.2 A Grammar for Interactions

Below, we define a simple BackusNaur form grammar [4] to allow arbitrary large interaction trees to be constructed. In practice, the size of these trees is restricted in the search algorithm. The operations are the binary operations over patterns as defined in the previous section and the existing patterns are unique identifiers to studies in the Edinburgh Mouse Atlas database (http://genex.hgu.mrc.ac.uk/), where in the study we have used the conjunction of strong and moderate strength gene expression patterns.

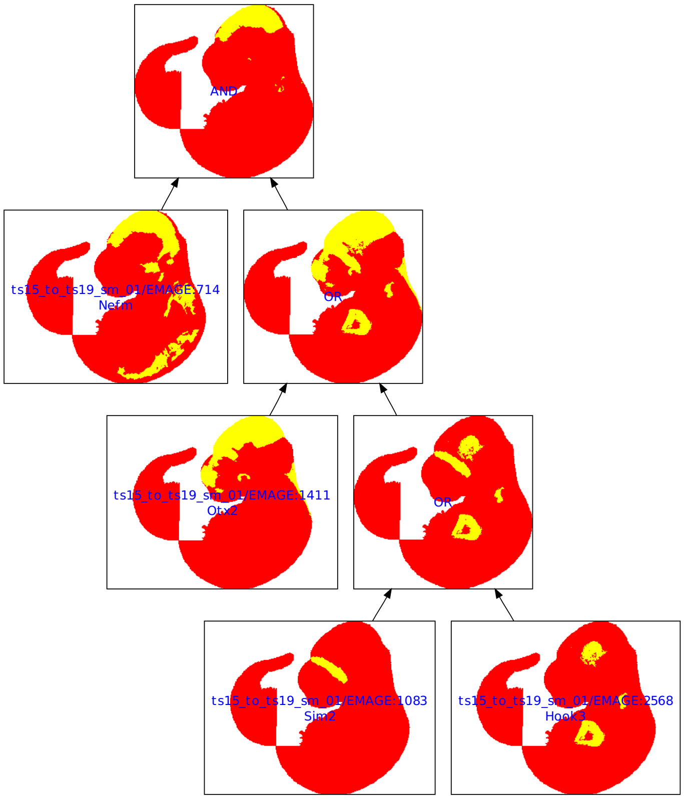
```
<pattern> ::= <existing pattern> | (<pattern> <operation> <pattern>)
<operation> ::=  AND | OR | XOR
<existing pattern> ::= EMAGE:1, EMAGE:2, ..., EMAGE:1970
```

When combining patterns in this fashion it is possible to create the empty set. For example, let us apply AND to two patterns, where one pattern expresses only in the tail and the other pattern only expresses in the head; as these patterns do not overlap, the operation will yield an empty pattern. This has two major consequences. First, sub-trees that produce empty patterns can increase the complexity of a large tree without adding any value. Second, allowing useless sub-trees inflates the size of the search space. To counteract this, we prune these empty patterns from trees by removing their sub-trees. If exactly one of the inputs of an operation is empty, we replace the operation by the non-empty pattern. If both patterns are empty, the parent will resort to an empty pattern itself and then let the next parent node deal with the problem. As this is a recursive mechanism. it does mean the result of a whole tree can be empty if the root node produces an empty pattern, in which case we discard the whole tree from the search. This procedure has two positive effects. First, the search space is reduced as useless sub-trees are not considered. Second, in the experiments this has lead to the prevention of so-called bloat [5, page 191] in the evolutionary algorithm, which is described later.

Worth noting is that the operation NOT cannot be included in the operations due to the nature of the patterns. A gene expression pattern is the result of a stain observed through a curation process. If we would negate the pattern, i.e., take the whole embryo and subtract the pattern, we then explicitly assert that no gene expression occurs in the negated pattern. This assertion is false on many



(a) Target pattern (EMAGE:2903) consists of the conjunction of strong and moderate expression of the gene Dmbx1 in the forebrain


AND
ts15_to_ts19_sm_01/EMAGE:714
Nefm
OR
ts15_to_ts19_sm_01/EMAGE:1411
Otx2
OR
ts15_to_ts19_sm_01/EMAGE:1083
Sim2
ts15_to_ts19_sm_01/EMAGE:2568
Hook3

(b) Evolved interaction tree, that tries to match the target pattern in Figure 3(a) and conforms to the grammar: (EMAGE:714 AND (EMAGE:1411 OR (EMAGE:1083 OR EMAGE:2568))); the similarity to the target is equal to 0.804

Fig. 3. An example of an interaction tree and a target. The yellow parts (bright) represent strong and moderate expression of the corresponding genes in the model embryo, represented in red (darker).

grounds; the curator may have seen only part of the embryo and the curator also notes weak and possible expression, which is not considered in the patterns we use here.

3.3 Matching Patterns

A function is required to compute the quality of a match between two patterns. This will allow the evolutionary algorithm described next to direct its search toward better matches. Given two patterns p_1 and p_2, we measure their similarity using the Jaccard Index [6]:

$$\text{similarity}(p_1, p_2) = \frac{\text{area}(p_1 \cap p_2)}{\text{area}(p_1 \cup p_2)}, \tag{1}$$

In Figure 3(a), a target pattern is given. The similarity of this pattern with the result from the evolved interaction tree, i.e., the pattern in the root of the tree, shown in Figure 3(b) is equal to 0.804.

The Jaccard Index was used in two previous studies on hierarchical clustering and association rules mining in which it gave the best results. It will be used also as a measure of quality of success in the experiments.

3.4 An Evolutionary Algorithm to Search for Sentences

The evolutionary algorithm [7] operates on the representation defined in Section 3.2, i.e., a binary tree where each internal node is one of the three binary operators defined over images and each leaf is a pattern takes from a predefined set of patterns.

Initially one hundred trees are randomly generated by growing them randomly [8]. A stochastic process is used to determine at each decision point whether a given node becomes either a leaf node, i.e., a pattern, or an internal node, i.e., an operation. If it becomes an operation we repeat this process for all the children of that node. The stochastic process makes a node a leaf with the probability of $1/(1+\text{depth of the node in the tree})$ and an internal node otherwise. The actual choice of operation or pattern is a random uniform selection. Important to note is that the same mechanism is used to create random trees that serve to provide the target patterns in the experiments.

To create new individuals, two genetic operators are used. A crossover which picks one node (which can be a leaf) uniform randomly in two distinct trees and then replaces the sub-tree of the node pointed at in the second tree with the sub-tree of the node pointed at in the first tree. The result is the new offspring, which then undergoes mutation by again selecting a node (which can be a leaf) and then replacing its sub-tree with a randomly generated sub-tree. Trees that exceed 200 nodes and leafs are discarded, although this has not occurred in the experiments.

The objective function is the similarity function (Equation 1), which needs to be maximised. The offspring will always replace the tree that represents the pattern with the worst match, i.e., lowest fitness, in the population.

The algorithm terminates when the mean fitness of the population has converged to a preset value. More specifically, if μ is the mean fitness of the population and σ is the standard deviation of the population, then the algorithm terminates if $\frac{\mu}{\mu+\sigma} \geq 0.85$. To ensure timely termination, a maximum is set of 5 000 evaluations. The algorithm also terminates if a perfect match with the target pattern is found, i.e., if the optimisation function is equal to 1.0.

4 Experiments and Results

We perform two experiments. The first experiment will be used to determine the robustness of the method. More specifically, it will be used to determine the number of runs required of the evolutionary algorithm to get a reliable result. The second experiment will take patterns from the gene expression database and use these as targets for the methodology, after which the significance of the interaction trees is validated using an overrepresentation analysis.

4.1 Robustness of the Methodology

To evaluate the robustness of the approach we devise an experiment whereby target patterns are randomly perturbed. We provide the perturbed pattern to the evolutionary algorithm and measure how well it is able to match the original pattern. Both the amount of perturbation and how well the evolutionary algorithm matches the original pattern are measured in the same way as the objective function of the evolutionary algorithm (see Equation 1).

The following procedure is repeated 261 times. We create a random tree in the same manner as described in the initialisation phase of the evolutionary algorithm. The resulting pattern of this tree forms the *original target pattern*. Each pixel in the original target pattern undergoes a translation using a uniform random distribution over a domain of -20 and $+20$ in both x and y directions; the size of the bounding box of the embryo domain is 267×258. This process yields a *perturbed pattern*. The evolutionary algorithm is then run sixty times[2] with a unique seed to its random generator with as its target pattern the perturbed pattern. The best solution of one run of the evolutionary algorithm is called the *output pattern*. On average the size a the tree creating the original target pattern consists of 6.30 nodes (with a standard deviation 3.90).

We measure the amount of perturbation of the original target pattern with the perturbed pattern using the similarity defined in Equation 1. We measure the success of a run of the evolutionary algorithm by applying the same function to the output pattern of the run with the original target pattern. If the similarity between these two patterns is 1.000 we then check if the evolved tree is equal to the tree that created the original target pattern.

Figure 4 shows the average success of the evolutionary algorithm in matching the original target pattern. Each point is the average of sixty unique runs and

[2] One run of the algorithm takes about six minutes on a 2.33Ghz Intel Core Duo.

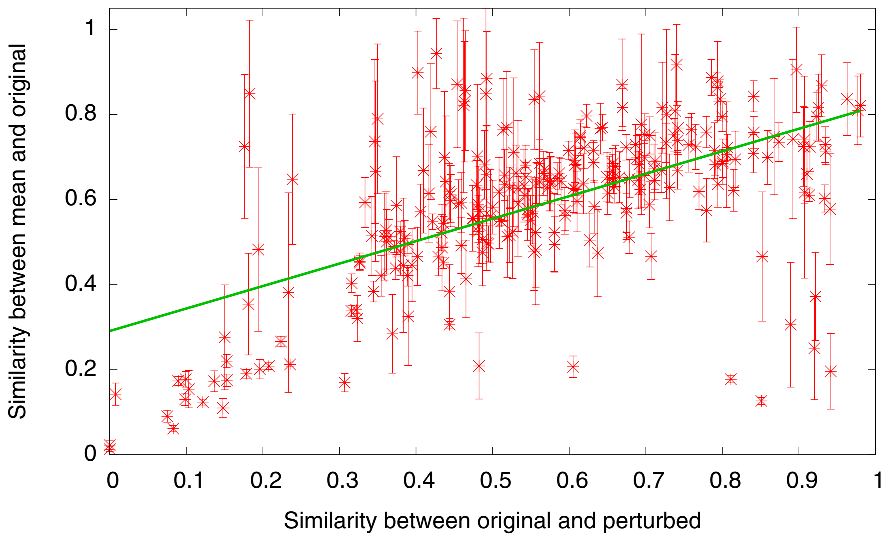


Fig. 4. Average amount of perturbation (*x-axis*) to the average success of matching the original pattern (*y-axis*). Every point is averaged over 60 unique runs of the evolutionary algorithm with 95% confidence intervals included. A linear regression is provided over the averaged points.

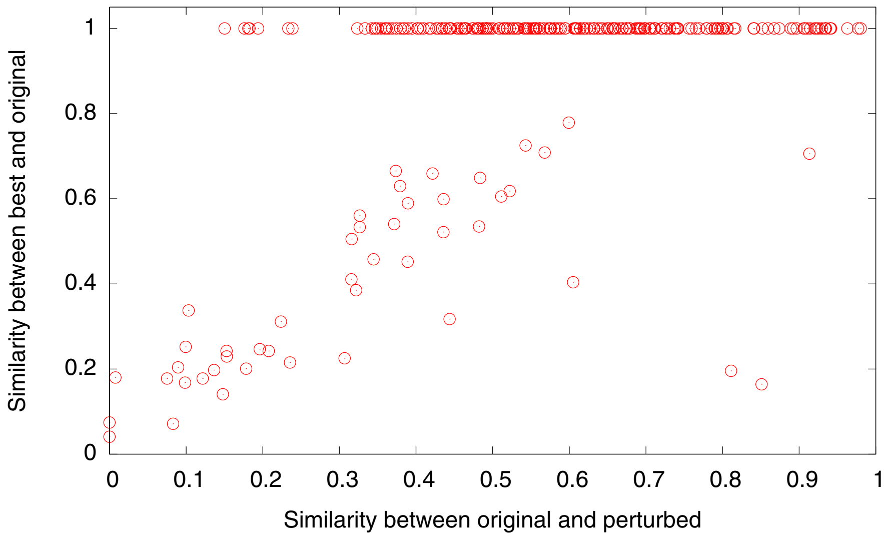


Fig. 5. Every point is the best match found (*y-axis*) in 60 unique runs of the evolutionary algorithm on one perturbed target pattern (*x-axis*)

is accompanied by a 95% confidence interval. A regression is performed with Marquardt-Levenberg's nonlinear least-squares algorithm [9, page 683] on the averages, which results in a regression of $y = 0.528x + 0.291$ where the correlation coefficient is $r = 0.593$, the rms of residuals is 0.156, and the variance of the residuals is 0.024.

Figure 5 shows the best match found in the sixty runs for each original target pattern generated. The match is calculated between the best result from the evolutionary algorithm on the perturbed target pattern and the original target pattern. A large set of the points, 82.8% correspond to a similarity match of 1.0. After examining the corresponding trees for each of these matches, we confirm the evolutionary algorithm was able to recreate all the original trees for these cases.

4.2 Matching Gene Expression Patterns from the Mouse Atlas

We take the set of 1970 gene expression patterns as introduced in Section 2. Every pattern belongs to a gene for which the expression has been mapped to a standard model embryo. We repeat the following operation for every pattern in the set. The pattern is temporarily removed from the total set and is deemed the target pattern. The evolutionary algorithm will then try to construct a pattern that matches it as close as possible by creating an interaction tree that only makes use of the remaining gene expression patterns. In other words, we are using mapped gene expression patterns from the Mouse Atlas itself as target patterns.

For each target pattern we run the evolutionary algorithm sixty times, as the results from the experiments in Section 4.1 show this leads to robust solutions. The total number of runs of the evolutionary algorithm in this experiment becomes 118 200. For the resulting interaction tree of each run, we determine whether the genes used in that interaction tree are statistically overrepresented with respect to groups of genes associated with annotations in the *Gene Ontology* (GO) [10]. Numerous software package support this type of statistical verification. Here we use the free and Open Source GO::TermFinder Perl module [11]. It works by calculating a p-value using the following hypergeometric distribution without re-sampling:

$$p\text{-value} = \sum_{j=x}^{n} \frac{\binom{M}{j}\binom{N-M}{n-j}}{\binom{N}{n}} \qquad (2)$$

In this equation, N is the total number of genes in the background distribution, which is all genes in our study and therefore equal to 1131, M is the number of genes within that distribution that are annotated (either directly or indirectly) to the node of interest in the Gene Ontology, n is the size of the list of genes of interest, i.e., in the tree interaction, and k is the number of genes within that list annotated to the node. To account for falsely finding significant hypothesis due to random chance given multiple events, we use Bonferroni correction. We deem results significant if $p < 0.05$.

The statistical analysis gives us a list of items, where each item consists of a target pattern that belongs to a gene, the resulting interaction tree of the corresponding run, a value to express the match between the target pattern and the pattern originating from the interaction tree as calculated using Equation 1, a p-value as calculated using Equation 2, a GO term G, and a list of genes that simultaneous appear in the interaction tree and are attributed to the GO term G. In addition to these results, we also include the size of the target pattern in relation to the total embryo. This helps us to discard patterns that take up almost all of the embryo, which tend to correspond to housekeeping genes.

Some disadvantages exist in using this methodology to validate statistical significance, as it depends fully on the current annotations of the Gene Ontology. It may prevent us from extracting new discoveries in the following ways. It may happen the evolutionary algorithm finds a set of genes that interact, but which are not associated with an annotated term in the Gene Ontology, or because the term simply does not exist. Another possibility is that the set of interaction genes is small and can be found by chance alone, in which case it will be discarded on the basis of the p-value. Also plausible is that genes cannot contribute to the likelihood that the interaction tree they are part of passes the significance test because although they do exist in the Mouse Atlas Database, they do not exist in the Gene Ontology,.

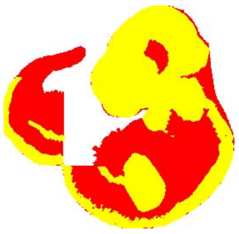
This overrepresentation analysis yields 6666 items from 1992 unique runs of the evolutionary algorithm. It is possible a set of genes is involved in multiple Gene Ontology terms. The total number of items is too large to include here or to ask a developmental biologist to poor over. We therefore filter the list of items by only including those where the result of the interaction tree matches with more than 0.70 similarity to the target pattern and the relative size of the target pattern is less than 0.50 of the embryo.

After filtering, we are left with 230 items, of which we present 45 filtering in Table 1. Each row in the table corresponds to the results of performing an overrepresentation analysis in the context of one GO term with the genes from one interaction tree that tries to match one gene expression pattern from the database. To illustrate the real output, we show the corresponding target pattern and the interaction tree of two of these results, which are shown in Figure 6 and Figure 7.

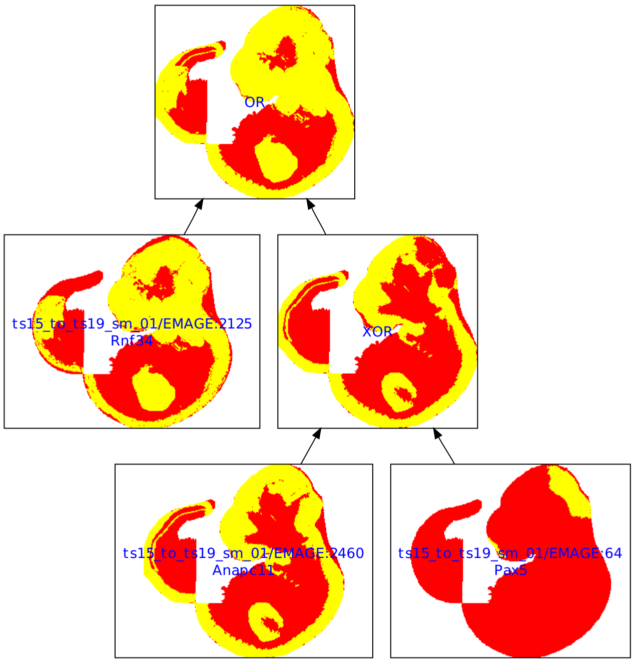
In Figure 7(b), the interaction tree shows all three types of interactions. First the intersection of the patterns of Hoxa9 and Gli3 is taken (AND), after which the result undergoes a XOR with the pattern of Otx2. Last, the result of that interaction is then merged (OR) with the pattern of the gene Lsr to form the result of the interaction tree. This result has a similarity of 0.722 when compared to the target gene expression pattern of Pygo1 shown in Figure 7(a) using Equation 1. In [12], the interaction between the genes Otx2 and Gli3 is described in the context of the development of the inner ear.

Table 1. Items resulting from checking statistically overrepresentation of genes against annotated Gene Ontology terms. Every row consists of the target pattern as an EMAGE id, the run number, the relative size of the target pattern, the match between the target pattern and the result from the interaction tree, the *p*-value from the overrepresentation analysis, the genes corresponding to both the tree and the GO term, and the GO term where the analysis was performed against.

EMAGE	run	size	match	*p*-value	genes	GO term
1182	10	0.39	0.75	0.021	Ube2g2 Zfp36l2	cellular macromolecule catabolic process
1182	34	0.39	0.74	0.00066	Shh Pecam1	lipid raft
1182	34	0.39	0.74	0.0027	Ube2g2 Shh	proteasomal ubiquitin-dependent protein catabolic process
1182	34	0.39	0.74	0.027	Ube2g2 Shh	cellular protein catabolic process
1182	34	0.39	0.74	0.027	Ube2g2 Shh	modification-dependent macromolecule catabolic process
1182	34	0.39	0.74	0.027	Ube2g2 Shh	modification-dependent protein catabolic process
1182	34	0.39	0.74	0.027	Ube2g2 Shh	proteolysis involved in cellular protein catabolic process
1182	34	0.39	0.74	0.027	Ube2g2 Shh	ubiquitin-dependent protein catabolic process
1204	8	0.0049	0.72	0.0095	Lpp Cxadr Bcl6	biological adhesion
1204	8	0.0049	0.72	0.0095	Lpp Cxadr Bcl6	cell adhesion
1205	2	0.0041	0.74	0.029	Nkx2-2 Nkx6-2	pancreas development
1224	3	0.0059	0.71	0.007	Rnf6 Rnf14 Hlcs Pias2	ligase activity, forming carbon-nitrogen bonds
1224	3	0.0059	0.71	0.046	Rnf130 Rnf6 Mbtps1	proteolysis
130	18	0.024	0.73	0.0026	Actc1 Shh Nkx2-5 Bmp4	muscle cell differentiation
130	18	0.024	0.73	0.0029	Actc1 Shh Bmp4	myoblast differentiation
130	18	0.024	0.73	0.0081	Actc1 Shh Nkx2-5	striated muscle cell differentiation
130	18	0.024	0.73	0.012	Actc1 Shh Bmp4	muscle fiber development
130	18	0.024	0.73	0.012	Actc1 Shh Bmp4	skeletal muscle fiber development
130	18	0.024	0.73	0.013	Actc1 Shh Nkx2-5 Bmp4	striated muscle development
130	18	0.024	0.73	0.017	Nkx3-1 Shh	prostate gland development
130	18	0.024	0.73	0.017	Shh Bmp4	telencephalon regionalization
130	4	0.024	0.74	0.00013	Actc1 Nkx2-5 Bmp4	muscle cell differentiation
130	4	0.024	0.74	0.00045	Actc1 Nkx2-5 Bmp4	striated muscle development
130	4	0.024	0.74	0.0015	Actc1 Nkx2-5 Bmp4	muscle development
130	4	0.024	0.74	0.0033	Actc1 Bmp4	myoblast differentiation
130	4	0.024	0.74	0.0062	Actc1 Nkx2-5	striated muscle cell differentiation
130	4	0.024	0.74	0.008	Actc1 Bmp4	muscle fiber development
130	4	0.024	0.74	0.008	Actc1 Bmp4	skeletal muscle fiber development
130	4	0.024	0.74	0.023	Actc1 Bmp4	skeletal muscle development
130	7	0.024	0.73	0.02	Csrp3 Nkx2-5	cardiac muscle development
130	7	0.024	0.73	0.021	Actc1 Csrp3	I band
130	7	0.024	0.73	0.021	Actc1 Csrp3	contractile fiber
130	7	0.024	0.73	0.021	Actc1 Csrp3	myofibril
130	7	0.024	0.73	0.021	Actc1 Csrp3	sarcomere
1326	18	0.0049	0.71	0.047	Nr1i3 Nr2e3	steroid hormone receptor activity
1326	29	0.0049	0.74	0.021	Lpp Sox9 Tnxb Bcl6	biological adhesion
1326	29	0.0049	0.74	0.021	Lpp Sox9 Tnxb Bcl6	cell adhesion
1327	16	0.0059	0.74	0.033	Nefl Nbn	neuromuscular process controlling balance
1327	17	0.0059	0.76	0.025	Tnxb Zfp146	heparin binding
1327	17	0.0059	0.76	0.035	Tnxb Zfp146	carbohydrate binding
1327	17	0.0059	0.76	0.035	Tnxb Zfp146	glycosaminoglycan binding
1327	17	0.0059	0.76	0.035	Tnxb Zfp146	pattern binding
1327	17	0.0059	0.76	0.035	Tnxb Zfp146	polysaccharide binding
1327	29	0.0059	0.91	0.0041	Cdkn1c Mxi1 Nr2e3	negative regulation of transcription from RNA polymerase II promoter
1327	29	0.0059	0.91	0.009	Cdkn1c Mxi1 Nr2e3	negative regulation of transcription, DNA-dependent

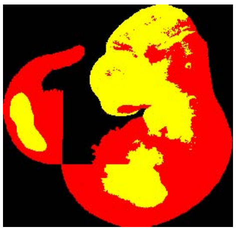


(a) Target pattern (EMAGE:2188) consists of the conjunction of strong and moderate expression of the gene Mid1

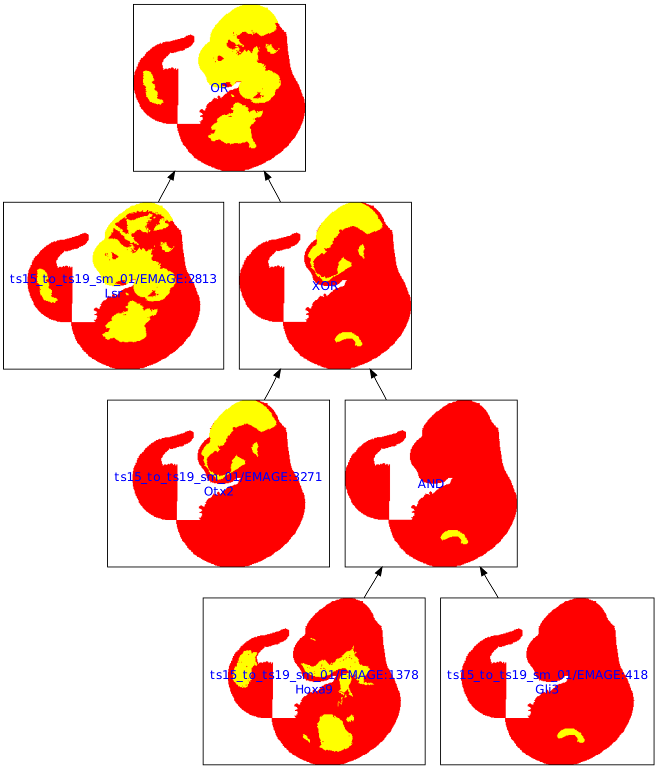

OR
ts15_to_ts19_sm_01/EMAGE:2125
Rnf34
XOR
ts15_to_ts19_sm_01/EMAGE:2460
Anapc11
ts15_to_ts19_sm_01/EMAGE:64
Pax5

(b) Evolved interaction tree, that tries to match the target pattern in Figure 6(a) and conforms to the grammar: (EMAGE:2125 OR (EMAGE:2460 XOR EMAGE:64)); the similarity to the target is equal to 0.753

Fig. 6. An example of an interaction tree and a target. The yellow parts (bright) represent strong and moderate expression of the corresponding genes in the model embryo, represented in red (darker). The genes Anapc11, Rnf34 and Pax5 are overrepresented with respect to the GO terms *biopolymer modification*, *cellular macromolecule metabolic process*, *cellular protein metabolic process*, *post-translational protein modification* and *protein modification process* with p-values 0.0069, 0.021, 0.021, 0.0057 and 0.0064, respectively.



(a) Target pattern (EMAGE:2625) consists of the conjunction of strong and moderate expression of the gene Pygo1


OR
ts15_to_ts19_sm_01/EMAGE:2813
Lsr
XOR
ts15_to_ts19_sm_01/EMAGE:3271
Otx2
AND
ts15_to_ts19_sm_01/EMAGE:1378
Hoxa9
ts15_to_ts19_sm_01/EMAGE:418
Gli3

(b) Evolved interaction tree, that tries to match the target pattern in Figure 7(a) and conforms to the grammar: (EMAGE:2813 OR (EMAGE:3271 XOR (EMAGE:1378 AND EMAGE:418))); the similarity to the target is equal to 0.722

Fig. 7. An example of an interaction tree and a target. The yellow parts (bright) represent strong and moderate expression of the corresponding genes in the model embryo, represented in red (darker). The genes Otx2 and Gli3 are overrepresented with respect to the GO term *cell fate specification* with p-value $= 0.046$.

5 Discussion

With the increase of resolution and speed in which *in-situ* data can be captured [13,14], these type of studies become more useful in a similar manner to how microarray studies have become mainstream techniques. Where microarray gives insight into many genes at once, *in-situ* studies have the benefit of high precision in terms of spatial context. To make the data that result from from this technique more digestible to developmental biologists, we need mechanisms that allow investigation of multiple gene interactions within a spatio-temporal context. In this paper, we have identified, designed, implemented and evaluated one such mechanism in the spatial context a developing mouse embryo.

The approach uses a grammar to define a search space that allows several types of spatial gene interaction patterns to be combined. An evolutionary algorithm is used to search this space with the aim of maximising the match with a given target pattern. This target pattern can be created by a human, or represent a particular space of an embryo, such as anatomical components, the spatial expression pattern of one gene or even the combination of expression patterns of a number of genes. The output consists of interaction trees that show how a set of spatial expression patterns of multiple genes should be combined using either the conjunction, disjunction or exclusive OR operations over these patterns.

The domain of *in-situ* hybridization studies potentially contains much noise and uncertainties. To validate whether our approach will cope within such an environment we generated gene interactions that then represent target patterns. These patterns were then slightly perturbed to form a test suite that allowed us to analyse the robustness of the system. The results show that running the evolutionary algorithm sixty times and then selecting the best solution from those runs leads to sufficiently matching solutions. The results from the second experiment show how interaction trees can be evolved to match gene expression patterns in the same database. By using an overrepresentation analysis against annotated genes that correspond to terms in the Gene Ontology, we were able to show the method is able to extract statistically significant gene interactions.

Our future goal is to provide an interface freely accessible via any web browser to allow biologists to define target patterns that allow them to perform the combinatorial search introduced in this study.

Acknowledgements

This study is part of an EU-funded Design Study "Developmental Gene Expression Map" http://www.dgemap.org/, led by Susan Lindsay, which we thank for the fruitful discussions about spatial gene expression patterns. We acknowledge Bill Hill for his support in usage of the Woolz [15] libraries, wherein these data are made accessible, as well as the editorial team of the Edinburgh Mouse Atlas Project, Jeff Christiansen, Lorna Richardson and Shanmugasundaram Venkataraman, for curating the data from the Gray et al. [2] study. This work has made use of the resources provided by the Edinburgh Compute and Data Faclity (ECDF) http://www.ecdf.ed.ac.uk/.

References

1. Christiansen, J., Yang, Y., Venkataraman, S., Richardson, L., Stevenson, P., Burton, N., Baldock, R., Davidson, D.: Emage: a spatial database of gene expression patterns during mouse embryo development. Nucleic Acids Research 34, 637–641 (2006)
2. Gray, P., et al.: Mouse brain organization revealed through direct genome-scale tf expression analysis. Science 306(5705), 2255–2257 (2004)
3. Theiler, K.: The House Mouse Atlas of Embryonic Development. Springer, New York (1989)
4. Knuth, D.: Backus normal form vs. backus naur form. Communications of the ACM 7(12), 735–736 (1964)
5. Langdon, W.B., Poli, R.: Foundations of Genetic Programming. Springer, Heidelberg (2002)
6. Jaccard, P.: The distribution of flora in the alpine zone. The New Phytologist 11(2), 37–50 (1912)
7. Bäck, T., Fogel, D., Michalewicz, Z. (eds.): Handbook of Evolutionary Computation. Institute of Physics Publishing Ltd, Bristol and Oxford University Press, New York (1997)
8. Koza, J.: Genetic Programming: On the Programming of Computer by Means of Natural Selection. MIT Press, Cambridge (1992)
9. Press, W., Flannery, B., Teukolsky, S., Vetterling, W.: Numerical Recipes in C: The Art of Scientific Computing, 2nd edn. Cambridge University Press, Cambridge (1992)
10. The Gene Ontology Consortium: tool for the unification of biology. Nature Genet. 25, 25–29 (2000)
11. Boyle, E.I., Weng, S., Gollub, J., Jin, H., Botstein, D., Cherry, J.M., Sherlock, G.: GO:TermFinder—open source software for accessing Gene Ontology information and finding significantly enriched Gene Ontology terms associated with a list of genes. Bioinformatics 20(18), 3710–3715 (2004)
12. Bok, J., Dolson, D.K., Hill, P., Ruther, U., Epstein, D.J., Wu, D.K.: Opposing gradients of Gli repressor and activators mediate Shh signaling along the dorsoventral axis of the inner ear. Development 134(9), 1713–1722 (2007)
13. Hunt-Newbury, R., Viveiros, R., Johnsen, R., Mah, A., Anastas, D., Fang, L., Halfnight, E., Lee, D., Lin, J., Lorch, A., McKay, S., Okada, H.M., Pan, J., Schulz, A.K., Tu, D., Wong, K., Zhao, Z., Alexeyenko, A., Burglin, T., Sonnhammer, E., Schnabel, R., Jones, S.J., Marra, M.A., Baillie, D.L., Moerman, D.G.: High-throughput in vivo analysis of gene expression in caenorhabditis elegans. PLoS Biology 5(9), e237 (2007)
14. Lein, E.S., Zhao, X., Gage, F.H.: Defining a Molecular Atlas of the Hippocampus Using DNA Microarrays and High-Throughput In Situ Hybridization. J. Neurosci. 24(15), 3879–3889 (2004)
15. Piper, J., Rutovitz, D.: Data structures for image processing in a C language and Unix environment. Pattern Recognition Letters 3, 119–129 (1985)

Combining Molecular and Physiological Data of Complex Disorders

Emanuel Schwarz[1], F.Markus Leweke[3], Sabine Bahn[1,⋆], and Pietro Liò[2,⋆]

[1] Institute of Biotechnology, University of Cambridge, Tennis Court Road, Cambridge, CB2 1QT, UK,
[2] Computer Laboratory, University of Cambridge, 15 J.J. Thompson Avenue, Cambridge, CB30FD, UK
[3] Department of Psychiatry and Psychotherapy, University of Cologne, 50924 Cologne, Germany

Abstract. The accurate diagnosis of complex disorders is the prerequisite of appropriate and effective treatment. Ideally, the diagnostic process should include the evaluation of molecular and clinic tests alongside medical observations; these clinical observations and laboratory outcomes are usually integrated by an expert physician or one prevails and the other is used for mere confirmation. In clinical bioinformatics, complex datasets are investigated with the aim to improve the clinical management of diseases. The integration of data from disparate data sources is urgently needed and is the prerequisite in the implementation of network medicine which views disease relevant properties as networks and tries to untangle the information content in these networks. Here we developed a graph theoretical framework for combining and untangling the relationships of physiological and molecular data. We then applied the methodology to determine disease related abnormalities of a molecular network derived from serum of patients with schizophrenia and affective disorder. The universality of the concept is demonstrated by an integration of the metabolic data with a standard laboratory test of glucose measurements. The generation of compound networks allows the integrated analysis of disease relevant information and the detection of robust patterns ultimately facilitating the description of complex disorders. This approach could lead to an automatized methodology for improving disease classification and diagnosis.

1 Introduction

The correct identification of a medical condition is central in clinical practice. The diagnostic process is the evaluation of a patient's display of clinical symptoms that are more or less specific for a certain disease. While some diseases are clearly classifiable purely on the basis of the displayed symptoms, others are difficult to identify unambiguously or discriminate from confounding factors.

⋆ Corresponding authors.

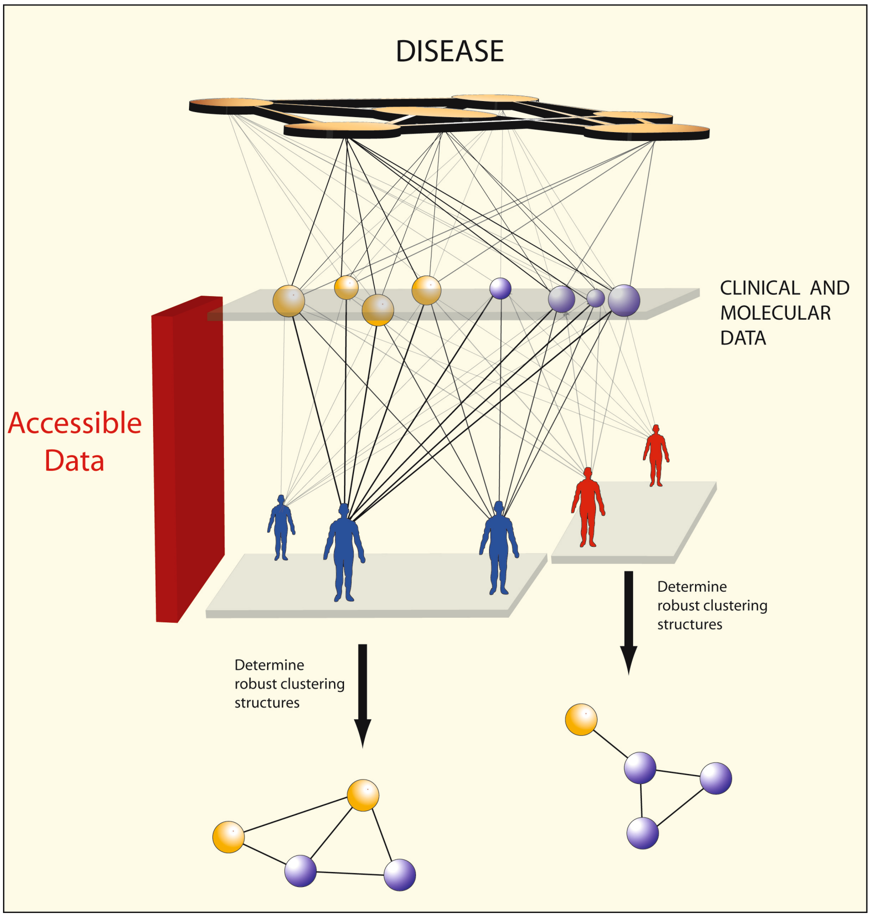

DISEASE
CLINICAL AND MOLECULAR DATA
Accessible Data
Determine robust clustering structures
Determine robust clustering structures

Fig. 1. We show a schematization of the approach to detect robust disease patterns in complex disorders. The upper layer shows the connectivity between disorders based on partially overlapping effects on clinical and metabolic measurements. The middle and lower layer show the relationship between disease phenotypes and clinical or molecular abnormalities, a directed network in which every patient is connected to the respective clinical or molecular abnormality. This information is used to determine robust disease patterns.

Due to their inherent heterogeneity, psychiatric disorders are a good example illustrating the problem of correct diagnosis. Symptoms often overlap with other neuropsychiatric or neurodegenerative disorders and other conditions such as drug or alcohol abuse. Therefore, relying on the display of symptoms may be insufficient as diagnostic criteria are often based on mere clinical observations defined decades ago and might not reflect the biochemical underpinning of a given disease. Extensive research efforts have thus been made to develop diagnostic tools based on biochemical alterations to facilitate objective diagnosis.

For some diseases, advanced statistical methods are already in place to facilitate automatic classification based on molecular readouts. In medical data repositories, phenotype as well as molecular data are now available for many disorders, turning the diagnostic process into a multiscale problem. While the identification of medical conditions based on only the physiological information may be inaccurate and subjective, relying on molecular information alone might result in the loss of valuable information. Therefore, a unifying concept is needed to combine molecular and physiological information to capture diverse hallmarks of complex disorders and facilitate better diagnostic approaches. Here, we present a novel approach based on graph theory to integrate quantitative as well as qualitative information gathered on a given disorder. We use complex networks to describe associations between these pieces of information and point to statistical measurements that can be used to investigate the resulting networks (Figure 1).

Clinical evaluation of medical conditions is based on the display of different symptoms. Therefore, symptoms can be represented as a network in which they are connected to each other when displayed simultaneously. The same concept applies for molecular alterations of a certain disease where molecules are connected in a network when altered in the same medical condition. Symptoms and molecular alterations can thus be represented in a directed graph in which each symptom is connected to the respective molecular alterations.

Here, we introduce a graph theoretical approach to combine molecular and clinical data as a novel concept to improve clinical diagnosis. In the next paragraph we will describe an approach to generate networks from the data and illustrate how to determine robust and relevant patterns within networks. Then we will present one case study exemplifying the application of the methodology on metabolic profiles of a large sample cohort of human serum and its integration with a standard laboratory test. Finally we will discuss the generalization of this concept in the broader context of molecular medicine.

2 State of the Art Clinical Bioinformatics

The investigation of complex disorders will take advantage from advances in the field of Clinical Bioinformatics [1] which is concerned with bringing together bioinformaticians and biostatisticians to develop methods and tools for analysis and visualisation of complex datasets. In contrast to the classical bioinformatics field which focuses on the analysis of molecular biological information (See [2,3,4,5] for an introduction to Bioinformatics), here the main focus is to collate heterogeneous data sets from disparate data sources (e.g. patient clinical records, proteomics and transcriptomics data, e.g. the analysis of translational patterns, and pathomorphology, i.e. readouts for cancer patterns) and develop novel algorithms for the analysis of heterogeneous data. Thus, the key goal is the simultaneous evaluation of clinical and basic research data with the aim to improve medical care and health science achievements (See [6] for data exploitation methods in cancer therapy development). It is noteworthy that the field of systems biology overlaps with these aims. Again, the challenge is the integration and interpretation of biological data generated by a range of technologies. Currently, the major focus of systems biology is the investigation of the inter- and intra-cellular networks through mathematical modelling and simulation, particularly focusing on the dynamic characteristics of biological systems [7,8,9,10]. Biologists represent biochemical and gene networks as state transition diagrams with rates on transition. This approach provides an intuitive and qualitative understanding of the gene/protein interaction networks underlying basic cell functions through a graphical and database-oriented representation of the models. More mathematical approaches focus on modelling the relationships between biological parameters using a connectivity network, where nodes are molecules and edges represent weighted ontological/evolutionary connections [16,13,11,14,15,12]. Network properties have been elucidated for several biological processes, for example metabolic pathways [23,24], signal transduction [25,26,27,28], transcriptional regulation [29] and other cellular processes [30,31]. In order to visualize the influence of connectivity on network topology, we would like to introduce the connectivity distribution $P(k)$, which is proportional to the number of sites with connectivity k. The probability distribution of a regular lattice is extremely peaked around the lattice's *coordination number* (the number of vertices directly connected in the lattice), being a δ-function. $P(k)$ of a random graph follows a Poisson distribution, centred around the mean connectivity $\langle k \rangle$. Both of those choices are characterized by a well defined value of the mean connectivity $\langle k \rangle$, and small variance $\langle k^2 \rangle - \langle k \rangle^2$. A meaningful measure is the degree of *clustering* of the network, which can be defined as the number of neighbours of a given node which share an edge. In a regular lattice the clustering coefficient is high, while in a random graph it is vanishingly low. Finally, one is interested in the average length of the minimum path that connects any two vertices, called the *diameter* of the graph. This is large in a regular lattice, and low in a random graph. The study of biological networks has shown major differences from regular and random graphs. The degree distribution often exhibits a power-law (i.e. $P(k) \simeq k^{-\gamma}$, with $\gamma \simeq 2$). This distribution is characterized by a relatively large number of highly connected hubs. Moreover, such distributions have a diverging second moment $\langle k^2 \rangle$ for $\gamma \leq 3$ and a diverging average connectivity $\langle k \rangle$ for $\gamma \leq 2$. An important class of biological networks show scale-free connectivity properties [22]. A simple model to generate networks with scale-free properties is based on the *preferential attachment.* A major challenge in analyzing complex networks is to identify communities of genes that share features and similarities. Clustering algorithms in complex network data allow classifying ensembles of nodes and untangling the information content of the networks of relationships.

3 Methods and Algorithm

Advanced clustering techniques have been implemented to better identify clusters of related proteins from local similarity searches; tribeMCL [19] implements a relatively recent procedure, the Markov Clustering Algorithm (MCL, [20]) and was introduced to partition proteins into families. The main advantage of the

MCL over other clustering methods is that there is no need to specify the number of clusters.

1. Generation of the homology network: The networks generated in this study were based on n x m matrices containing the measurement information of m different molecular readouts from n individuals. From this matrix a directed graph with n + m nodes was created. Edges were introduced between individual n_i and molecule m_j if its measurement value was outside two standard deviations of the control mean. From the resulting network, it is possible to build two graphs, one that connects all molecules that are simultaneously changed in the same patient and a second graph that connects all patients that share molecular alterations. Here, we focus on the latter type. The links of these graphs are weighted by the frequency of two connected molecules being altered in the same patient. The frequency of randomly co-occurring changes was estimated by a large number of random networks. These were generated by the above mentioned method starting from random matrices. These were drawn from a normal distribution and were of the same size and average variability as compared to the empirical matrix. The frequency of random co-occurrences were subtracted from the empirical frequencies increasing the computational speed due to the higher sparsity of the network.
2. Clustering of the homology network: we use a clustering algorithm that does not need information on the number of clusters which is often unknown in large–scale protein comparisons. Although there is now a wide range of clustering algorithms, only a restricted number can successfully handle a network with the complete and weighted graph properties. Among them, we cite the recent method proposed by [35] that is based on simulated annealing to obtain clustering by direct maximization of the modularity. The modularity has been introduced by [16] and it is a measure of the difference between the number of links inside a given module and the expected value for a randomized graph of the same size and degree distribution. The modularity Q of a partition of a network is defined as $Q = \sum_s \left[\frac{l_s}{L} - \left(\frac{d_s}{2L}\right)\right]$ where the sum is over all modules of the partition. l_s and d_s describe the number of links and the total degree of the nodes inside module s and L the total number of links of the network [36]. In a recent work on resolution limits in community detection [36] the authors give evidence that modularity optimization may fail to identify modules smaller than a certain scale, depending on the total number of links in the network and on the number of connections between the clusters. Because of its properties, at the end, we implemented the Markov Clustering Algorithm (MCL, [20]). Its input is a stochastic matrix where each element is the probability of a transition between adjacent nodes. To increase the computational speed of the community detection, a nuisance parameter was used to assign an exponential weight to the transition probabilities. The weights between m_i and m_j were given by $e^{(\beta \cdot \omega_{ij})}$ where β is the nuisance parameter and ω_{ij} is the frequency of molecules m_i and m_j being simultaneously altered in the same patient. β was always set to 5. We modified the Java version of the MCL algorithm [37] to include the strategy

of Gfeller et al., [38] which allow detecting unstable nodes and compare results obtained with different granularity (inflation) parameters. In this algorithm, the starting matrix is modified to produce a novel matrix with a certain amount of noise added. The noise is homogeneously distributed between $-\sigma w_{ij}$ and σw_{ij} where w_{ij} is the edge weight and σ a fixed noise parameter, $0 \leq \sigma \leq 1$. The noise is added randomly to edges and the MCL clustering is performed on many noisy realization of the matrix. At each 'noisy' repetition, the algorithm records all the nodes belonging to the same cluster. After the prefixed number of repetitions has been concluded we end up with a matrix storing P_{ij} values corresponding to the fraction of times nodes i and j have been grouped together. Unstable nodes are identified as those having edges with less than a fixed values θ. We then calculate several distinct measures informing on the clustering and its stability such as the following clustering entropy: $S = -1/L \sum_{ij} [P_{ij} log_2 P_{ij} + (1 - P_{ij}) log_2 (1 - P_{ij})]$, where the sum is over all edges and the entropy is normalized by the total number of edges, L [39]. This might be used to detect the best clustering obtained after a long series of clusterings with different granularity parameters each time.

The entropy can be used to study the stability of communities obtained from the clustering procedures. Due to the repeated noisy realisations of the original matrix, nodes may be attached to different communities after the clustering procedure. However, if the investigated system is very stable, nodes tend to cluster with the same communities regardless of the added noise. The stability of the different communities can be investigated by analysing the entropy as a function of the clustering parameter r as the network breaks down into increasingly separated clusters as r increases.

In alternative to the MCL it is possible to use the Super Paramagnetic Clustering [40] that implements a method used for Ising systems in theoretical physics (http://mips.gsf.de/proj/spc).

3.1 Data Used in This Study

Metabolites are the last element in the gene-protein-metabolite cascade and are closely associated with an organism's phenotype. They are often considered as the downstream readouts of regulatory and metabolic processes. The metabolome, defining the entire system of metabolites, is highly complex and dynamic in nature which accounts for its quick response to changes in the state of an organism. Thus, it lends itself especially for the investigation of disorders that are highly influenced by environmental factors. In this study, we performed metabolic profiling of serum from 312 individuals. The ethical committees of the Medical Faculty of the University of Cologne approved the protocols of this study. Informed consent was given in writing by all participants and clinical investigations were conducted according to the principles expressed in the Declaration of Helsinki. The generated dataset contained molecular information about metabolite levels as well as clinical information about the disease state of the investigated individuals. We assessed glucose concentrations in the serum and

CSF of all individuals and integrated the information with mass spectrometric data.

4 Results

We profiled serum metabolites of 312 individuals using a Liquid Chromatography Time of Flight (LCT, Waters, Milford MA) mass spectrometer. Raw data were processed using MarkerLynx V4.1 (Waters, Milford MA) and exported to the freely available software package R. Here, the number of variables (detected peaks corresponding to a certain number of metabolites) was reduced to 225 by a stringent filtering step that retained only those molecules that were measured in every sample. The sample collection included 70 samples taken from drug naive first onset schizophrenia patients, 37 samples from affective disorder patients, 59 control samples taken from healthy volunteers and other sample groups not considered in this manuscript. We then encoded the molecular and physiological information into a network. For every patient, molecules differing significantly from the control mean (p=0.05) were identified and connected to the respective patient. This procedure generated a directed graph with 312 patient nodes and 225 molecule nodes in which all patients were connected to their respective molecular abnormalities. Based on this information, a graph was generated that connected all molecules that were simultaneously changed in the same patient. As a second step, we performed clustering analysis to detect layers of relationships and communities composing the graph. For this purpose clustering procedures

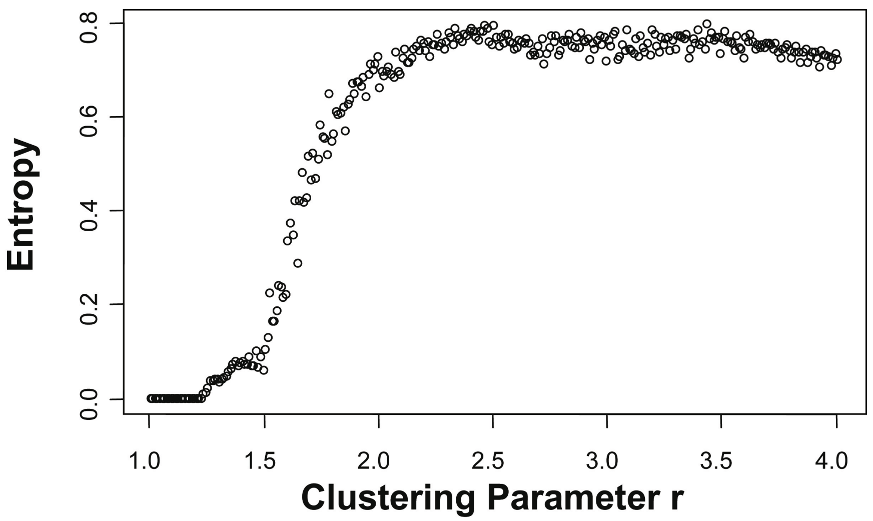


Fig. 2. Entropy of metabolic network created from all patient information throughout the clustering procedure. As the network is broken down into community structures, the entropy increases and reaches stable values at a clustering coefficient of approximately 2.

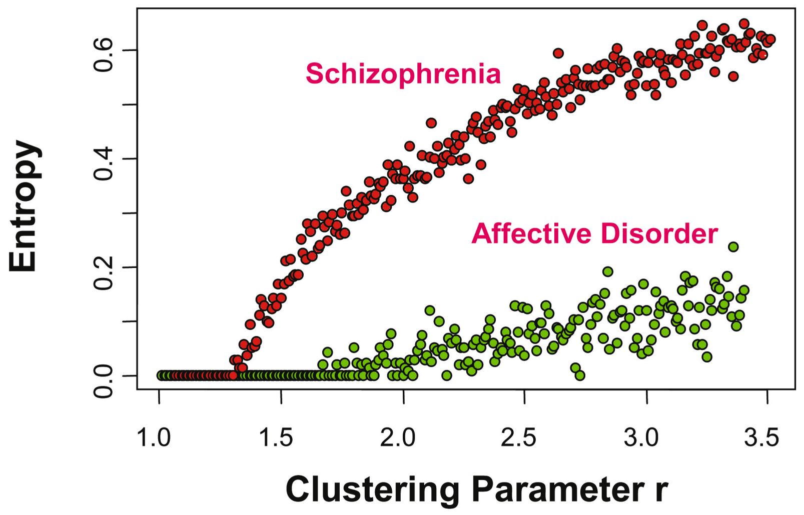


Fig. 3. Comparison of the stability of the primary fatty acid networks determined from schizophrenia and affective disorder patients. The stability is measured using the networks' entropy during the clustering procedure. The entropy is decreased at a lower rate in affective disorder, reflecting higher stability and a stronger alteration of the primary fatty acid network.

which don't require cluster number specification were used. An entropy measure was used to assess the stability of the detected communities when noise is applied during the clustering procedure. Figure 2 shows the entropy of the network of 225 molecules determined from all patients. The entropy increased with increasing clustering coefficients as the completely connected network was broken down into distinct community structures. The entropy was stable for clustering coefficients grater than two, implying that the detected communities were stable. At this clustering coefficient, the network derived from schizophrenia patients featured one prominent community structure consisting of 24 metabolic peaks. Based on exact mass and isotopic pattern, these molecules were manually identified to be primary fatty acid amides. The identity of two of these molecules (Oleamide and Linoleamide) was confirmed by re-analyzing commercially available standards that were infused in pure form or spiked into a serum sample. In a second replication of the study, similar results were obtained ruling out the influence of possible confounding factors such as plasticizers contained in the instrumental setup. Our analysis suggests that molecules with high connectivity might have a greater probability of being involved in connected physiological processes.

Figure 3 compares the stability of the network of 24 peaks between drug naive patients suffering from first onset, paranoid schizophrenia and affective disorder patients. The network was present for both disorders implying that significant abnormalities were detected for both patient groups. For schizophrenia patients, the entropy increased at a lower clustering coefficient and followed a log-linear shape. For affective disorder patients, the network was more stable and split

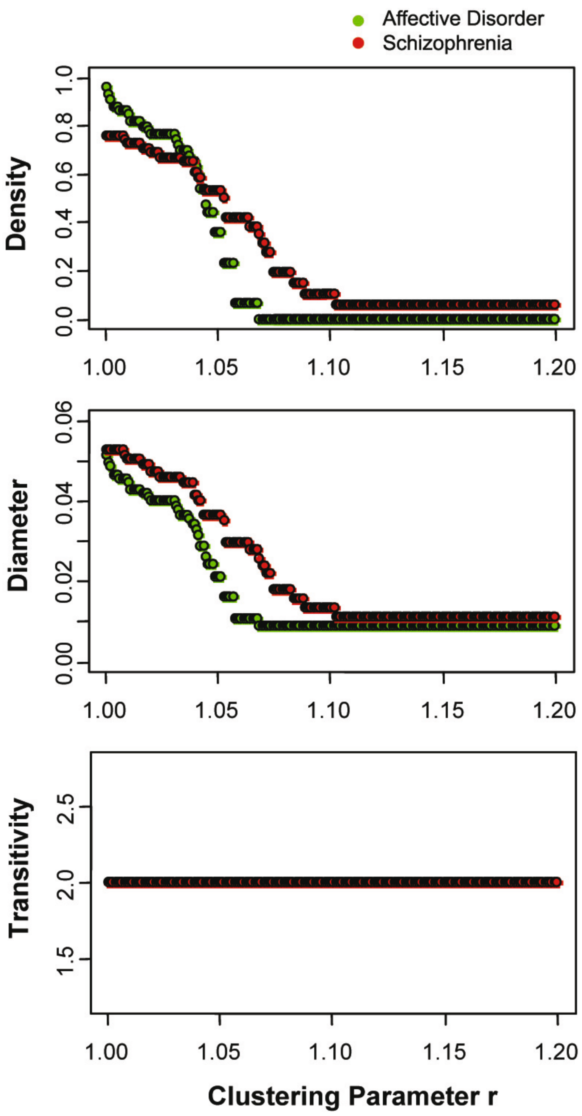


Fig. 4. Network properties during the clustering procedure. Graph density (top panel) and diameter (middle panel) decreased at a lower rate in schizophrenia and affective disorder. The transitivity is identical for both graphs and does not change during the clustering.

apart at a higher clustering coefficient. The shape of the entropy curve was linear for affective disorder patients and reached a far lower value than the network derived from schizophrenia patients. These results show that in affective disorder, the network of primary fatty acid amides was strongly connected and very stable due to a higher degree of alteration of the molecules in this patient group.

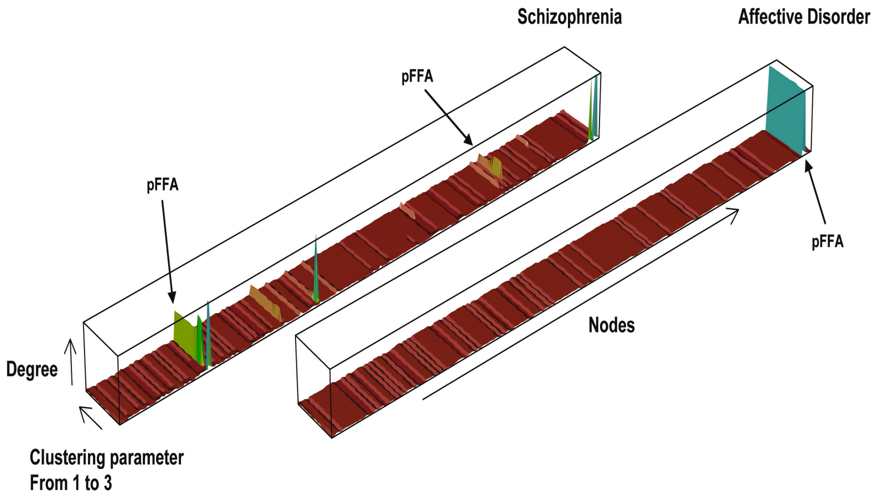


Fig. 5. Degree distribution during the clustering procedure in schizophrenia (left panel) and affective disorder. In both disorders, the community structures are centered around the primary fatty acid amide network. This effect is more pronounced in affective disorder.

Consistent with the finding that the network is more stable in affective disorder, at the same clustering coefficient (r=2), the network contained 24 nodes in schizophrenia, but 49 nodes in affective disorder after the clustering procedure. To exemplify the inclusion of clinical meta-data into this framework, we use a standard laboratory test of glucose levels in the cerebrospinal fluid (CSF) and serum of the patients. This information was added to the original matrix of metabolic measurements and the clustering procedure repeated. In schizophrenia, CSF glucose levels co-cluster with the metabolic network of primary fatty acid amides indicating their simultaneous involvement in the pathology. For affective disorder, both the CSF and serum glucose levels co-cluster with the network of 49 molecules.

Furthermore, we assessed properties of network topologies during the clustering procedure. Figure 4 shows graph density, diameter and transitivity as the clustering procedure continued and the network was broken down into community structures. Besides the high degree of similarities between networks derived from schizophrenia and affective disorder patients, the measured indicators decreased at a lower rate and reached higher equilibrium values in schizophrenia patients. This observation resulted from different stabilities as the schizophrenia graph decomposed into six clusters of connected molecules, whereas the affective disorder graph broke down into one stable cluster at r=2. Figure 5 shows the degree distribution during the clustering procedure for both disorders. It is apparent that in affective disorder, the degree distribution was dominated by one peak that refers to a molecule that is contained in the primary fatty acid networks in both disorders. In schizophrenia, the degree distribution was more complex but also centred around the primary fatty acid network.

5 Discussion

5.1 Robust Disease Patterns in Serum of Schizophrenia and Affective Disorder

We have performed an extensive analysis of robust metabolic patterns in the serum of schizophrenia and affective disorder patients. We were able to link the disease phenotype to its respective molecular abnormalities in a directed graph. Subsequently, a complete molecular network was created linking metabolites that were altered in the same patient. This complete graph is weighted by the degree of alteration in the disease state. It can be expected that robust and disease specific abnormalities are present in the majority of patients and more pronounced than randomly co-occurring alterations. To identify robust disease patterns, a technique is needed which allows the identification of robust patterns by breaking weaker connections between metabolites. For this purpose, we applied a clustering procedure and identified a network of primary fatty acid amides that was robustly altered in both disorders. We chose to use the Markov Clustering algorithm, but other clustering procedures are equally compatible with the framework presented here. An entropy measure that quantifies the stability of the networks was used to compare the disorders and revealed more stable graph topologies in affective disorder. We also exemplify several graph theoretical measures such as the degree distribution that can be used to assess the state of the network at any given time during the clustering procedure. Important nodes can thus be identified and their relation to other molecules investigated. For the serum metabolites investigated in this study, only molecules involved in the primary fatty acid amide network had relevant connectivities. One feature of the presented methodology that is of particular advantage is its extendibility to patient data of any nature. We exemplified this property by combining the metabolomics dataset with glucose measurements from the CSF and serum of all patients and controls. The generated network is thus a compound network that features molecular abnormalities and clinical meta-data at the same time. Robust and disease specific patterns can be detected as described above as the clustering procedure is invariant with respect to the origin or type of measurement associated with the disease phenotype. We showed that in schizophrenia, CSF glucose measurements co-cluster with the network of primary fatty acid amides; in affective disorder, glucose levels of both CSF and serum cluster with the metabolic network.

5.2 Towards Personalized Network Medicine

Networks are an intuitive concept of visualizing disease related information. Similar to social networks, relationships of any nature between individuals can be coded in a network. Networks can reflect familiar relationships, physical contacts or other interactions that may explain the spreading of a disease. Similarly, relationships between disorders can be illustrated in a network. Several lines of evidence point to the fact that many disorders have partially overlapping biochemical underpinnings. On the lowest level, interactions between molecules can be shown as networks which is already common practice. This concept of encoding relationships between individuals, diseases or molecules into networks is called network medicine [45]. Here, we extended this concept towards an integrated representation of complex disorders encoding all relevant and robust information simultaneously. This especially refers to clinical information about the disease state including important meta-data such as presence or absence of certain symptoms, rating scales, standard laboratory tests as well as molecular information such as a patient's genotype or molecular readouts from body fluids or tissues. Graph theoretical approaches are suited to capture the complexity of human diseases and provide a theoretical framework to easily incorporate molecular readouts and patient information to give a comprehensive description of a disease state. So far, these two types of data have been used as isolated pieces of information in clinical practice. Here, we describe an approach to generate networks combining molecular and clinical data and to detect robust patterns in these networks. The concept allows a fundamental extension of the connectivity between clinical and molecular data and facilitates the joint interpretation of the most relevant patterns.

Using a graph theoretical approach, the similarity of patients can be readily determined from the integrated patient information enabling the assessment of disease similarity and possibly, the subclassification of patients. This would be particularly desirable for psychiatric disorders for which the highly heterogeneous symptoms may result from different etiologies and possibly contribute to the low efficacy of current drug regimes. Extending the concept of subclassifying patient cohorts to the single patient level leads to a conceptual framework often referred to as personalized medicine. Patient specific information can be incorporated into the network approach and may allow for an individualized assessment of a given patient's disease state. Besides facilitating more efficient treatment approaches, a system of robust yet patient specific hallmarks of a complex disorder would be invaluable in the design of clinical trials, the development of new drug candidates or the identification of novel drug targets. In the context of psychiatric disorders, a personalized diagnosis and treatment approach would be of particular value as patients' responsiveness to treatment can currently not be predicted, impeding appropriate and successful therapy. The methodology presented here is centred around finding robust patterns of disease related information. Especially in complex disorders where patient and disease relevant information originates from many disparate sources, the integration into patterns of disease relevant abnormalities may advance the understanding of the diseases. Once these patterns are determined, they could be automatically recognized as soon as the relevant information is gathered from patients. Our perspective is that the approach may result in an automated classification of diseases based on the recognized pattern, speed up and improve the diagnostic process in an unprecedented way.

Acknowledgements

This research was kindly supported by the Stanley Medical Research Institute (SMRI). We want to thank all members of the Bahn Laboratory for discussions, help and encouragement. We also thank WATERS Corp. for technical support, especially Drs Therese McKenna and Mark McDowell. E.S. holds a Cambridge European Trust scholarship. S. B. holds a NARSAD Essel Independent Investigator Fellowship.

References

1. Ronald, J.A.: Trent: Clinical Bioinformatics: Preliminary Entry 2008 (Methods in Molecular Medicine). Humana Press Inc., U.S. (2007)
2. Jones, N.C., Pevzner, P.A.: An Introduction to Bioinformatics Algorithms (Computational Molecular Biology). The MIT Press, Cambridge (2004)
3. Durbin, R., Eddy, S.R., Krogh, A., Mitchison, G.: Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids. Cambridge University Press, Cambridge (1999)
4. Gusfield, D.: Algorithms on Strings, Trees and Sequences: Computer Science and Computational Biology. Cambridge University Press, Cambridge (1997)
5. Gibas, C., Jambeck, P.: Developing Bioinformatics Computer Skills, 1st edn. O'Reilly Media, Inc. (2001)
6. Nagl, S.: Cancer Bioinformatics: From Therapy Design to Treatment. John Wiley & Sons, Chichester (2006)
7. Alon, U.: An Introduction to Systems Biology: Design Principles of Biological Circuits. Crc Mathematical and Computational Biology. Chapman & Hall, Boca Raton (2006)
8. Wagner, A.: Robustness and Evolvability in Living Systems (Princeton Studies in Complexity). Princeton University Press, Princeton (2005)
9. Palsson, B.O.: Systems Biology: Properties of Reconstructed Networks. Cambridge University Press, Cambridge (2006)
10. Wilkinson, D.J.: Stochastic Modelling for Systems Biology (Mathematical and Computational Biology). Chapman & Hall/CRC, Boca Raton (2006)
11. Pastor-Satorras, R., Vespignani, A.: Epidemic Spreading in Scale-Free Networks. Phys. Rev. Lett. 86, 3200 (2001)
12. Gross, T., Dommar D'Lima, C.J., Blasius, B.: Epidemic dynamics on an adaptive network. Phys. Rev. Lett. 96, 208701 (2006)
13. Boccaletti, S., Latora, V., Moreno, Y., Chavez, M., Hwang, D.-U.: Complex Networks: Structure and Dynamics. Physics Reports 424, 175 (2006)
14. Dorogovtsev, S.N., Mendes, J.F.F., Samukhin, A.N.: Structure of Growing Networks with Preferential Linking. Phys. Rev. Lett 85, 4633 (2000)
15. Colizza, V., Barrat, A., Barthélemy, M., Vespignani, A.: Prediction and predictability of global epidemics: the role of the airline transportation network. Proc. Natl. Acad. Sci. U.S.A. 103, 2015 (2006)

16. Newman, M.E., Girvan, M.: Finding and evaluating community structure in networks. Phys Rev. E Stat. Nonlin. Soft Matter Phys. 69, 26113 (2004)
17. Bower, J.M., Bolouri, H. (eds.): Computational Modeling of Genetic and Biochemical Networks. MIT Press, Cambridge (2001)
18. Fall, C.P., Marland, E.S., Wagner, J.M., Tyson, J.J. (eds.): Computational Cell Biology. Springer, Heidelberg (2002)
19. Enright, A.J., Van Dongen, S., Ouzounis, C.A.: An efficient algorithm for large-scale detection of protein families. Nucleic Acids Research 30(7), 1575–1584 (2002)
20. van Dongen, S.: Graph Clustering by Flow Simulation. PhD thesis, University of Utrecht (May 2000)
21. Watts, D.J., Strogatz, S.H.: Collective dynamics of 'small-world' networks. Nature 393, 440–442 (1998)
22. Albert, R., Barabasi, A.: Statistical mechanics of complex networks. Rev. Mod. Phys. 74, 47 (2002)
23. Hatzimanikatis, V., Li, C., Ionita, J.A., Broadbelt, L.J.: Metabolic networks: enzyme function and metabolite structure. Curr. Opin. Struct. Biol. 14, 300–306 (2004)
24. Forster, J., Famili, I., Fu, P., Palsson, B.O., Nielsen, J.: Genome-scale reconstruction of the Saccharomyces cerevisiae metabolic network. Genome Res. 13, 244–253 (2003)
25. Mishra, N.S., Tuteja, R., Tuteja, N.: Signaling through MAP kinase networks in plants. Arch Biochem. Biophys. 452, 55–68 (2006)
26. Rousseau, F., Schymkowitz, J.: A systems biology perspective on protein structural dynamics and signal transduction. Curr. Opin. Struct. Biol. 15, 23–30 (2005)
27. Galperin, M.Y.: Bacterial signal transduction network in a genomic perspective. Environ. Microbiol. 6, 552–567 (2004)
28. Monge, R.A., Roman, E., Nombela, C., Pla, J.: The MAP kinase signal transduction network in Candida albicans. Microbiology 152, 905–912 (2006)
29. Herrgard, M.J., Covert, M.W., Palsson, B.O.: Reconstruction of microbial transcriptional regulatory networks. Curr. Opin. Biotechnol. 15, 70–77 (2004)
30. Kanehisa, M.: Post-genome Informatics. Oxford University Press, Oxford (2000)
31. Barabasi, A.L., Oltvai, Z.: Network biology: understanding the cell's functional organization. Nat. Rev. Genet. 5, 101–113 (2004)
32. Zhang, Z., Luo, Z.W., Kishino, H., Kearsey, M.J.: Divergence Pattern of Duplicate Genes in Protein-Protein Interactions Follows the Power Law. Mol. Biol. Evol. 22, 501–505 (2005)
33. Wagner, A.: The yeast protein interaction network evolves rapidly and contains few redundant duplicate genes. Mol. Biol. Evol. 18, 1283–1292 (2001)
34. Wagner, A., Fell, D.A.: The small world inside large metabolic networks. Proc. R. Soc. London Ser. B 268, 1803 (2001)
35. Guimera, R., Sales-Pardo, M., Amaral, L.A.: Modularity from fluctuations in random graphs and complex networks. Phys. Rev. E Stat. Nonlin. Soft Matter Phys. 70(2 Pt 2), 25101 (2004)
36. Fortunato, S., Barthelemy, M.: Resolution limit in community detection. Proc. Natl. Acad. Sci. U S A 104, 36–41 (2007)
37. http://www.arbylon.net/projects/
38. Gfeller, D., Chappelier, J.C., De Los Rios, P.: Finding instabilities in the community structure of complex networks. Phys. Rev. E Stat. Nonlin. Soft Matter Phys 75, 56135 (2005)

39. Caretta-Cartozo, C., De Los Rios, P., Piazza, F., Lió, P.: Bottleneck genes and community structure in the cell cycle network of S. pombe. PLoS Comput. Biol. 3(6), e103
40. Tetko, I.V., Facius, A., Ruepp, A., Mewes, H.W.: Super paramagnetic clustering of protein sequences. BMC Bioinformatics 6, 82 (2005)
41. Uetz, P., Giot, L., Cagney, G., Mansfield, T.A., Judson, R.S., Knight, J.R., Lockshon, D., Narayan, V., Srinivasan, M., Pochart, P., Qureshi-Emili, A., Li, Y., Godwin, B., Conover, D., Kalbfleisch, T., Vijayadamodar, G., Yang, M., Johnston, M., Fields, S., Rothberg, J.M.: A comprehensive analysis of protein-protein interactions in Saccharomyces cerevisiae. Nature 403, 623–627 (2000)
42. Goh, K.I., Oh, E., Jeong, H., Kahng, B., Kim, D.: Classification of scale-free networks. Proc. Natl. Acad. Sci. U S A 99, 12583–12588 (2002)
43. Li, S., Armstrong, C.M., Bertin, N., Ge, H., Milstein, S., Boxem., M., Vidalain, P.-O., Han, J.-D.J., Chesneau, A., Hao, T., Goldberg, D.S., Li, N., Martinez, M., Rual, J.-F., Lamesch, P., Xu, L., Tewari, M., Wong, S.L., Zhang, L.V., Berritz, G.F., Jacotot, L., Vaglio, P., Reboul, J., Hirozane-Kishikawa, T., Li, Q., Gabel, H.W., Elewa, A., Baumgartner, B., Rose, D.J., Yu, H., Bosak, S., Sequerra, R., Fraser, A., Mange, S.E., Saxton, W.M., Strome, S., van den Heuvel, S., Piano, F., Vandenhaute, J., Sardet, C., Gerstein, M., Doucette-Stamm, L., Gunsalus, K.C., Harper, J.W., Cusick, M.E., Roth, F.P., Hill, D.E., Vidal, M.: A map of the interactome network of the metazoan C. elegans. Science 303, 540–543 (2004)
44. Koonin, E.V., Wolf, Y.I., Karev, G.P.: The structure of the protein universe and genome evolution. Nature 420, 218–223 (2002)
45. Barabasi, A.L.: Network medicine–from obesity to the "diseasome. N Engl. J. Med. 357(4), 404–407 (2007)

Combining Protein-Protein Interaction (PPI) Network and Sequence Attributes for Predicting Hypertension Related Proteins

Richard J.B. Dobson, Patricia B. Munroe, Charles A. Mein, Mark J. Caulfield, and Mansoor A.S. Saqi

The Genome Centre, St Barts and The London School of Medicine and Dentistry, Charterhouse Sq., London, EC1 6BQ
r.j.dobson@qmul.ac.uk, m.saqi@qmul.ac.uk

Abstract. Cardiovascular disease is set to become the number one cause of deaths worldwide. It is therefore important to understand the etiologic mechanisms for hypertension, in order to identify new routes to improved treatment. Human hypertension arises from a combination of genetic factors and lifestyle influences. Here we study hypertension related proteins from the perspective of protein-protein interaction (PPI) networks, pathways, Gene Ontology (GO) categories and sequence properties. We find that hypertension related proteins are not generally associated with network hubs and do not exhibit high clustering coefficients. Despite this, they tend to be closer and better connected to other hypertension proteins on the interaction network than we would expect, with 23% directly interacting. We find that molecular function category 'oxidoreductase' and biological process categories 'response to stimulus' and 'electron transport' are overrepresented. We also find that functional similarity does not correlate strongly with PPI distance separating hypertension related protein pairs and known hypertension related proteins are spread across 36 KEGG pathways. Finally, weighted Bagged PART classifiers were used to build predictive models that combined amino acid sequence with PPI network and GO properties.

1 Introduction

OMIM (Online Medelian Inheritance in Man) is a resource for studying disease genes [16]. It provides a set of positive examples for machine learning approaches to build classifiers for prediction of disease genes. Early work using decision tree based classifiers showed disease genes tend to be longer and more conserved than non-disease genes [21]. Subsequent work included additional sequence based attributes that were also used to construct classifiers [2]. Other annotation related attributes such as co-expression and similarity of Gene Ontology (GO) [4] terms and text mining approaches have also been used for selection of disease gene candidates [3] [26] [22]. More recently attributes based on protein-protein interactions (PPI) have been used in classification approaches [30] [14]. PPI interaction data has shown that disease genes are likely to interact with other disease genes or to share interaction neighbours. Barabassi et al. have shown that 'essential' disease genes, in which mutations are lethal, form hubs (highly connected nodes) whereas 'non-essential' disease genes do not display this tendency [15]. A k-nearest neighbours classifier using network features achieved a prediction accuracy of 0.76 [30]. Many of the studies on classification of disease genes have not distinguished features of specific disease conditions, although PPI network properties for Alzheimers related proteins have been studied by Chen etc al [9] who found that these proteins form a highly connected subnetwork.

Hypertension is a complex disease, it has proved difficult to define the genetic architecture of this phenotype [20]. Human hypertension affects over one billion people worldwide and it contributes to approximately 50% of all cardiovascular disease [12]. The study of simpler Mendelian forms of hypertension has highlighted several causative genes, which alter sodium regulation [20]. However, there have been few clear reproduceable essential hypertension candidate loci identified [1]. Recent genome-wide studies have identified important genes in other complex traits such as type 2 diabetes and obesity [1] [24]. The approach of George et al. [14] which exploits PPI and pathway data together with sequence similarity, had limited success in correctly identifying any of the five known hypertension genes included in the dataset. There is therefore value in attempting to develop a classifier to predict genes that may be likely to be implicated in hypertension. In this exploratory study we examine properties of 65 proteins from OMIM which are listed as being associated with a hypertension phenotype and we report the performance of a classifier which includes PPI network, sequence and GO attributes for the detection of hypertension related proteins.

2 Computational Method

2.1 Dataset

Each record in the OMIM database is associated with a unique identifier which relates to a disease, the observed symptoms and the associated genes. The symptoms field of the OMIM entries were parsed for the term 'hypertension' and the results were manually filtered. The genes associated with OMIM entries displaying hypertension as a symptom were then mapped onto their SWISSPROT protein identifiers [5].

2.2 Protein-Protein Interaction Network Properties

Protein-protein interactions form networks which can be explored using graph theoretic approaches. The networks can be thought of as undirected cyclic graphs where the proteins are nodes and the interactions are edges. If proteins A and B directly interact then there exists an edge connecting nodes A and B. Protein-protein interactions involving the identified hypertension related SWISSPROT identifiers were extracted from the OPHID database [7]. OPHID is an on-line database of human protein-protein interactions built by mapping high-throughput model organism data to human proteins. It also integrates data from yeast two-hybrid based, literature-based interaction and orthology-based interaction sources. The hypertension related SWISSPROT proteins (nodes) present in OPHID are referred to as HTd (hypertension dataset). One thousand datasets, each containing the same number of proteins as HTd (65), were then generated by randomly selecting proteins (nodes) from OPHID. We refer to this group of datasets as Rd1..1000.

In order to investigate the PPI properties relating to hypertension, a 2 step approach was taken. Firstly, the 'general topology' of each HTd protein was investigated whereby PPI properties of each HTd protein were investigated in relation to all surrounding proteins. Secondly, network properties were investigated specifically in relation to other HTd proteins ('dataset topology'). Comparisons were made with the Rd1..1000 datasets. The aim of this analysis was to identify whether HTd proteins were better connected than random and whether any differences could be explained by their general background connectivity. For example, can short distances between HTd proteins be explained through HTd proteins being interaction hubs?

General topology. *Degree of nodes:* The mean degree (total number of edges associated with protein (p)) was calculated for OPHID as a whole, for HTd and Rd1..1000. This measure was then extended to identify the number of proteins within a radius of 3 interaction steps from p (figure 1). *Clustering coefficient:* The clustering coefficient (C) for protein p is the number of links between the proteins that directly interact with p divided by the number of links that could possibly exist between them (if the directly interacting proteins were a clique). This measure originates from Watts and Strogatz [28] who used it to determine whether a network was 'small-world'. The clustering coefficient was calculated for each HTd and each Rd1..1000 protein.

Dataset topology. *Degree of nodes:* The mean degree was recalculated for each dataset (HTd, Rd1..1000) where only interactions with proteins from the same dataset were considered. This measure was then extended to identify the number of proteins from the same dataset within a radius of 3 interaction steps (green nodes in figure 1). *Geodesic distance:* The length of the shortest connecting path between each pair of HTd proteins (HTd protein A to HTd protein B), and each pair of random proteins (Rdx protein A to Rdx protein B)was calculated using Dijkstra's algorithm [10]. *Interaction subnetworks:* We derived *expanded* subnetworks for each of the datasets, using the approach of Chen et al. [9], whereby all the proteins and their directly interacting partners were selected. The proportion of all proteins from each of these *expanded* subnetwork datasets that were contained within the largest connected component were calculated. A connected component is a set of proteins whereby each protein can be reached from any other protein via a combination of interaction steps.

2.3 Hypertension Pathways and Protein Function

To investigate pathway properties of hypertension related proteins, proteins from HTd were mapped to identifiers from the KEGG database [18]. We excluded the

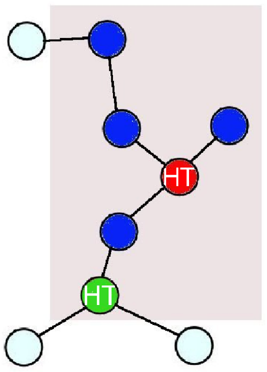


Fig. 1. Figure to show the number of proteins within a chosen radius of a selected hypertension related protein (red node). This figure displays a radius of 2 as an example (greyed area). The blue and green nodes are proteins falling within this radius. The blue node indicates that the protein is not hypertension related whereas the green node indicates a hypertension related protein.

following KEGG identifiers that related to types of interactions as opposed to pathways, although we are aware there is some subjectivity in this selection: ABC transporters, phosphotransferase system (PTS), two-component system, neuroactive ligand-receptor interaction, cytokine-cytokine receptor interaction, ECM-receptor interaction, cell adhesion molecules (CAMs), aminoacyl-tRNA biosynthesis, type II secretion system, type III secretion system, type IV secretion system, SNARE interactions in vesicular transport, ubiquitin mediated proteolysis, proteasome, cell cycle - yeast. The distribution of HTd proteins in the remaining pathways was investigated and compared to Rd1..1000.

The semantic similarity of Gene Ontology (GO) terms from each aspect (biological function, molecular process and cellular location) was obtained for the HTd proteins using the program G-Sesame [27]. The correlation between the semantic similarity of GO terms and geodesic distance apart in the PPI network was then measured for pairs of HTd proteins.

GO slims are cut-down versions of the GO categories containing a subset of the terms in the whole GO. They give a broad overview of the ontology content without the detail of the specific fine grained terms. The distribution of GO slim [4] molecular functions and biological processes were studied in order to identify categories that were overrepresented or underrepresented in hypertension proteins compared to the Rd1..1000.

2.4 Classification

A machine learning classifier was built to identify hypertension proteins using a combination of attributes from the PPI and GO analysis, combined with physicochemical properties of the protein sequences. The training dataset comprised the proteins contained within Rd1..30 (1950 instances) and the HTd dataset (65 instances).

Because there was a large imbalance in the training dataset (many more random proteins than hypertension proteins), a CostSensitive classifier [29] was used as a wrapper around a Bagged PART classifier [13][6]. A cost could then be applied for an incorrect HTd protein classification during ten fold cross validation in an attempt to address the imbalance. This weighted approach has been shown to be a succesful method for coping with class imbalance [8] using a similar type of classifier and has an advantage over undersampling in that there is no loss of information. Choosing a cost depends on priorities. For example, a researcher may be prepared to accept a high false positive rate (FPR) in order to obtain a high rate of recall for hypertension related proteins. The classifier was run 400 times with a range of cost matrices that applied varying penalties for incorrectly predicting a HTd protein.

Physicochemical properties for each protein sequence were calculated using the Protparam program at Expasy (`www.expasy.org`). A bioperl (`www.bioperl.org`) module (Bio::Tools::Protparam) was created specifically for this purpose. Sequence properties used in the classifier were: amino acid length; number of negative amino acids; number of positive amino acids; molecular weight; theoretical pI; number of carbon atoms; number of hydrogen atoms; number of nitrogen atoms; number of oxygen atoms; number of sulphur atoms ; half life; instability Index; stability class; aliphatic index; GRAVY; amino acid composition; The GRAVY (Grand Average of Hydropathy) value for a peptide or protein was calculated as the sum of hydropathy values of all the amino acids, divided by the number of residues in the sequence [19]. In addition, attributes were selected based on the properties of the PPI networks. The attributes calculated for each protein were: the length of the shortest path to the closest known HTd protein; the average and standard deviation of distances from each HTd protein; the number of direct interactions; the number of direct interactions with HTd proteins; the number of proteins up to 2 interactions away (up to one intermediary); the number of HTd proteins up to 2 interactions away; the number of proteins up to 3 interactions away (up to two intermediaries); the number of HTd proteins up to 3 interactions away;

Attributes relating to molecular function and biological process were selected from GO slim categories that were found to be either over or underrepresented within the hypertension dataset, namely, ‘response to stimulus’ (GO:0050896), ‘electron transport’ (GO:0006118) and ‘oxidoreductase activity’ (GO:0016491.

3 Results

We isolated 96 hypertension related genes from OMIM, 90 of which could be mapped to SWISSPROT identifiers. Where an OMIM id had multiple associated proteins, we made the assumption that all were associated with hypertension and included them in the dataset as there was insufficient evidence to assume otherwise. Of the 90 ids, 65 were present within OPHID. These 65 proteins were associated with 47 diseases (distinct OMIM ids) where hypertension was recorded as a symptom. The average number of proteins per OMIM id was 1.5. We refer to this dataset as HTd. The OPHID database used in this study contained 48,222 interactions.

3.1 Network Properties

General topology. *Degree of nodes:* The average degree (number of direct interactions associated with a protein) for the whole of OPHID was 9.04. The HTd proteins had an average degree of 10.0615. The average degree for OMIM genes (that are present in OPHID) was 12.91. The number of proteins within radii of 1 (degree), 2 and 3 interactions from each protein is shown in the top row of quantile-quantile plots in figure 2. The difference in distributions between HTd and Rd1..1000 was only marginally significant for direct interactions and was not significant for interactions within radii of 2 and 3 interactions when using the Wilcoxon rank sum test (p-values = 0.03, 0.09, 0.08 respectively). *Clustering coefficient:* Figure 3 shows the quantile-quantile plot of clustering coefficients (C) for HTd and Rd1..1000. If they come from similar distributions, the distributions should align. Wilcoxon rank sum test with continuity correction shows that they come from the same distribution (p-value = 0.1085). However the Bartlett's K-squared test shows there is heterogeneity of variance (p-value = 0.001368) with the random genes having a wider variance of C. In terms of interacting partners that are involved in no further interactions (C=0), there was no significant difference between the 2 sets, 52.3% HTd proteins and 39% of Rd1..1000 proteins (Chi-squared = 1.857 p-value = 0.1730). There was no significant difference in the proportion of HTd and Rd1..1000 proteins that only have a single interacting partner with 18 HTd proteins and an average of 18.86 across the Rd1..1000 datasets.

Dataset topology. *Degree of nodes:* The second row of quantile-quantile plots in figure 2 show the subset of interactions within radii of 1 (degree), 2, 3 interactions that belonged to the same dataset as the protein p under study. These plots can be compared with the first 3 plots displaying all interactions within similar radii. The difference in distributions between HTd and Rd1..1000 for these subsets of interactions up to a radius of 3 interactions is significant when using the Wilcoxon rank sum test (p-value = 2.49e-11, 3.842e-06, 0.0003 respectively). *Geodesic distance:* Figure 4 shows the geodesic distance between each pair of HTd proteins and each pair of proteins from Rd1..100. We limited to the first 100 random datasets due to the computationally expensive process involved in calculating the distance. The difference in the distribution of distances was significant (Wilcoxon rank sum p=0.004). Fifteen out of 65 (23%) HTd proteins are directly connected. In comparison, on average, only 3 out of every 65 (6%) Rd1..100 proteins are directly connected. *Interaction subnetworks*: There were 623 proteins (646 interactions) in the dataset comprising the HTd proteins and their direct interaction partners. The average number of proteins and directly interacting partners for the Rd1..1000 datasets was 583 (std 109). The largest

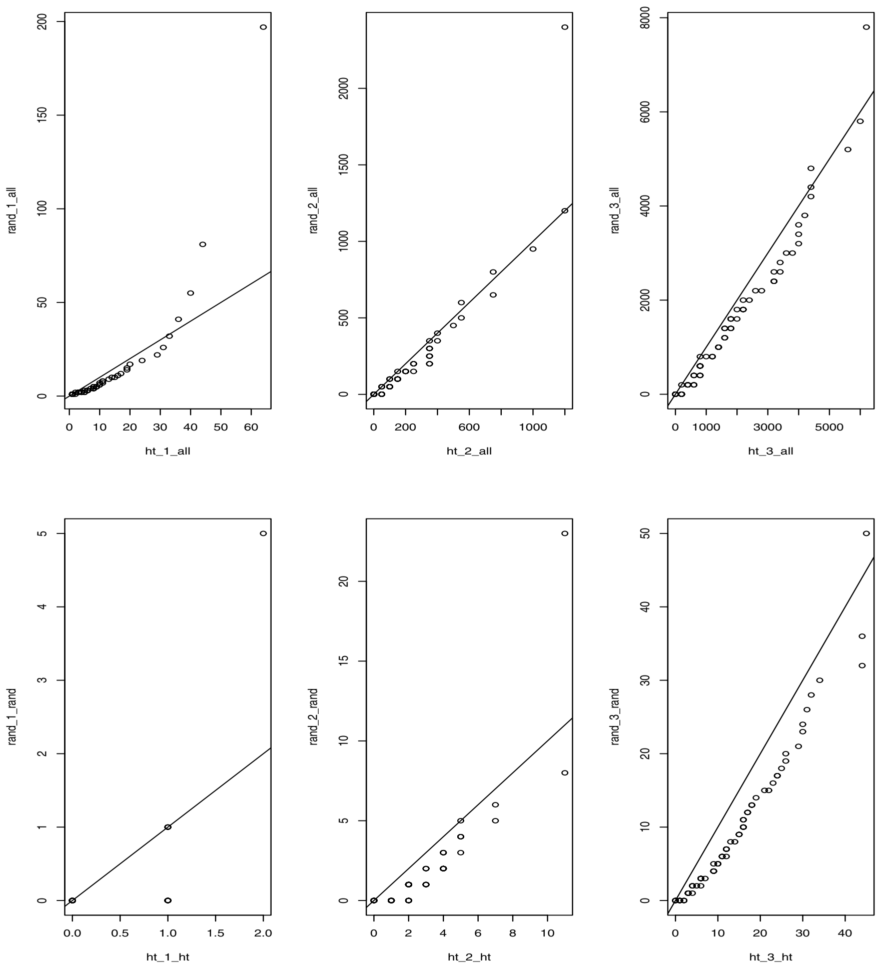


Fig. 2. Quantile-quantile plots for the number of proteins up to a distance of 3 interactions away from HTd and Rd1..1000 proteins. The first three plots relate to all interactions and the second three plots limit to interactions with proteins belonging to the same dataset as the protein being studied. *Axis definitions*: rand_1_all, rand_2_all, rand_3_all - number of proteins within radii of 1, 2, 3 interactions of each Rdx protein (x=1 to 1000); ht_1_all, ht_2_all, ht_3_all - number of proteins within radii of 1, 2, 3 interactions of each HTd protein; rand_1_rand, rand_2_rand, rand_3_rand - number of Rdx proteins within radii of 1, 2, 3 interactions of each Rdx protein; ht_1_ht, ht_2_ht, ht_3_ht - number of HTd proteins within radii of 1, 2, 3 interactions of each HTd protein. The top 3 plots show that both RD1..1000 and HTd proteins have similar distributions in terms of the total number of proteins up to a distance of 3 interaction steps away (Wilcoxon rank sum test p-values = 0.03, 0.09, 0.08 respectively), although there are outiers in the Rd1..1000 proteins acting as hubs. The bottom 3 plots show that there are larger numbers of HTd proteins surrounding any given HTd protein than there are Rdx proteins surrounding Rdx proteins (within the radii up to 3 interactions) (Wilcoxon rank sum test p-values = 2.49e-11, 3.842e-06, 0.0003 respectively).

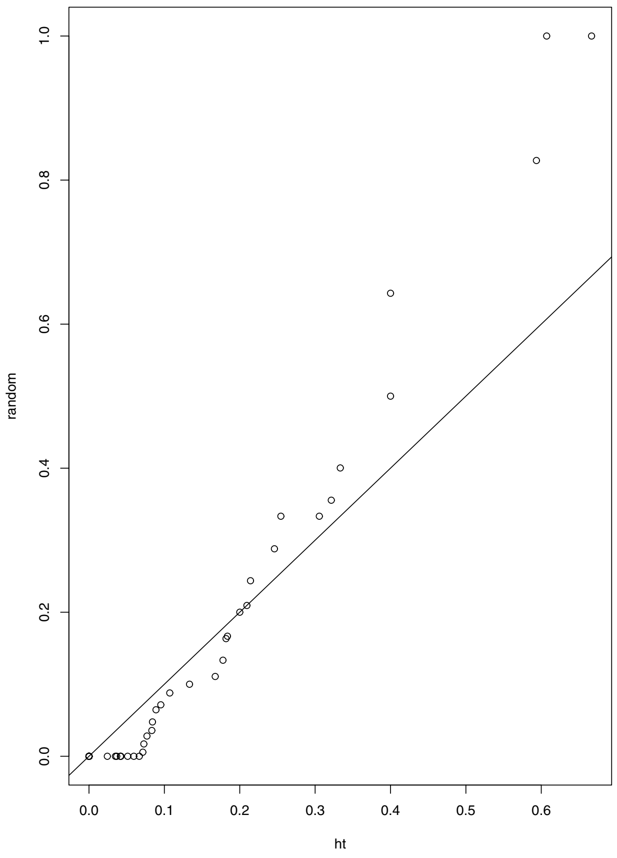


Fig. 3. Quantile-quantile plot of clustering coefficients (C) for the HTd and Rd1..1000 proteins. Wilcoxon rank sum test with continuity correction shows that they come from the same distribution (p-value = 0.1085). However the Bartlett's K-squared test shows there is heterogeneity of variance (p-value = 0.001368) with the Rd1.1000 proteins having a wider variance of C.

connected component in the *expanded* subnetwork involving the HTd proteins and their direct partners contained 550 of the 623 proteins (88%). The size of this subnet is in the upper 5% of the distribution over Rd1..1000 (figure 5).

3.2 Hypertension Pathways and Protein Function

The HTd proteins are spread across 36 KEGG pathways. Three (8%) of these pathways contain 3 HTd proteins, 10 (28%) contain 2 HTd proteins and the remaining pathways (64%) contain single HTd proteins. Table 1 shows the pathways that contain multiple HTd proteins. By comparison, for the subset of 22 Rd1..1000 datasets that map to the same number of pathways (36), only 3%

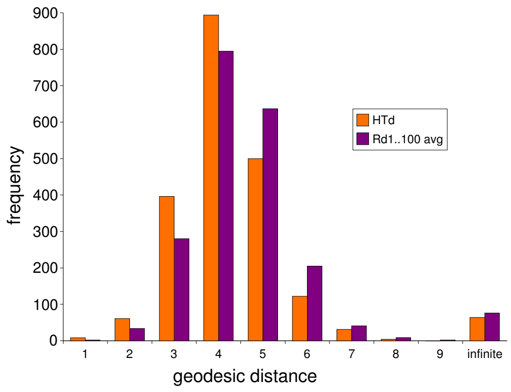


Fig. 4. Figure to show the geodesic distances between HTd protein pairs and Rdx protein pairs. Infinite relates to protein pairs that are unconnected, both directly and indirectly.

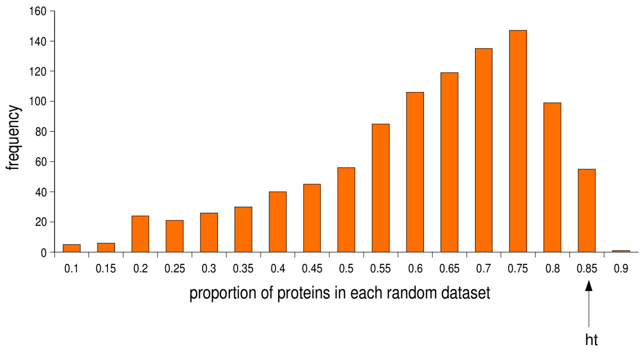


Fig. 5. The proportion of proteins in the largest connected component for HTd and each Rd1..1000 *expanded* subnetworks. In the HTd *expanded* subnetwork, the largest connected component contains 88% of the proteins.

of the pathways contain 3 proteins, 15% contain 2 proteins and 82% contain 1 protein. The clustering of HTd proteins in KEGG pathways is significantly different to the pattern observed in the subset of Rd1..1000 datasets that map to 36 pathways (Wilcoxon rank sum test p=0.02).

We wanted to investigate the origin of the observed high level of connectivity in 'dataset topological' properties of the HTd dataset. HTd proteins that clustered in pathways were investigated to see whether they originated from the same OMIM record. For those that did we noted the geodesic distance separating them. This might help identify any potential biases in the HTd dataset. Of the 3 pathways that each contain 3 HTd proteins, 2 pathways contain HTd proteins that map to the same hypertension related OMIM id. The first of these pathways is the human cell communication pathway (path:dhsa01430). An OMIM

Table 1. Table to show the KEGG Homo sapiens pathways that contain multiple HTd proteins

Pathway ID	Description	No. of HTd proteins
path:dhsa00500	Starch and sucrose metabolism	3
path:dhsa01430	Cell Communication	3
path:dhsa04610	Complement and coagulation cascades	3
path:dhsa00052	Galactose metabolism	2
path:dhsa00140	C21-Steroid hormone metabolism	2
path:dhsa00561	Glycerolipid metabolism	2
path:dhsa00600	Sphingolipid metabolism	2
path:dhsa03320	PPAR signaling pathway	2
path:dhsa04350	TGF-beta signaling pathway	2
path:dhsa04630	Jak-STAT signaling pathway	2
path:dhsa04640	Hematopoietic cell lineage	2
path:dhsa04742	Taste transduction	2
path:dhsa05216	Thyroid cancer	2

id (215600 Cirrhosis, familial) is shared between 2 of the 3 HTd proteins in this pathway. The respective proteins are: K1C18_HUMAN [P05783] (Keratin, type I cytoskeletal 18 (Cytokeratin-18) and K2C8_HUMAN [P05787] (Keratin, type II cytoskeletal 8 (Cytokeratin-8). These proteins are separated by a geodesic distance of 4. The second pathway containing 3 HTd proteins is the complement and coagulation cascades pathway (path:dhsa04610). Again, one OMIM id (235400 hemolytic uremic syndrome) is shared between 2 of the 3 HTd proteins in this pathway. The proteins are: CFAH_HUMAN [P08603] (Complement factor H precursor (H factor 1)) and MCP_HUMAN [P15529] (Membrane cofactor protein precursor (Trophoblast leukocyte common antigen)). The geodesic distance between these proteins is 2. Only 1 of the 10 pathways that contain 2 HTd proteins have proteins that map to the same hypertension related OMIM id. This pathway is the taste transduction pathway (path:dhsa04742). The shared OMIM id is 177200 (Liddle syndrome). The 2 proteins in this pathway that share this OMIM id are: SCNNB_HUMAN [P51168] (Amiloride-sensitive sodium channel subunit beta (Epithelial Na(+) channel subunit beta)) and SCNNG_HUMAN [P51170] (Amiloride-sensitive sodium channel subunit gamma (Epithelial Na(+) channel subunit gamma)). These proteins directly interact in the PPI network.

There was not a strong correlation between GO semantic similarity and geodesic distance for HTd protein pairs. Correlations were calculated for each aspect of GO (molecular function, biological process and cellular component).

Most of the hypertension related proteins fall into GO slim categories of binding (GO:0005488), protein binding (GO:0005515) and catalytic activity (GO:0003824). The difference in the overall distribution of GO slim biological process categories between hypertension and Rd1..1000 proteins is significant (p-value = 0.01554) whereas the distribution of molecular function GO slim categories is not (p-value = 0.5369). In terms of biological processes, specific

GO slim categories 'response to stimulus' (GO:0050896) and 'electron transport' (GO:0006118) are overrepresented within the hypertension dataset with p = 0.005277 and p = 0.0009852 repectively. In terms of molecular functions, 'oxidoreductase activity' (GO:0016491) is overrepresented within the hypertension dataset (p-value = 0.01219). These categories are still significantly overrepresented following the removal of 3 homologs in HTd.

3.3 Classification

When benchmarking the classifier we wished to identify any sequence similar proteins as some of our attributes are sequence based. BLASTClust (at a level of 25% identity) [11] showed that the HTd dataset was not heavily populated with sequence homologs. Only 2 pairs of proteins were found to share more than 25% identity. The first protein pair was: SCNNB_HUMAN [P51168] (Amiloride sensitive sodium channel subunit beta) and SCNNG_HUMAN [P51170] (Amiloride sensitive sodium channel subunit gamma). These proteins shared 34% sequence identity (E=3e-102). The second protein pair was: C11B1_HUMAN [P15538] (Cytochrome P450 11B1, mitochondrial precursor) and C11B2_HUMAN [P19099] (Cytochrome P450 11B2, mitochondrial precursor). These proteins shared 85% sequence identity (E=0.0). All proteins were included in the machine learning classification.

Various feature selection methods were tested using the WEKA workbench [29] to remove redundancy and identify key attributes. The CfsSubsetEval evaluator used with the BestFirst search method identified seven key attributes: percentage amino acid composition of G; percentage amino acid composition of K; the geodesic distance to the closest HTd protein; the standard deviation

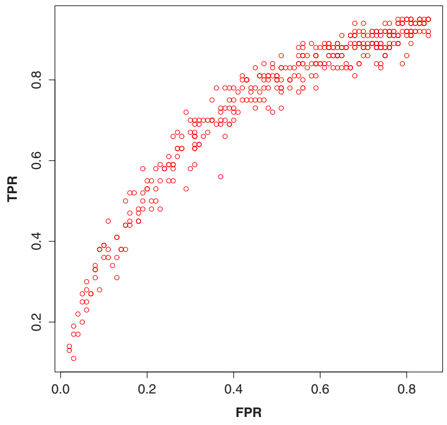


Fig. 6. Figure to show the true positive rate [TPR] against false positive rate [FPR]) when predicting hypertension proteins using a weighted Bagged PART classifer. The penalty for an incorrect prediction was varied by using a CostSensitive classifier.

of the geodesic distances to each HTd protein; whether the protein belonged to GO slim categories 'response to stimulus' (GO:0050896) and 'electron transport' (GO:0006118); the number of direct connections with HTd proteins. A weighted Bagged PART classifier was run 400 times over a range of penalties (using a cost matrix) for incorrectly predicting a HTd protein using the 7 key attributes. The runs were repeated using the simple majority-rule approach but the TPR never exceeded the FPR. Figure 6 shows the true positive rate (TPR) plotted against the false postive rate (FPR) when predicting hypertension proteins.

4 Discussion

We find that there is little difference in the general background topology of protein-protein interaction networks between HTd proteins and Rd1..1000 proteins. We find that HTd proteins do not form large hubs and they do not display high cluster coeffecient (C) scores. Previous studies [23] [25] [17] [30] have suggested that disease genes were likely to form hubs. However, Barabassi [15] suggested recently that these studies included 'essential' genes in which any mutations are lethal. Once these genes had been excluded it was shown that the remaining 'non-essential' disease genes did not tend to form hubs. HTd are likely to be 'non-essential' genes and our findings are consistent with [15]. OMIM has a average degree of 13 which is higher than the hypertension proteins (10) and OPHID (9) possibly because it includes these 'essential' disease genes.

Despite the insignificant differences in background network topology, we find that HTd proteins display greater connectivity in relation to each other than we might expect. HTd protein pairs exhibit shorter geodesic distances than random and the largest *expanded* subnet size lies within the top 5% of the distribution for the random datasets. This means that 88% of the proteins are connected (directly or indirectly) when a network is created using HTd proteins and their direct partners. It is similar to previously observed distributions in Alzheimers disease proteins where the largest subnet contained 83% of the proteins [9]. There is also a significant difference from random in the number of HTd proteins within a radius of 3 interactions from any other HTd protein.

The HTd proteins are spread over 36 KEGG pathways, reflecting the complex, locus rich nature of hypertension related proteins. We might have expected to see HTd proteins that cluster in the same pathway to have originated from the same OMIM id and be close in the PPI network. We found that this was not always the case. The proteins were usually associated with different diseases where hypertension was a symptom. Where proteins sharing a pathway originated from the same OMIM id, only 1 of the 3 HTd protein pairs were directly connected.

We expected to see a negative correlation between distance separating two HTd proteins in the PPI network and GO semantic similarity. However, we were unable to show this correlation in our dataset. The difference in the distribution of GO slim biological process categories between HTd proteins and the Rd1.1000 was significant. There were a number of notable molecular function and biological process categories that were overrepresented in the hypertension dataset, namely 'response to stimulus' (GO:0050896), 'oxidoreductase activity' (GO:0016491), 'nucleic acid binding' (GO:0003676).

There are caveats with the OMIM hypertension dataset, however at present the OMIM database is the most complete repository of diseases and their associated genes. There was concern that the increased connectivity of the HTd proteins may be due to biases in the PPI resource. We might expect the hypertension related proteins to have been studied more than the randomly selected proteins and therefore to see a larger number of documented interactions. However, if this were the case, we would have expected more of them to be hubs. Potential interaction biases could be further investigated by considering interactions, such as those from high throughput experiments, separately. Because the sources of OPHID interactions vary in their reliability, we created a second weighted network, retaining the same proteins and interactions but assigning a weight (or distance) to each interaction in a similar manner to [9]. In this weighted network, proteins were separated by a distance relating to annotation confidence. Interactions with high quality annotation retained their default distance of 1, medium quality interactions were separated by a distance of 1.5 and low quality interactions a distance of 2. We then repeated relevant analyses. Our results did not show significant trend differences to the unweighted analyses with respect to GO semantic similarity and geodesic distance correlations.

The methods described here could easily be applied to other disease datasets in OMIM. The hypertension dataset itself would be improved with the addition of further hypertension related proteins. However, the model constructed shows that there are patterns within PPI networks, shared function and sequence based properties that can be used to aid prioritisation of candidate gene lists identified through experiments such as genome wide association studies. We anticipate that machine learning analyses that combine such attributes will be useful in helping to characterise disease related genes in future studies.

References

1. Genome-wide association study of 14,000 cases of seven common diseases and 3,000 shared controls. Nature 447(7145) 661–678 (2007)
2. Adie, E.A., Adams, R.R., Evans, K.L., Porteous, D.J., Pickard, B.S.: Speeding disease gene discovery by sequence based candidate prioritization. BMC Bioinformatics 6, 55 (2005)
3. Adie, E.A., Adams, R.R., Evans, K.L., Porteous, D.J., Pickard, B.S.: SUSPECTS: enabling fast and effective prioritization of positional candidates. Bioinformatics 22(6), 773–774 (2006)
4. Ashburner, M., Ball, C.A., Blake, J.A., Botstein, D., Butler, H., Cherry, J.M., Davis, A.P., Dolinski, K., Dwight, S.S., Eppig, J.T., Harris, M.A., Hill, D.P., Issel-Tarver, L., Kasarskis, A., Lewis, S., Matese, J.C., Richardson, J.E., Ringwald, M., Rubin, G.M., Sherlock, G.: Gene ontology: tool for the unification of biology. The Gene Ontology Consortium. Nat. Genet. 25(1), 25–29 (2000)

5. Boeckmann, B., Bairoch, A., Apweiler, R., Blatter, M.C., Estreicher, A., Gasteiger, E., Martin, M.J., Michoud, K., O'Donovan, C., Phan, I., Pilbout, S., Schneider, M.: The SWISS-PROT protein knowledgebase and its supplement TrEMBL in 2003. Nucleic Acids Res. 31(1), 365–370 (2003)
6. Breiman, L.: Bagging predictors. Machine Learning 24(2), 123–140 (1996)
7. Brown, K.R., Jurisica, I.: Online predicted human interaction database. Bioinformatics 21(9), 2076–2082 (2005)
8. Chen, C., Liaw, A., Breiman, L.: Using random forest to learn imbalanced data. Technical Report 666, Department of Statistics, University of California, Berkeley (2004), http://www.stat.berkeley.edu/tech-reports/666.pdf
9. Chen, J.Y., Shen, C., Sivachenko, A.Y.: Mining Alzheimer disease relevant proteins from integrated protein interactome data. In: Pac. Symp. Biocomput., pp. 367–378 (2006)
10. Dijkstra, E.W.: A note on two problems in connexion with graphs. Numerische Mathematik 1, 269–271 (1959)
11. Dondoshansky, I.: Blastclust (NCBI Software Development Toolkit), 6.1 edn., NCBI, Bethesda, MD (2002)
12. Ezzati, M., Vander Hoorn, S., Lawes, C.M., Leach, R., James, W.P., Lopez, A.D., Rodgers, A., Murray, C.J.: Rethinking the "diseases of affluence" paradigm: global patterns of nutritional risks in relation to economic development. PLoS Med 2(5), e133 (2005)
13. Frank, E., Witten, I.H.: Generating accurate rule sets without global optimization. In: Proc. 15th International Conf. on Machine Learning, pp. 144–151. Morgan Kaufmann, San Francisco (1998)
14. George, R.A., Liu, J.Y., Feng, L.L., Bryson-Richardson, R.J., Fatkin, D., Wouters, M.A.: Analysis of protein sequence and interaction data for candidate disease gene prediction. Nucleic Acids Res. 34(19), e130 (2006)
15. Goh, K.I., Cusick, M.E., Valle, D., Childs, B., Vidal, M., Barabasi, A.L.: The human disease network. Proc. Natl. Acad. Sci. U S A 104(21), 8685–8690 (2007)
16. Hamosh, A., Scott, A.F., Amberger, J., Bocchini, C., Valle, D., McKusick, V.A.: Online Mendelian Inheritance in Man (OMIM), a knowledgebase of human genes and genetic disorders. Nucleic Acids Res. 30(1), 52–55 (2002)
17. Jonsson, P.F., Bates, P.A.: Global topological features of cancer proteins in the human interactome. Bioinformatics 22(18), 2291–2297 (2006)
18. Kanehisa, M., Goto, S., Hattori, M., Aoki-Kinoshita, K.F., Itoh, M., Kawashima, S., Katayama, T., Araki, M., Hirakawa, M.: From genomics to chemical genomics: new developments in KEGG. Nucleic Acids Res. 34(Database issue), 354–357 (2006)
19. Kyte, J., Doolittle, R.F.: A simple method for displaying the hydropathic character of a protein. J. Mol. Biol. 157(1), 105–132 (1982)
20. Lifton, R.P., Gharavi, A.G., Geller, D.S.: Molecular mechanisms of human hypertension. Cell 104(4), 545–556 (2001)
21. Lopez-Bigas, N., Ouzounis, C.A.: Genome-wide identification of genes likely to be involved in human genetic disease. Nucleic Acids Res. 32(10), 3108–3114 (2004)
22. Perez-Iratxeta, C., Wjst, M., Bork, P., Andrade, M.A.: G2D: a tool for mining genes associated with disease. BMC Genet. 6, 45 (2005)
23. Rual, J.F., Venkatesan, K., Hao, T., Hirozane-Kishikawa, T., Dricot, A., Li, N., Berriz, G.F., Gibbons, F.D., Dreze, M., Ayivi-Guedehoussou, N., Klitgord, N., Simon, C., Boxem, M., Milstein, S., Rosenberg, J., Goldberg, D.S., Zhang, L.V., Wong, S.L., Franklin, G., Li, S., Albala, J.S., Lim, J., Fraughton, C., Llamosas,

E., Cevik, S., Bex, C., Lamesch, P., Sikorski, R.S., Vandenhaute, J., Zoghbi, H.Y., Smolyar, A., Bosak, S., Sequerra, R., Doucette-Stamm, L., Cusick, M.E., Hill, D.E., Roth, F.P., Vidal, M.: Towards a proteome-scale map of the human protein-protein interaction network. Nature 437(7062), 1173–1178 (2005)

24. Sladek, R., Rocheleau, G., Rung, J., Dina, C., Shen, L., Serre, D., Boutin, P., Vincent, D., Belisle, A., Hadjadj, S., Balkau, B., Heude, B., Charpentier, G., Hudson, T.J., Montpetit, A., Pshezhetsky, A.V., Prentki, M., Posner, B.I., Balding, D.J., Meyre, D., Polychronakos, C., Froguel, P.: A genome-wide association study identifies novel risk loci for type 2 diabetes. Nature 445(7130), 881–885 (2007)
25. Stelzl, U., Worm, U., Lalowski, M., Haenig, C., Brembeck, F.H., Goehler, H., Stroedicke, M., Zenkner, M., Schoenherr, A., Koeppen, S., Timm, J., Mintzlaff, S., Abraham, C., Bock, N., Kietzmann, S., Goedde, A., Toksoz, E., Droege, A., Krobitsch, S., Korn, B., Birchmeier, W., Lehrach, H., Wanker, E.E.: A human protein-protein interaction network: a resource for annotating the proteome. Cell 122(6), 957–968 (2005)
26. Tiffin, N., Kelso, J.F., Powell, A.R., Pan, H., Bajic, V.B., Hide, W.A.: Integration of text- and data-mining using ontologies successfully selects disease gene candidates. Nucleic Acids Res 33(5), 1544–1552 (2005)
27. Wang, J.Z., Du, Z., Payattakool, R., Yu, P.S., Chen, C.F.: A new method to measure the semantic similarity of GO terms. Bioinformatics 23(10), 1274–1281 (2007)
28. Watts, D.J., Strogatz, S.H.: Collective dynamics of 'small-world' networks. Nature 393(6684), 440–442 (1998)
29. Witten, I., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations. Morgan Kaufmann, San Francisco (1999)
30. Xu, J., Li, Y.: Discovering disease-genes by topological features in human protein-protein interaction network. Bioinformatics 22(22), 2800–2805 (2006)

Genome Structure and Characterisation of an Endogenous Retrovirus from the Zebrafish Genome Project Database

Roziah Kambol[1,*] and M. Faris Abtholuddin[2]

Biotechnology Programme[1,2], School of Science and Technology,
Universiti Malaysia Sabah, Locked Bag No 2073
88999 Kota Kinabalu, Sabah
roziahk@ums.edu.my

Abstract. A 9.5Kb nucleotide sequence of an endogenous retrovirus was obtained during an attempt to screen a reverse transcriptase-like element in the zebrafish genome project database. The element termed here as ZFERV-2 has the typical genomic organization of a retrovirus in which the LTR flanks the viral genes in this way: *LTR-gag-pol-env-LTR*. ZFERV-2, resembles a zebrafish endogenous retrovirus, ZFERV discovered previously by Shen and Steiner (2004) and has all the major conserved motifs of the viral genes. In addition, ZFERV-2 has several characteristics possessed by a retrovirus. Firstly, ZFERV-2 shows a replication defective retrovirus because of huge deletion at the protease gene proximal to *gag* gene is observed. Secondly, an intact genome structure of ZFERV-2 together with 99% sequence similarity on both ends of the LTRs indicate a recent integration into the zebrafish genome. Thirdly, a long leader sequence at 1.5Kb upstream of the viral genes and a large genome size are characteristics shared by retroviruses isolated from lower vertebrates.

Keywords: endogenous retrovirus, genome, reverse transcriptase, piscine.

1 Introduction

Endogenous retroviruses are transmitted vertically through the host germ cells in most identified vertebrate genomes (Herniou et al., 1998). Although defective in nature due to extensive mutation in their genomes, some apparently possessed intact genome and proved infectious in the past. For example, an endogenous retrovirus isolated from mouse genome, Mouse Mammary Tumor Virus (MMTV) functions as a complementary or helper virus for the exogenous counterpart and hence induces the tumour cells to grow (Muhlbock and Bentvelzen, 1968). Another example from several groups of Human Endogenous Retroviruses (HERVs) displayed their pathogenic characteristics (Muir et al., 2004, Johnson et al., 1990, Herbs et al., 1996, Wang-Johannig et al., 2001, Portis, 2002, Ponferrada et al., 2003).

The genomic structure of the endogenous retrovirus is not very much different from the exogenous form. For example, both Murine Leukaemia Virus (MuLV) and

[*] Corresponding author.

MMTV have closely related genomic structure to their exogenous counterpart and may suggest potential infectious characteristics for many years. Feline Endogenous Virus (FeLV), also has similar structure and sequence to the exogenous form. The only difference is in the U3 region of the LTR of endogenous and exogenous FeLV.

Despite being widely investigated from murine and human hosts, little attentions have been given to the study of endogenous retrovirus from lower vertebrate hosts. Nevertheless, one full-length sequence was found in both reptilian (Crocodylian Endogenous Retrovirus, CnEV) and amphibian hosts (*Xenopus laevis* Endogenous Retrovirus, Xen1). Similarly, one full-length sequence was also found in a piscine host, namely the Zebrafish Endogenous Retrovirus (ZFERV). Ironically, all the full-length sequences of endogenous retroviruses detected from lower vertebrates to date shared non-infectious characteristics and yet had an intact genomic structure (Martin et al., 2002, Kambol et al., 2003, Shen and Steiner, 2004). Both CnEV and Xen1 were detected by screening a genomic library of *Crocodylus niloticus* and African Clawed Toad, *Xenopus laevis*. ZFERV, on the other hand, was found when searching for unknown genes required for lymphocyte development in zebrafish. It was proved that ZFERV is expressed by thymus in both larval and adult fish (Shen and Steiner, 2004). All of these viruses have unique features of genomic organization when compared to the retroviruses present within murine and human hosts.

Here we report the characterisation of the second full-length sequence of endogenous retrovirus from Zebrafish, known as Zebrafish Endogenous Retrovirus-2 (ZFERV-2) from the screening of the Zebrafish genome project database. ZFREV-2 has similar features and genomic organisation to ZFERV, despite having a huge deletion at the protease gene that may indicate its non-infectious nature in zebrafish genome.

2 Materials and Methods

Probe development

Six probes in the amino acid form were obtained from the reverse transcriptase gene fragments isolated from various piscine hosts during the previous screening method in the lab using PCR screening method (Kambol, 2003). Briefly, the probes were isolated from the following fish: ZFERV, *Fugu rubripes, Leptobarbus hovenii, Amblyceps* sp. and Pufferfish. The probes targeted the protease and domain 1 to 5 of the reverse transcriptase gene. The length of probes varied according to the type of fish ranging from 103 to163 amino acids.

Genome characterisation and BLAST searching

The amino acid probes were subjected to BLAST searches using BLAST webpage at www.ncbi.nlm.nih.gov/BLAST. A database from the Zebrafish Genome Project database was used to screen out possible retroviral fragments from the database of Zebrafish Genome Project located at www.sanger.ac.uk/Projects/D_rerio/. Series of BLAST program were investigated to search for a significant match between query and subject sequences. This involved the application of tBLASTn program and non-redundant database and then a combination of BLASTp and the protein database (pdb) were used throughout the searching. A significant match which was at a high percentage of identity (more than 60%), high score bit (more than 100) and low E value was taken into consideration for further analysis. Further analysis included protein family determination and LTR identification. For protein family determination at the *pol* gene, Pfam searches were used to confirm the reverse transcriptase, RNAseH and integrase family member from the protein family database. Pfam search was carried out at www.pfam.com. For LTR identification, BLAST2 Sequences were used at the following webpage www.ncbi.nlm.nih.gov/BLAST2Sequences. Subsequent determination for other viral protein from *gag* and *env* genes was carried out manually by investigating the conserved motif in each gene. Conserved motif for *gag* gene was identified by the presence of major homology region (MHR) with the following motif: Φ-X-Q-G-X2-E-X4-F-Φ-X-R-L-X3-Φ where Φ was a hydrophobic residue and X can be any amino acid (Wills and Craven, 1991, Patarca and Heseltine, 1983). Conserved motif for immunosuppressive motif in the *env* gene has the following motif, QNRRLA/LD (Cianciolo et al., 1985, Sonigo et al.,1986).

Initially, BLAST searching was performed at the reverse transcriptase gene region. Then, BLAST searching was expanded to other regions of the full-length sequence of ZFREV-2. After obtaining the reverse transcriptase gene, a probe walking method was performed in which the resultant sequence from the reverse transcriptase gene was then used to read the remaining sequence in both upstream and downstream directions. Basically, 3 to 5 KB nucleotide sequences were cut and investigated further for translation purposes. Then the step (reading and translation on both directions) was repeated until all the viral genes and LTR flanking the viral genes were determined. In this study the element was found at the accession no. CR 792423.5 from clone CH211-190H10 of the Zebrafish genome project database.

DNA translation and sequence analysis
DNA translation was performed using DNAsis software and Open Reading Frame determination was carried out at www.ncbi.nlm.nih.gov/ORFinder. Only a very long Open reading frame was investigated and analysed further.

3 Results

```
          LTR
1     TGAGGGAATTATATTTTTGAGGGAATCATATTTTTTAATGTTTTATGCATGTTCAGAATTGAT
64    TTAAGTTTTTCATATTTTTGATGTTTTATGTTTGATATTGATATTAAGTTTTACTATTTTCTT
127   TACTGATGAAATGCTTTGTTTGAAATATGAGCCTATTGCTTTGTTTGAAATGTGAGCCTATCG
190   CTTAAGCTTCGCTTCTAGAGAAACGGGGTTTTTCCACAGAAGTAAGACGAGAGATCTGAGCTC
253   ATGGACAGCTGTGCTTTGGTACAGGATGTGGTTTAACACCTGCAAGGAATCAGCCGTATCTCA
316   GTCTGTGTAGGGGTGCATGTGTGTTTTGTGTGTGTGTGTGTTTGCAGGCTACTGGCAGATTCT
379   AACTGGTTGAGATGAGATCTGATGATGGGAAGGCGAATAATGGATTGGAGGCTCATCGAGAGT
442   GGGGAGGAGACAACCAAGAATAAAAACTGTTGTTTTATGAATTGTACGGCAATACTGGAGCGC      TATA
505   CCAGTTTGCCTTTTCATGGTGTCTCTTCAGTATTCCTGGGCTCAGCATTGTTGATCTGATCCT      box
568   TCGTTTATTAAACTTTGATTTCTCGATAAAGCTAAGTTTCCTGCGTCTACTTTCAAGCGAATC

                                   LTR
631    CTGAAATCGGGGAGGGAAAAACCATTTTCCTACAA
        PBS
666   ATTTGGAGGGGGCACCCGGATTCGCTCAGACGCTTCGTCGAGTTGGAGGAC
        F  G  G  G  T  R  I  R  S  D  A  S  S  S  W  R  T
717    GATCTCAAATCAACCTCTGCGCAGAACTTCTAATAAGGTAAGCAGTCACCT
         I  S  N  Q  P  L  R  R  T  S  N  K  V  S  S  H  L
768    TTTTGATAAAAGTACTGCAAGATTGGAATTACCAAATTTAAGGTTGATTGG
```

```
      F  D  K  S  T  A  R  L  E  L  P  N  L  R  L  I  G
819  GGATTGAATACCAACATAATCCATAAATAATTTGAATTAGATCAACAATGG
      D  *  I  P  T  *  S  I  N  N  L  N  *  I  N  N  G
870  CAATTAAAATGAGTTGACCCGGCGTGGTTACAAGGTTTAAAAACTCATACT
      N  *  N  E  L  T  R  R  G  Y  K  V  *  K  L  I  L
921  ATTAATTGATTATTAACGAGTCGACCCGGCGTGGTTAAACAAATTTATAAA          repeat 1
      L  I  D  Y  *  R  V  D  P  A  W  L  N  K  F  I  K
972  GACTCAGAATTGGTTTATTATTGAGTTGACCCGGCGTGGTTAAACAAATTT          repeat 2
      T  Q  N  W  F  I  I  E  L  T  R  R  G  *  T  N  L
1023 ATAAAAACTCAGAGTTGACCCGGCGTGGTTAACAATTGGTTATTAAAGGAG
      K  N  S  E  L  T  R  R  G  *  Q  L  V  I  K  G  V
1074 TTGACCCGGCGTGGTTAAACAAATTTATAAAAACTCCGAATTGGTATATTA          repeat 3
      D  P  A  W  L  N  K  F  I  K  T  P  N  W  Y  I  K
1125 AACGAGTCGACCCGGCGTGGTTAAACAAATTTATAAAGACTCAGAATTGGT          repeat 4
      R  V  D  P  A  W  L  N  K  F  I  K  T  Q  N  W  L
1176 TATATTATAGAGTTGACCCGGCGTGGTTAAGCAAATTTATAAAAACTCTGA          repeat 5
      Y  Y  R  V  D  P  A  W  L  S  K  F  I  K  T  L  N
1227 ATTGGTTATTAAAGGAGTTGACCCGGCGCGGTTAAGAAAACTTATAAAAAC
      W  L  L  K  E  L  T  R  R  G  *  E  N  L  *  K  L
1278 TCCGAATTGGTAATTAAACGAGTTGACCCGGCGTGGTTAAGAAATATTATA
     S  E  L  V  I  K  R  V  D  P  A  W  L  R  N  I  I
1329 AAAACTCGTATCGTGTTTTCCTGGCATGGTTGAGAATTGAAGTTTACCCGG
      N  S  Y  R  V  F  L  A  W  L  R  I  E  V  Y  P  A
1380 CGAGGTTAACTCACTCGTAAAAACACATAAAAGAGAGTAAAAGGCAAAACG
      R  L  T  H  S  *  K  H  I  K  E  S  K  R  Q  N  V
1431 TTAAAGGGTGTAAAATTGTGTTAAAGTGTCAATTGTGTGCGGATTCCTGAA            Leader
      K  G  C  K  I  V  L  K  C  Q  L  C  A  D  S  *  M           peptide, LP
1482 TGAGCATGTGTGTGAGTGAATAACTCCGTATTTGGGAGGTCAGAGTGCTGC            667-1818bp
      S  M  C  V  S  E  *  L  R  I  W  E  V  R  V  L  R
1533 GCACCTTCTCTGGACCGTCGCTTAAGTGTGACCCGGTAGGAACGGACGCAG
      T  F  S  G  P  S  L  K  C  D  P  V  G  T  D  A  A
1584 CATTCTGGGAGCTCAGAGAGGAGGGTGTGATTGATGAGCCCTATATTCATA
      F  W  E  L  R  E  E  G  V  I  D  E  P  Y  I  H  R
1635 GGGACTGAGAGCATGTAAATAAAACATCGAGTGAATGAAGGAATCATAATT
     G  T  E  S  M  *  I  K  H  R  V  N  E  G  I  I  I
1686 AGATGCTAAAGTGTTATATTACAAACCAAGAAAGGTGAAAAATATTCCTAA
      D  A  K  V  L  Y  Y  K  P  R  K  V  K  N  I  P  K
1737 GTGGTGGGTGTGTAGATTATTAGGACATAAGGAAGATTGTGAAGTGTGTGA
      W  W  V  C  R  L  L  G  H  K  E  D  C  E  V  C  D
1788 CCCCAATAGGGAGACAAAACCATTATTAAAATGACAACATATAAAGAGACG
     P  Q  *  G  D  K  T  I  I  K  M  T  T  Y  K  E  T
1839 TGGGATGGTATCTGTAAGCAGATAATACATCACACCGAACCAAAAGACAGA
     W  D  G  I  C  K  Q  I  I  H  H  T  E  P  K  D  R
1890 AAGAAAATAAGTAAAACATTAGAAAGTCAGTGGACATATCTCTCAGGTCTA
     K  K  I  S  K  T  L  E  S  Q  W  T  Y  L  S  G  L
1941 AATGGCTGGACAGAGACAACTAAAAATCAGGGAAAGTTACTGAACATACTT
     N  G  W  T  E  T  T  K  N  Q  G  K  L  L  N  I  L
1992 TTGACAGAGGAGAAAAGGGCAGATACAGATTTGGTACTGGCCAGACAAAAT
     L  T  E  E  K  R  A  D  T  D  L  V  L  A  R  Q  N
2043 AGGGACAAAACTGGGGTTTGGAAGAAGGGAGCAGCAAACAGAGAAATAGAA
     R  D  K  T  G  V  W  K  K  G  A  A  N  R  E  I  E
2094 AAAGCAGAAGATAGACTGAGACAATGGAGCAGGGTAGCTGCAGCCATGGTT
     K  A  E  D  R  L  R  Q  W  S  R  V  A  A  A  M  V
2145 ACCAAAGTAAAAAGTGCTGTGTGGGCAGATCAAAGGGACTCACCAGAGAGG
     T  K  V  K  S  A  V  W  A  D  Q  R  D  S  P  E  R
2196 CCCCCACCCTATAGTAGTGCAGAAGAAAAAACCAAAACCCCCTCAGCACCA
     P  P  P  Y  S  S  A  E  E  K  T  K  T  P  S  A  P
2247 ATGCAGATGCCAGTAATGATAGTAAAAGGGGGAGAAATAGAAACTACTACT        Proline
     M  Q  M  P  V  M  I  V  K  G  G  E  I  E  T  T  T
2298 AAACAAGCTAACAAAGATTTCAAGTTCATGATAAAACAGGGCGCTATAGAG
     K  Q  A  N  K  D  F  K  F  M  I  K  Q  G  A  I  E
2349 GTCGAAGAAACAGAGGAAGATAAAAAACAAAGACAGTTGAAAACGGCCCTA
     V  E  E  T  E  E  D  K  K  Q  R  Q  L  K  T  A  L
2400 AAAGAAGTTCAGACTAAGATAGAAATAGAGGAAGCAGTAAAACAGGAAATA        Major   homology
     K  E  V  Q  T  K  I  E  I  E  E  A  V  K  Q  E  I        region
2451 GCAAAGCAGTTTACCGCAAACATTCAACTCGCAAGAGACAGTGTAAACCAA
     A  K  Q  F  T  A  N  I  Q  L  A  R  D  S  V  N  Q
2502 ACAATGTCGAGGGTAGCAGAAATGGAGGAAGAGCTGAAGAGAAACCTCAAC
     T  M  S  R  V  A  E  M  E  E  E  L  K  R  N  L  N
```

```
2553 GAAGAAGAGGGAAGTGTAAGATCTGAAGGGACCACATCAGTAGGGGGTACC
     E  E  E  G  S  V  R  S  E  G  T  T  S  V  G  G  T
2604 TGTGAGGACAGAAACAAGATAATAGTATCAGGGTATCAATTGGATGTAGAC
     C  E  D  R  N  K  I  I  V  S  G  Y  Q  L  D  V  D
2655 TGGGGAAGAGAGGCTGCATTACAAAGAGAGCACAGAGCACAAGGTCCCGAG
     W  G  R  E  A  A  L  Q  R  E  H  R  A  Q  G  P  E
2706 CTAGATAGCTTGTCAGACAGAACTAAAAGATGGGTGCATGCAGAGGAGAAT
     L  D  S  L  S  D  R  T  K  R  W  V  H  A  E  E  N
2757 AGGGAACAACAAATGCCCTTGTTTATGAAAAACTTACACAGCACCCCAAAA
     R  E  Q  Q  M  P  L  F  M  K  N  L  H  S  T  P  K
2808 TTAGTGATGCCACTAATCAGGGCAGCTACAGGAAGGAAGGAATACAAACCC
     L  V  M  P  L  I  R  A  A  T  G  R  K  E  Y  K  P
2859 TGGGGACATACAGATATGAATGCTGTGCTAAGTAAATTACCAGAAATTACT
     W  G  H  T  D  M  N  A  V  L  S  K  L  P  E  I  T      Gag
2910 AAGGGAGGTCAGAGGTGGTTCACTAAACTGTTGACCCTGACCCATGGGACA     1819-3521bp
     K  G  G  Q  R  W  F  T  K  L  L  T  L  T  H  G  T
2961 GACCTAGCACTGGGAGATGTTAGAGCATTATGGGGAAGCATACTAACTAGA
     D  L  A  L  G  D  V  R  A  L  W  G  S  I  L  T  R
3012 ACACAGGTAGAACTGATAGAAAGGGAAGCTAACACGACCACAGAAGAAAAT
     T  Q  V  E  L  I  E  R  E  A  N  T  T  T  E  E  N
3063 GAAGAACCTTTAAACAGGTTCAGTACTGAAGTGGGAAGCGCCATGAGGCGA
     E  E  P  L  N  R  F  S  T  E  V  G  S  A  M  R  R
3114 ATTTACCCCACCCCAAAATTGACTTATCAAAGCATAAAGTTCAAAATAATG
     I  Y  P  T  P  K  L  T  Y  Q  S  I  K  F  K  I  M
3165 ACTGGAGAATCTGCATCAGCATATCTGCACAGGTGTGAGGCTGAGTGGGAA
     T  G  E  S  A  S  A  Y  L  H  R  C  E  A  E  W  E
3216 GATAGGACAGGGGAAAATCCAGACTCCTCTGACATATGTAAAGAATTTTTC
     D  R  T  G  E  N  P  D  S  S  D  I  C  K  E  F  F
3267 AGACAGGCTGTGATTAAGGGTGTTCCTGCGAGCGCTGTAGCCGCCATAGAA
     R  Q  A  V  I  K  G  V  P  A  S  A  V  A  A  I  E
3318 AATAGTCCAGACATGCAGGGGGGAGAAGGGGAGGTGTGGACTCGCCATTTC
     N  S  P  D  M  Q  G  G  E  G  E  V  W  T  R  H  F
3369 ATCCACCACGTAGAACCAGTATTATTTCAGATAGACAGAACAGAAATAGTA
     I  H  H  V  E  P  V  L  F  Q  I  D  R  T  E  I  V
3420 AACGTCCGACAGTATAAACTTCGACCAGAAGCTGTGGAGGGAATAGGGGAA
     N  V  R  Q  Y  K  L  R  P  E  A  V  E  G  I  G  E
3471 ACCATAGAAGAGTTAGAGGCTGCTGAGGTCCTACGCAGAACAGTGTCTGAC
     T  I  E  E  L  E  A  A  E  V  L  R  R  T  V  S  D
3522 TGGAACACCCCAATCCTTCCTGTTTTGAAAAAGACAACTGGAAAATACAGA
     W  N  T  P  I  L  P  V  L  K  K  T  T  G  K  Y  R
3573 ATGGTGCATGATTTAAGGCTAATCAATGAGAAAGTATTGACTGCTACCTTA
     M  V  H  D  L  R  L  I  N  E  K  V  L  T  A  T  L
3624 CCCACCCCCAACCCCTATACCATCATGTCTAAATTGACACCAAAACATTCT
     P  T  P  N  P  Y  T  I  M  S  K  L  T  P  K  H  S
3675 CATTTTACGTGCATAGACTTGGCTAATGCATTTTTCTGCATGCCCCTGGCA      reverse
     H  F  T  C  I  D  L  A  N  A  F  F  C  M  P  L  A       transcriptase
3726 GAACAATGTCAAGACATTTTTGCTTTTAGCTATCAGGGAGCGCAATATACT
     E  Q  C  Q  D  I  F  A  F  S  Y  Q  G  A  Q  Y  T
3777 TACAACAGACTACCACAAGGGTTTATTTTAAGCCCAGGTCTGTTCAACCAA
     Y  N  R  L  P  Q  G  F  I  L  S  P  G  L  F  N  Q
3828 GCATTAAGGGAGCTGTTGGACAGCTGTACTTTGCATGAAGGTACCATTGTT
     A  L  R  E  L  L  D  S  C  T  L  H  E  G  T  I  V
3879 ATCCAGTATGTTGATGACTTGTTATTGGCAGCACACTCCAACGAGGTCTGC
     I  Q  Y  V  D  D  L  L  L  A  A  H  S  N  E  V  C
3930 CTGCAGGACACACGAAAAGTACTAACATTACTAAGTACTGCAGGGTTAAAG
     L  Q  D  T  R  K  V  L  T  L  L  S  T  A  G  L  K
3981 GTGAGCAAAGAAAAAATACAAATCAGCAGGGCAACAGTGCACTTCCTTGGA
     V  S  K  E  K  I  Q  I  S  R  A  T  V  H  F  L  G
4032 AGAATAATTGGACAAACAGGCACGGCCCTATCTGATGACACCAAACAAACT
     R  I  I  G  Q  T  G  T  A  L  S  D  D  T  K  Q  T
4083 GTGTTGTCACACCCAAAGCCACTAGTAGTAAAAGACATGATGTCATTTCTG
     V  L  S  H  P  K  P  L  V  V  K  D  M  M  S  F  L
4134 GGGTTGATAGGGTATAGCAGACAATATGTACCAAATTACTCAGAAAGAACT
     G  L  I  G  Y  S  R  Q  Y  V  P  N  Y  S  E  R  T
4185 GCAACATTGAGGGCATTGGCAAAAGAAGTTGGGATGAAAAACAGTAGAGCA
     A  T  L  R  A  L  A  K  E  V  G  M  K  N  S  R  A       Pol
4236 CGACTAAACTGGACACAAGAGGCTGAAGCTGTTTTCTGTGGGCTTAAGGCT       3522-6965bp
     R  L  N  W  T  Q  E  A  E  A  V  F  C  G  L  K  A
4287 GATTTAGCCACTGCAGCAGCCTTGCAGACTCCAAATTATGAATTGCCTTTC
```

```
     D  L  A  T  A  A  A  L  Q  T  P  N  Y  E  L  P  F
4338 TTTCTGGACGTTAGCACCACGGCCTCCACCACAAATGGAGTGTTGTATCAA
     F  L  D  V  S  T  T  A  S  T  T  N  G  V  L  Y  Q
4389 AAACAACACCAGCAAAGGAGGGTTTTGCATTATTTGAGTGCACCCCTAGAT
     K  Q  H  Q  Q  R  R  V  L  H  Y  L  S  A  P  L  D
4440 AAGATAGAGCAAAAACAGCCCACTTGTGCTAGGTATGCTGCTGGCCTGGCT
     K  I  E  Q  K  Q  P  T  C  A  R  Y  A  A  G  L  A
4491 AAATTGATAGAAAAATCTGAGCACATAGTCATGGGACATCCATTACATGTA
     K  L  I  E  K  S  E  H  I  V  M  G  H  P  L  H  V
4542 CTGACGTCACACTCTGTTATATCATTCATCACTTCATCTGCATTCACTTTT
     L  T  S  H  S  V  I  S  F  I  T  S  S  A  F  T  F
4593 TCTGCACAGAGACAGAACAAGCTATTGACCGCCCCACACATCATATATGAA
     S  A  Q  R  Q  N  K  L  L  T  A  P  H  I  I  Y  E
4644 CACCAAGGGGTAAACATGGCTCATGCAGGAGAGGGTGAACCACATGAATGC
     H  Q  G  V  N  M  A  H  A  G  E  G  E  P  H  E  C
4695 ATCCCCAGGGCAGAGGCAGAGGAACAGATCAGACCAGGTTTAAGTAGCATT
     I  P  R  A  E  A  E  E  Q  I  R  P  G  L  S  S  I
4746 CCATTAACTAAACCACAGCTAACTCTGTTCTGTGATGGTTGCTGTTTTAAA
     P  L  T  K  P  Q  L  T  L  F  C  D  G  C  C  F  K
4797 ACTGATTCAGGCAAACTTGTAGCCAGTTACGCCATCGTGGAACAGACTGAC
     T  D  S  G  K  L  V  A  S  Y  A  I  V  E  Q  T  D
4848 GACGGGTATACAATAAGGGAACAGCAGGTGTTGCAAGATAGACCATCAGCA
     D  G  Y  T  I  R  E  Q  Q  V  L  Q  D  R  P  S  A
4899 CAGCGAGCTGAGTTATTGGCCCTTGTGAGAGCCTTGCACATGGCCAAAGAT
     Q  R  A  E  L  L  A  L  V  R  A  L  H  M  A  K  D
4950 AAGACTGTAAATATTTATTCAGACTCAGCATATGCAGTAGGGGCAGCTACA
     K  T  V  N  I  Y  S  D  S  A  Y  A  V  G  A  A  T           RNAse H
5001 TCTGAACTGACTGGTTGGGCAAGAGTGGGGTTTGTAACATCCTCTGGGAAG
     S  E  L  T  G  W  A  R  V  G  F  V  T  S  S  G  K
5052 CCTATAAAGCACGCACAAGAAGCCTCTGATTTGTTAGAATCAATTATGTTG
     P  I  K  H  A  Q  E  A  S  D  L  L  E  S  I  M  L
5103 CCACAAGAGGTAGCAATAATTAAATGTGCAGCACACACCAAAGGAAAAGAT
     P  Q  E  V  A  I  I  K  C  A  A  H  T  K  G  K  D
5154 CCTGTTTCATTAGGAAACGAGGCAGCAGACGCCGCTGCTAAAACAGTGGCA
     P  V  S  L  G  N  E  A  A  D  A  A  A  K  T  V  A
5205 GGGTACAAACCATTGCAAATGACTGTGACTGCAGTTGACGAGCTGCATCAA
     G  Y  K  P  L  Q  M  T  V  T  A  V  D  E  L  H  Q
5256 ATTGACGAACATCTAACTACAAGTTTTCTATCTAAAGAACAAAGTTTGGCT
     I  D  E  H  L  T  T  S  F  L  S  K  E  Q  S  L  A
5307 GCAGCTGAGGAAATTTCAGTATGGTTAGAAAAGGGAGGAAGAAAAGATTCC
     A  A  E  E  I  S  V  W  L  E  K  G  G  R  K  D  S
5358 CAGACAGGACTATGGGTCGGTCCTACAGGTAGACCAATTATGCCTGCAAAT
     Q  T  G  L  W  V  G  P  T  G  R  P  I  M  P  A  N
5409 TTAGCAGGGAAAGTCCTGACAGAGGCCCACTCTCTGGCTCACAGCAGCGAG
     L  A  G  K  V  L  T  E  A  H  S  L  A  H  S  S  E
5460 AAGGATATGACTAAACGAGTGTCTCAATGGTGGCACCCTTTCATGCCACAC
     K  D  M  T  K  R  V  S  Q  W  W  H  P  F  M  P  H
5511 ATGATAAGTGGAGTAATAGCCTCTTGTCAAACATGTGCAGAGTTCAATGTC
     M  I  S  G  V  I  A  S  C  Q  T  C  A  E  F  N  V
5562 AAGCCAACCTCCAAACCCACTGCAGGGCATTTCCCCACAGATAGGGGTCCA
     K  P  T  S  K  P  T  A  G  H  F  P  T  D  R  G  P
5613 GGGTGTACAGTAGTCATGGACTTTACTGACATGATCACAAGAGTAAACGGA
     G  C  T  V  V  M  D  F  T  D  M  I  T  R  V  N  G
5664 AAAAGGTATCTGCTGGTCCTAGTAGACCAATTCACTGGTTGGCCAGAAGCT
     K  R  Y  L  L  V  L  V  D  Q  F  T  G  W  P  E  A
5715 TTCCCATGCGCTAGGGAAGATGCAGTTTCTGTGGTAAAATGTTTGATCAAC
     F  P  C  A  R  E  D  A  V  S  V  V  K  C  L  I  N           integrase
5766 CAATACATTCCAAGGCATGGGTTCCCTCGTATAATCAGGTCAGACAATGGC
     Q  Y  I  P  R  H  G  F  P  R  I  I  R  S  D  N  G
5817 ACTCATTTCAAAAATGAACATCTGGCGGATGTAGAAAAATTGTTAGGTCTA
     T  H  F  K  N  E  H  L  A  D  V  E  K  L  L  G  L
5868 AAACACCGGTATGGAGCTGTATACCACCCACAAAGTCAGGGAAAAGTGGAG
     K  H  R  Y  G  A  V  Y  H  P  Q  S  Q  G  K  V  E
5919 CGCCTCAATCTAACGCTAAAAAACAAACTGGCAAAAATTTGTCACAGGTCC
     R  L  N  L  T  L  K  N  K  L  A  K  I  C  H  R  S
5970 AAATTGAATTGGGTAGATGCACTTCCCATTGCACTAATGTCAGTACGCTGT
     K  L  N  W  V  D  A  L  P  I  A  L  M  S  V  R  C
6021 TCTATTAATCGAACTACAGGTTTTACTCCATTTGAATTGGCTACAGGAAGA
     S  I  N  R  T  T  G  F  T  P  F  E  L  A  T  G  R
```

```
6072 CAATTCCCGGGCCCATGGGCCCCCTTGCATGCTGGAGACACTGACTCACCG
     Q  F  P  G  P  W  A  P  L  H  A  G  D  T  D  S  P
6123 CAAATGTATCATGATAAAGTATGTGCTGTGATTAACATGTTTTCTCCGCAG
     Q  M  Y  H  D  K  V  C  A  V  I  N  M  F  S  P  Q
6174 AAAAATTGGCCAACAGAGAGTGAAGCATCTAGACCTGCAGAAAACACAACA
     K  N  W  P  T  E  S  E  A  S  R  P  A  E  N  T  T
6225 CTGTGGGTGAGATTGAAACAGCACAAACGAAAATGGTCTAGCCCAAGGTGG
     L  W  V  R  L  K  Q  H  K  R  K  W  S  S  P  R  W
6276 TCTGAACCGTTAAGAGTCACGGCTAGGACATCCCATTGTGTGCAATTAGCA
     S  E  P  L  R  V  T  A  R  T  S  H  C  V  Q  L  A
6327 GGTAAAGGCACAACATGGTATCATTTGTCTGCTTGTATGTTCTGTCCTTCT
     G  K  G  T  T  W  Y  H  L  S  A  C  M  F  C  P  S
6378 CCAGACAGGTCTTTGGCGGATGTCAGAGTTGACCTCAGGAGAGGTCAAAGA
     P  D  R  S  L  A  D  V  R  V  D  L  R  R  G  Q  R
6429 GAGGGAGAAGGAGAAGGAGACACTGAGGAGCCAGAAAGAGAGGGAGAAGGC
     E  G  E  G  E  G  D  T  E  E  P  E  R  E  G  E  G
6480 GGGGAAAGAAGCGGACAAACAGAAGCAGCCCCAGCCCCCAGAACACAGCTA
     G  E  R  S  G  Q  T  E  A  A  P  A  P  R  T  Q  L
6531 AGCGCAACCCCTGTATTTAAAATCTTGCAACGAACTGGAGACTTGCTTCAG
     S  A  T  P  V  F  K  I  L  Q  R  T  G  D  L  L  Q
6582 ACACCAGAACACATTCCCATTGCACATTGTATAAGTGCTGATTATGCACTA
     T  P  E  H  I  P  I  A  H  C  I  S  A  D  Y  A  L
6633 GGAGCTGGGGTAGCTAAACAAATTAGAGACAAATACGGTGTAGAAGAATTG
     G  A  G  V  A  K  Q  I  R  D  K  Y  G  V  E  E  L
6684 AACACTTCAGTCGCCCAACCAGGGGATTGTATCAAAACTACACACGGTCCA
     N  T  S  V  A  Q  P  G  D  C  I  K  T  T  H  G  P
6735 CGACAAATTTACCATTTGGTGACCAAATGGTGGTGTAGGGACCTACCCACC
     R  Q  I  Y  H  L  V  T  K  W  W  C  R  D  L  P  T
6786 TACGAGCATCTCGAGGCTAGTCTGATAAAATTGTGTTACCAGTGTAAGAAA
     Y  E  H  L  E  A  S  L  I  K  L  C  Y  Q  C  K  K
6837 GATAAAAATAAAATATTAGCTATACCAAAATTAGGGTGTGGATTAGACAAA
     D  K  N  K  I  L  A  I  P  K  L  G  C  G  L  D  K
6888 TTAGACTACACCAAAGTAAAAGAAATAATTGAGAAAGTATTCAAAGAGGGC
     L  D  Y  T  K  V  K  E  I  I  E  K  V  F  K  E  G
6939 CACATTCAGGTAATTCTCCTAACAAAATGAATAAAATAAACAAATTGGTGG
       H  S  G  N  S  P  N  K  M  N  K  I  N  K  L  V  V
6990 TGGCATGGGGAATAATGTTGGTGGTAATCCTATGGATAATCTGCCATATGG
       A  W  G  I  M  L  V  V  I  L  W  I  I  C  H  M  E
7041 AATTCGGAGGGGCAACAAAGGAAAAACGGAGTGTAAAAGAAGGGGGGGGAA
       F  G  G  A  T  K  E  K  R  S  V  K  E  G  G  G  T
7092 CAGGGGTGCGTATTACCTTAGAAAGGGAAGAGGGAAAGGATGGGATGTGGC
       G  V  R  I  T  L  E  R  E  E  G  K  D  G  M  W  Q
7143 AGTTTGATCTTTGTCAAGTGATAGATTGTGGGAAGGACCAAGTGGCATGGA
       F  D  L  C  Q  V  I  D  C  G  K  D  Q  V  A  W  R
7194 GAAGATATGATGTGTATGGGTGTTTGTGGCCATCTACCACCACGCTTCCGA
       R  Y  D  V  Y  G  C  L  W  P  S  T  T  T  L  P  N
7245 ATGGTCCCCATTGTCATACCTGGCATGATGTAAATTGGAAAACAGTTCCAT
       G  P  H  C  H  T  W  H  D  V  N  W  K  T  V  P  F
7296 TCACTAAAAAGATGTTGAAAAATAGTCCTTTGGCCAAAGCAGACAGCATCC
       T  K  K  M  L  K  N  S  P  L  A  K  A  D  S  I  Q
7347 AAAATCGACTTAGGTTGTCTAGGGGATACCAACATAGTTGGGGAGGCTGGA
       N  R  L  R  L  S  R  G  Y  Q  H  S  W  G  G  W  K
7398 AAAACACGTTGATTATATCACTAAAAAATGGCAATGATGAAACTGACACGT
      N  T  L  I  I  S  L  K  N  G  N  D  E  T  D  T  Y
7449 ACATCACTCTAGGGGTAGATGTAGAAGGAAAAGACCCACTGGGGTTGATAA
       I  T  L  G  V  D  V  E  G  K  D  P  L  G  L  I  K
7500 AAATCTCCATAAAAAAGCCTAAACCCACAGGGGCGCCAATAATCACAGACC
       I  S  I  K  K  P  K  P  T  G  A  P  I  I  T  D  L
7551 TAACCAAAAACAAAAAGAAAGTAATACAGAGCACTGACTATAGCAATCTGA
       T  K  N  K  K  K  V  I  Q  S  T  D  Y  S  N  L  T
7602 CCCCACTAGATCTGATGACGCTGGAGACAGGCTATCACGAAACAAATTTAT
       P  L  D  L  M  T  L  E  T  G  Y  H  E  T  N  L  W
7653 GGTTAGAATGGGTGACAAACGCAGCCGAAGAATTAGGATTTGAGGGGTGCC
       L  E  W  V  T  N  A  A  E  E  L  G  F  E  G  C  L
7704 TAGCTTGTGCAGCAGGAAGGCCACAATTAAATACTGAACCCGCTCCATTAC
       A  C  A  A  G  R  P  Q  L  N  T  E  P  A  P  L  H
7755 ATGATTATGATAGTTGGGGGTACAAATGTATGTTAAAACTAACAAAGGAAA
       D  Y  D  S  W  G  Y  K  C  M  L  K  L  T  K  E  K
7806 AAAGTCCCAAAAACTGCACTGCACTCAGTAATCTATACCCACCAGCTGGAA
```

Env
6966-8899bp bp

```
          S  P  K  N  C  T  A  L  S  N  L  Y  P  P  A  G  N
7857 ATAAGTCATCAATGGGAGTTGCATTAAAAAGAAGAAAGGGTAACTATACCT
          K  S  S  M  G  V  A  L  K  R  R  K  G  N  Y  T  C
7908 GTTTCAATTTAACTACCTCCAGTGCATCCAAAATATTAGTGGGAGCGTTTA
          F  N  L  T  T  S  S  A  S  K  I  L  V  G  A  F  K
7959 AAAGGGAATGGTGTGGACAGATGATTCAGGAAGGTCATGACAAACTAGGGG
          R  E  W  C  G  Q  M  I  Q  E  G  H  D  K  L  G  G
8010 GATGGGGCAGAGTGGGTTTGTATTACGCGTGTGGAGAAAAGGTGGTATTGG
          W  G  R  V  G  L  Y  Y  A  C  G  E  K  V  V  L  A
8061 CTAGAATAGAACCCACCATGGAAGGAGTATGTGCTATGATAAGAGTCGCTG
          R  I  E  P  T  M  E  G  V  C  A  M  I  R  V  A  V
8112 TTCCTATGGTCGTGATAGGAAATAGGGCTATAAATGGGCATAAAAACAGGC
          P  M  V  V  I  G  N  R  A  I  N  G  H  K  N  R  R
8163 GAAAAAGGTCAGTAAATAGTGATTTCGATTTAACACGCAATAGCCCCACCT
          K  R  S  V  N  S  D  F  D  L  T  R  N  S  P  T  Y
8214 ACATAGACGCAATAGGCATCCCCAGGGGAGTTCCAGATGAATATAAATTGG
          I  D  A  I  G  I  P  R  G  V  P  D  E  Y  K  L  A
8265 CTGACCAAGTAGCAGCAGGGTTCGAAAATATACCTGTAATAGCAGCATTAT
          D  Q  V  A  A  G  F  E  N  I  P  V  I  A  A  L  F
8316 TCCCCGTGACCCCAAATAAAAATGTAGACAGAATAAATTACATACATTACA
          P  V  T  P  N  K  N  V  D  R  I  N  Y  I  H  Y  N
8367 ACGTACAAAGGCTGTCAAATCTCACTAGAGATGCTGTGTCTGGATTAAAAG
          V  Q  R  L  S  N  L  T  R  D  A  V  S  G  L  K  E
8418 AACAATTGGCTGCTACCTCTCTGATGACTATTCAAAATCGGCTAGCTTTAG
          Q  L  A  A  T  S  L  M  T  I  Q  N  R  L  A  L  D
8469 ACATGTTACTATCAGAAAGGGGCGGGGTTTGTTCTATGTTTAAAGACACGT   Immunosuppressive
          M  L  L  S  E  R  G  G  V  C  S  M  F  K  D  T  C    motif
8520 GTTGCACCGTCATACCAAACAACACTGCCCCTGATGGTTCCGTATCCAGGG
          C  T  V  I  P  N  N  T  A  P  D  G  S  V  S  R  A
8571 CTCTGGAGGGACTAAAAGAATTGTCAAATGAACTGAAAGCTAGCTCAGGGA
          L  E  G  L  K  E  L  S  N  E  L  K  A  S  S  G  I
8622 TAGAAAATGCTGTGTCTAAATGGATGTCTGATATGTTTGGGGAATGGAAAG
          E  N  A  V  S  K  W  M  S  D  M  F  G  E  W  K  G
8673 GGGTGATTATTGCTATGTTAACATCAATGGGTATTTTCCTAGGAATTTTGG
          V  I  I  A  M  L  T  S  M  G  I  F  L  G  I  L  V
8724 TGACATGTGGGTGTTGCTGTATTCCCTGTATCAGAAGTTTGATAAATAGGC
          T  C  G  C  C  C  I  P  C  I  R  S  L  I  N  R  L
8775 TGATAATAACTGCAATTGAAAAGAAGGAAAACCCACCTCCTTATCAGATGC
          I  I  T  A  I  E  K  K  E  N  P  P  P  Y  Q  M  P
8826 CCCTCCTTGCTGCAGCGGAGGGGGATGACGTGATGGAGGAAGTGGAGGAAC
          L  L  A  A  A  E  G  D  D  V  M  E  E  V  E  E  L
8877 TATTGGATATAGTGTGATCTCTTTCGAGATCAAGAGAGGGAAT PPT
         I  G  Y  S  V  I  S  F  E  I  K  R  G  N
             LTR
8920 ATTGAGGGAATTATATTTTTGAGGGAATCATATTTTTTAATGTTTTATGCATGTTCAGAATTG
          ------------------------>
8983 ATATTAAGTTTTTCATATTTTTGATGTTTTATGTTTGATATTGATATTAAGTTTTACTATTTT
9046 CTTTACTGATGAAATGCTTTGTTTGAAATATGAGCCTATTGCTTTGTTTGAAATGTGAGCCTA
9109 ATCGCTTAAGCTTCGCTTCTAGAGAAACGGGGTTTTTCCACAGAAGTAAGACGAGAGATCTGA
9172 GCTCATGGACAGCTGTGCTTTGGTACAGGATGTGGTTTAACACCTGCAAGGAATCAGCCGTAT
9235 CTCAGTCTGTGTAGGGGTGCATGTGTGTTTTGTGTGTGTGTGTGTTTGCAGGCTACTGGCAGA
9298 TCTAACTGGTTGAGATGAGATCTGATGATGGGAAGGCGAATAATGGATTGGAGGCTCATCGAG
9361 AGTGGGGAGGAGACAACCAAGAATAAAAACTGTTGTTTTATGAATTGTACGGCAATACTGGAG   Polyade-
9424 CCGCCCAGTTTGCCTTTTCATGGTGTCTCTTCAGTATTCCTGGGCTCAGCATTGTTGATCTGA   nylation
9487 TCCTTCGTTTATTAAACTTTGATTTCTCGATAAAGCTAAGTTTCCTGCGTCTACTTTCAAGCG   signal
9550 AATCCTGAAATCGGGGAGGGAAAAACCATTTTCCTACA 9587
                                           <--------------
                                    LTR
```

Fig. 1. Complete nucleotide and translated amino acid of Zebrafish Endogenous Retrovirus 2 (ZFERV-2). Some common amino acid motifs found in retroviruses are indicated by bold type. Abbreviations are shown as follows: PBS = Primer binding site, LP = Leader peptide, LTR = Long terminal repeat, PPT = Polypurine tract.

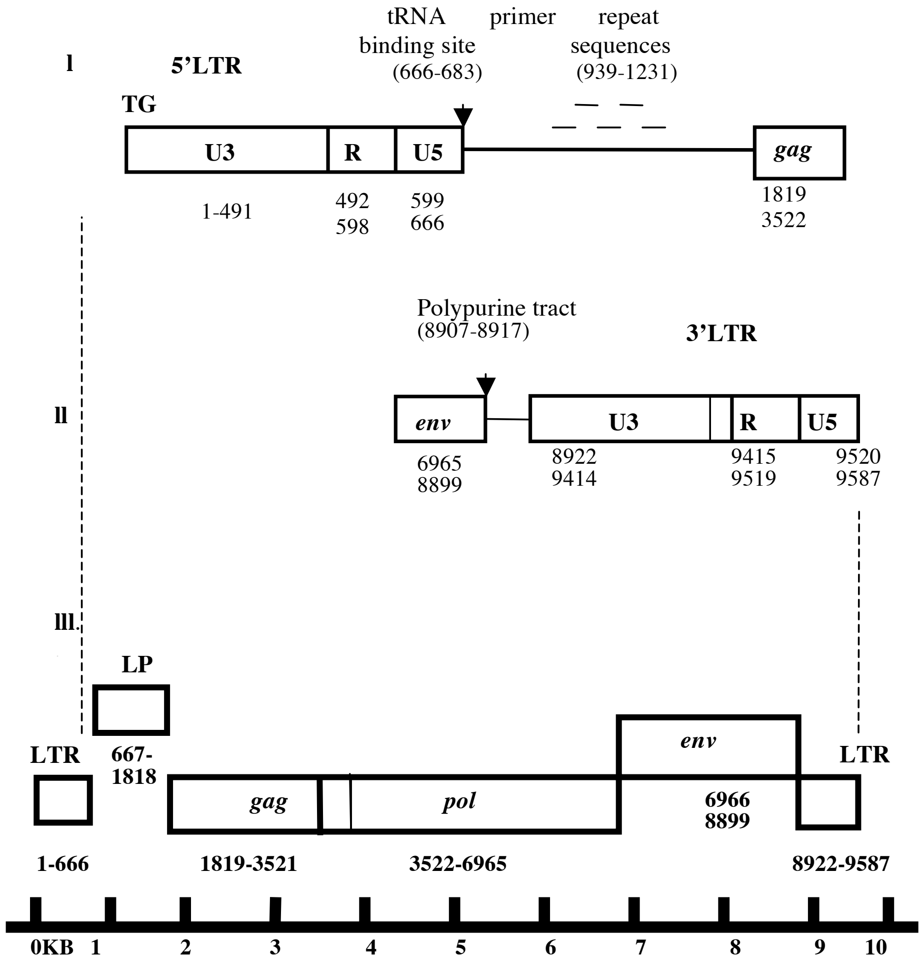


Fig 2. Genomic structure of ZFERV-2. Panel l shows 5'LTR, Leader Peptide region, tRNA primer binding site and *gag* gene. Panel ll shows 3'LTR, polypurine tract and *env* gene. Panel lll shows a complete genomic organization of ZFERV-2, 5'LTR-*gag-pol-env*-LTR 3'.

4 Discussion and Conclusion

The ZFERV-2 genome consists of 9587bp and contains ORFs for *gag, pol* and *env* genes and two flanking LTRs. The 5'LTR and 3'LTR are 666bp and 665bp respectively and are 99% identical (Fig 2 panel I and III). This may indicate a recent integration of ZFERV-2 into the Zebrafish genome (Tristem, 2000).

ZFERV-2 has a long leader peptide sequence (LP) with the size of 1,151 bp. Inside the LP, five repeat sequences of 39bp each have been identified (Fig 1 and Fig 2 panel I). The characteristics of long leader peptide sequence and many repeat sequences within the leader peptide are shared by most retroviruses isolated from lower vertebrates such as *Xenopus Endogenous* Retrovirus 1 (XEN1), Walleye Dermal Sarcoma Virus (WDSV), Zebrafish Endogenous Retrovirus (ZFERV) and Salmon Swim

Bladder Sarcomas (SSSV). It is possible that a large leader peptide size with several repeats may therefore be a typical feature of lower vertebrate retroviruses (Lapierre et al., 1999, Holzschu et al., 1985, Shen and Steiner, 2004, Paul et al., 2006)

The *gag* and *pol* genes reside in the same reading frame, whereas the *env* is in a different reading frame (Fig 2 panel lll). The genomic organization of ZFERV-2 provirus appears to be similar to that of ZFERV, MuLV and WDSV.

The size of *gag* gene is 1,702bp which includes motif for Major Homology Region (MHR), KQGAIEVEETEEDKKQRQL in Capsid region (Wills and Craven, 1991). However, motifs for Matrix and Nucleocapsid were not found during several BLAST searches of Protein Databases (Fig 1).

```
ZFERV     GESASAYLHRCEAEWEDRTGENPDSSDICKEFFRQAVIKGVPASAVVAIENSPDMQGG
ZFERV-2   GESASAYLHRCEAEWEDRTGENPDSSDICKEFFRQAVIKGVPASAVAAIENSPDMQGG

ZFERV     EGVVWTRHFIHHVDKAVERQGKEETEVEKLKTQLLKMQIEAEKEKGKKKKENLQLPVL
ZFERV-2   EGEVWTRHFIHHV???????????????????????????????????????????????

ZFERV     TGPGVEGARSPVHNIHPISGHINSPHGIPDSYPYGPSPNWPPSGNQNQNPVMQGCFGC
ZFERV-2   ????????????????????????????????????????????????????????????

ZFERV    GAQDHWKRECPYGGQRGQPPARGRAPGRGGPQGGRGRGAPRNVSWYSNQQVPRQSPVW
ZFERV-2   ????????????????????????????????????????????????????????????

ZFERV     GPLDHADYXRGPQRTSEGALAEPLLTILVDNQPVQALVDTGATFSTIQRQMMEKDSLS
ZFERV-2   ????????????????????????????????????????????????????????????

ZFERV     TQSEEVRGFSGETEKWPLTKPLRVQVAGQTLLHSFLCSANVPSALLGRDLLVKLGVKI
ZFERV-2   ????????????????????????????????????????????????????????????

ZFERV     LCHPEGLIIIFPNGLTTNCSTPVTTTVRGQWVLQADKTVKARCYWLKLTDKTKLQEVI
ZFERV-2   ????????????????????????????????????????????????????????????

ZFERV     TKWEPWLNTLHKFQTPTDPWHCTLLYDISEDEEYEQNFSEVLGRSTSIKCTKLFVARE
ZFERV-2   ????????????????????????????????????????????????????????????

ZFERV     GVAAEIILTEEQEGYFEMTETSTPHVTIFMNEGHEAKALGPMVKRMQKVGDWGKTDNE
ZFERV-2   ????????????????????????????????????????????????????????????

ZFERV     WEYSQALDGYRLTIDIAEEVIYQMVEVRRIVTTQEGDGEGAERLLQEMPESLWSKAAG
ZFERV-2   ??????????????????????????????????????????????????????????

ZFERV     DVGKWSIEPVLFQIDRTEIVNVRQYKLRPEAVEGIGETIKELEAAEVLRRTVSGWNTP
ZFERV-2   ???????EPVLFQIDRTEIVNVRQYKLRPEAVEGIGETIEELEAAEVLRRTVSDWNTP

ZFERV     ILPVLKKTTGKYRMVHDLRLINEKVLTATLPTPNPYTIMSKLTPKHSHFTCIDLANAF
ZFERV-2   ILPVLKKTTGKYRMVHDLRLINEKVLTATLPTPNPYTIMSKLTPKHSHFTCIDLANAF

ZFERV     FCMPLAEQCQGIFAFSYQGAQYTYNRLPQGFILSPGLFNQALRELLDSCTLHEGTIVI
ZFERV-2   FCMPLAEQCQDIFAFSYQGAQYTYNRLPQGFILSPGLFNQALRELLDSCTLHEGTIVI

ZFERV     QYVDDLLLAAHSNEVCLQDTRKVLTLLSTAGLKVSKEKIQISRATVHFLGRIIGQTGT
ZFERV-2   QYVDDLLLAAHSNEVCLQDTRKVLTLLSTAGLKVSKEKIQISRATVHFLGRIIGQTGT
```

Fig 3. Comparison of protease and reverse transcriptase genes between ZFERV and ZFERV-2. Note that the missing 516 amino acids of the protease gene were deleted and marked by the question mark symbol (?).

The size of *pol* gene is 3,443bp, which includes noticeable motif for Reverse Transcriptase, RNAseH and Integrase genes. Despite having all the necessary protein for polymerase, ZFERV-2 lacks motif for protease which could be observed in a huge deletion comprising 516 amino acids that are symbolized as question marks (Fig. 2). This is the only apparent difference between ZFERV and ZFERV-2.

The size of *env* gene is 1,934bp which includes motifs for Surface Protein and Transmembrane. Interestingly, within the Transmembrane protein, an immunosuppressive motif (QNRLALDMLLSERGGVCSMFK) which is observed mainly in lymphotrophic viruses such as betaretrovirus, deltavirus, lentivirus and certain MLV-related (gammaretrovirus) has been found in ZFERV-2(Fig 1). (Cianciolo et al., 1985, Sonigo et al., 1986)

The genomic organisation and characterisation of ZFERV-2 indicate a typical retrovirus endogenous from the lower vertebrates especially from the piscine host. In addition, the *gag* and *pol* genes resemble viruses from other genera of retroviruses. However, the presence of immunosuppressive motif in *env* gene may indicate the probability that ZFERV-2 was infectious in the past. Further investigation on the phylogenetic analyses of the viral genes needs to be carried out to establish the relationship of ZFERV-2 with other genera of retroviruses.

References

1. Cianciolo, G.J., Copeland, T.D., Oroszlan, S., Snyderman, R.: Inhibition of Lymphocyte Proliferation by a Synthetic Peptide Homologous to Retroviral Envelope Proteins. Science 230, 453–455 (1985)
2. Herbs, H., Sauter, M., Muller-Lantzsch, N.: Expression of Human Endogenous Retrovirus K Elements in Germ Cell and Trophoblastic Tumors. Am. J. Pathol. 149, 1727–1735 (1986)
3. Herniou, E., Martin, J., Miller, K., Cook, J., Wilkinson, M., Tristem, M.: Retroviral Diversity and Distribution in Vertebrates. J. Virol. 72, 5955–5966 (1998)
4. Holzschu, D.L., Martineou, D., Fodor, S.K., Bowser, P.R., Casey, J.M.: Nucleotide Sequence and Protein Analysis of a Complex Piscine Retrovirus Walleye Dermal Sarcoma Virus. J. Virol. 72, 5320–5331 (1985)
5. Johnson, P.M., Lyden, T.W., Mwenda, J.M.: Endogenous Retroviral Expression in Human Placenta. Am J. Reprod. Immunol. 23, 115–120 (1990)
6. Kambol, R.: Distribution and Evolution of Endogenous Retroviruses within Amphibian and Piscine Hosts. Unpublished PhD thesis. Department of Biological Sciences, Imperial College London, University of London, UK (2003)
7. Kambol, R., Kabat., P., Tristem., M.: Complete Nucleotide Sequence of an Endogenous Retrovirus from the Amphibian, Xenopus Laevis. Virology 311, 1–6 (2003)
8. Lapierre, L.A., Holzschu, D.L., Bowser, P.R., Casey, J.W.: Sequence and Transcriptional Analyses of the Fish Retroviruses Walleye Epidermal Hyperplasia Virus Types 1 and 2: Evidence for Gene Duplication. J. Virol. 73, 9393–9403 (1999)
9. Martin, J., Kabat, P., Herniou, E., Tristem, M.: Characterisation and Complete Nucleotide Sequence of an Unusual Reptilian Retrovirus Recovered from the Order Crocodylia. J. Virol. 76, 4651–4654 (2002)
10. Muhlbock, O., Bentvelzen., P.: The Transmission of the Mammary Tumor Viruses. Perspect. Virol. 6, 75–78 (1968)

11. Muir, A., Lever, A., Moffett, A.: Expression of Human Endogenous Retroviruses in the Placenta. An Update. Placenta, 1–10 (2004)
12. Patarca, R., Heseltine, W.A.: A Major Retroviral Core Protein Related to EPA and TMP. Nature 318, 390 (1983)
13. Paul, T.A., Quakenbush, S.L., Sutton, C., Casey, R.N., Bowser, P.R., Casey, W.: Identification and Characterization of an Exogenous Retrovirus from Atlantic Salmon Swim Bladder Sarcomas. J. Virol. 80, 2941–2948 (2006)
14. Ponferrada, V.G., Mauck, B.S., Wooley, D.P.: The Envelope Glycoprotein of Human Endogenous HERV-W Induces Cellular Resistance to Spleen Necrosis Virus. Arch Virol. 148, 659–675 (2003)
15. Portis, J.L.: Perspectives on the Role of Endogenous Retroviruses in Autoimmune Diseases. Virology 296, 1–5 (2002)
16. Shen, C.H., Steiner, L.A.: Genome Structure and Thymic Expression of an Endogenous Retrovirus in Zebrafish. J. Virol. 78, 899–911 (2004)
17. Sonigo, P., Barker, C., Hunter, E., Wain-Hobson, S.: Nucleotide Sequence of Mason Pfizer Monkey Virus: An Immunosuppressive D-Type Retroviruses. Cell 45, 385–437 (1986)
18. Tristem, M.: Identification and Characterisation of Novel Human Endogenous Retrovirus Families by Screening of the Human Genome Mapping Project Database. J. Virol. 74, 3715–3739 (2000)
19. Wang-Johannig, F.W., Frost, A.R., Johannig, G.L., Khazaeli, M.B., Lobuglio, A.F., Shaw, D.R.: Expression of Human Endogenous Retrovirus K Envelope Transcripts in Human Breast Cancer. Clin. Cancer Res. 7, 1553–1660 (2001)
20. Wills, J.W., Craven, R.C.: Form, Function and Use of Retroviral Gag Proteins. AIDS 5, 639–654 (1991)

Bayesian Phylogeny on Grid

Richard C. van der Wath[1], Elizabeth van der Wath[1], Antonio Carapelli[2], Francesco Nardi[2], Francesco Frati[2], Luciano Milanesi[3], and Pietro Lió[1]

[1]The Computer Laboratory, University of Cambridge, William Gates Building, 15 JJ Thomson Avenue, Cambridge CB3 0FD, UK
[2]Department of Evolutionary Biology, University of Siena, via A.Moro 2, 53100, Siena, Italy
[3]CNR-ITB, Via F.lli Cervi 93, 20090, Segrate, Italy

Abstract. Grid computing defines the combination of computers or clusters of computers across networks, like the internet, to form a distributed supercomputer. This infrastructure allows scientists to process complex and time consuming computations in parallel on demand. Phylogenetic inference for large data sets of DNA/protein sequences is known to be computationally intensive and could greatly benefit from this parallel supercomputing approach. Bayesian algorithms allows the estimation of important parameters on species divergence modus and time but at the price of running repetitive long series of MonteCarlo simulations. As part of the BioinfoGrid project, we ported parallel MrBayes to the EGEE (Enabling Grids for E-sciencE) grid infrastructure. As case study we investigate both a challenging dataset of arthropod phylogeny and the most appropriate model of amino acid replacement for that data set. Our aim is to resolve the position of basal hexapod lineages with respect to Insecta and Crustacea. In this effort, a new matrix of protein change was derived from the dataset itself, and its performance compared with other currently used models.

1 Introduction

Due to large data sets and accompanied large number of parameters being produced by high throughput techniques, it became necessary to develop high performance computers based on clustering technologies and high performance distributed platforms. Grid infrastructures are based on a distributed computing model where easy access to large geographical computing and data management resources is provided to large multi disciplinary VOs (Virtual Organizations). A VO comprises of a sampling Grid users sharing similar requirements and interests who are able to share resources and/or work collaboratively with the other members within the same grouping. In order to submit jobs to the Grid, all users are required to be a member of a VO, this helps to ensure the integrity of the data stored on the grid network. The distributed HPC (high performance computer) is considered as the way to realize the concept of virtual places where scientists and researchers work together to solve complex problems in Bioinformatics, despite their geographic and organizational boundaries. BioinfoGRID (Bioinformatics

Grid Application for life science) is a project funded by the European Union within the Sixth Framework Programme for Research and Technological Development(FP6). The BioinfoGRID project is associated with the Biomedical VO, which consists of a large number of biomedical scientists working in several different fields and countries under one banner.

DNA and amino acid sequences contain both the information of the phylogenetic relationships among species and of the evolutionary processes that have caused the sequences to diverge [1,2,33,34,35,36]. The statistical and computational methods try to detect this information to determine how and why DNA and protein molecules work the way they do. Arthropoda (insects, crustaceans and their kins) account for more than 80% of described animal species, and display an extraordinary diversity in terms of morphology and lifestyle adaptations. This diversity, as well as the age of the major taxa, have considerably complicated our possibility to reconstruct their phylogenetic relationships, which are still debated either among and within major lineages [20]. Molecular phylogeny has recently contributed extensively to this issue, and large data sets of mitochondrial sequences are now available for analysis. The mitochondrial genome is a closed circular molecule, inherited via the maternal line. The organization of the genome is quite simple, and its gene content is highly conserved across multicellular animals [27]: 13 protein coding genes (PCGs), 22 transfer RNA (tRNA) genes, and 2 ribosomal RNA (rRNA) genes. Mitochondrial sequences are commonly used as a standard benchmark for testing evolutionary models and systematic methodology (see for instance [6,21,22,3]). Several authors have found differences in phylogenies that result from different data sets on the same species ([5,4,7,8,23]). Because of these problems in reconstructing phylogenies, it is important to develop appropriate evolutionary models and investigate the performances of different existing models for specific phylogenetic problems.

Here we investigate the Grid setting as infrastructure for biomedical applications. In specific the utilisation of parallel MrBayes to resolve the phylogeny of Pancrustacea whilst testing different models of evolution. The article's framework comprises of three main sections starting with the Methodology Section which constitutes of the following: models of evolution, Bayesian phylogeny and Grid computing. Statistics of different models of evolution and visualization of the posterior probabilities of phylogenetic trees are reported as main results under the Results and Discussion Section. Lastly we summarise our findings towards Grid performance and future expectations in the Conclusion Section. For a comprehensive presentation of the underlying biological problem and a more general discussion of the biological relevance of the results obtained, we refer to the accompanying paper [3].

2 Methodology

2.1 Generating Models of Evolution

The process of phylogeny reconstruction requires 4 steps. The first step comprises sequence selection and alignment to determine site-by-site DNA or amino acid differences. This was done by using the data set described in [3]. The thirteen PCGs (protein coding genes) of one-hundred species of Pancrustacea for which the complete mitochondrial genome sequence is available were aligned and concatenated. After removal of taxa whose mt-sequences possess molecular features known to negatively affect phylogenetic reconstruction, such as extreme compositional bias, accelerated rates of nucleotide substitutions, and gene order rearrangements that involve a change in the strand where one or more PCGs are encoded [28,29,3], a data set with 81 pancrustacean sequences (ingroup) and 5 non-pancrustacean arthropod species (outgroup) was retained and analysed. The second step is to build a mathematical model describing the evolution in time of the sequences. Usually theory is based on continuous time models. We need to derive the instantaneous probabilities of the transition from one amino acid to another. The probabilities of a model can be generated empirically using properties calculated through comparisons of observed sequences or parameterically using chemical and biological properties of DNA and amino acids. Such models permit estimation of the genetic distance between two homologous sequences, measured by the expected number of nucleotide substitutions per site that have occurred on the evolutionary lineages between them and their most recent common ancestor. Such distances may be represented as branch lengths in a phylogenetic tree; the extant sequences form the tips of the tree, while the ancestral sequences form the internal nodes and are generally not known. The third step involves applying an appropriate statistical method to find the tree topology and branch lengths that best describe the phylogenetic relationships of the sequences. One of the most important methods is that of maximum likelihood (ML) which is also a necessary ingredient of the Bayesian approach [36,33,34,35]. The likelihood (L_H) of a hypothesis (H) is equal to the probability of observing the data if that hypothesis were correct. The observed data is again usually taken to be the alignment, although it would of course be more reasonable to say that the sequences are what have been observed and the alignment should then be inferred along with the phylogeny. The statistical method of ML chooses amongst hypotheses by selecting the one which maximizes the likelihood; that is, which renders the data the most plausible. In the context of molecular phylogenetics, a model of nucleotide or amino acid replacement permits the calculation of the likelihood for any possible combination of tree topology and branch lengths [1,30,32]. The topology and branch lengths that maximize this likelihood (or, equivalently, its natural logarithm, lnL_H, which is almost invariably used to give a more manageable number) are the ML estimates. Any parameters with values not explicitly specified by the replacement model can be simultaneously estimated, again by selecting the values that maximize the likelihood. The fourth step consists of the interpretation of results. When properties shared by a set of sequences are too subtle or hidden to be analytically represented (or there are too many degrees of freedom), amino acid replacement models should be obtained through an empirical approach. MtPan, a new model of amino acid replacement in Pancrustacea, was constructed using relative rates of estimated amino acid replacement from pairwise comparisons of

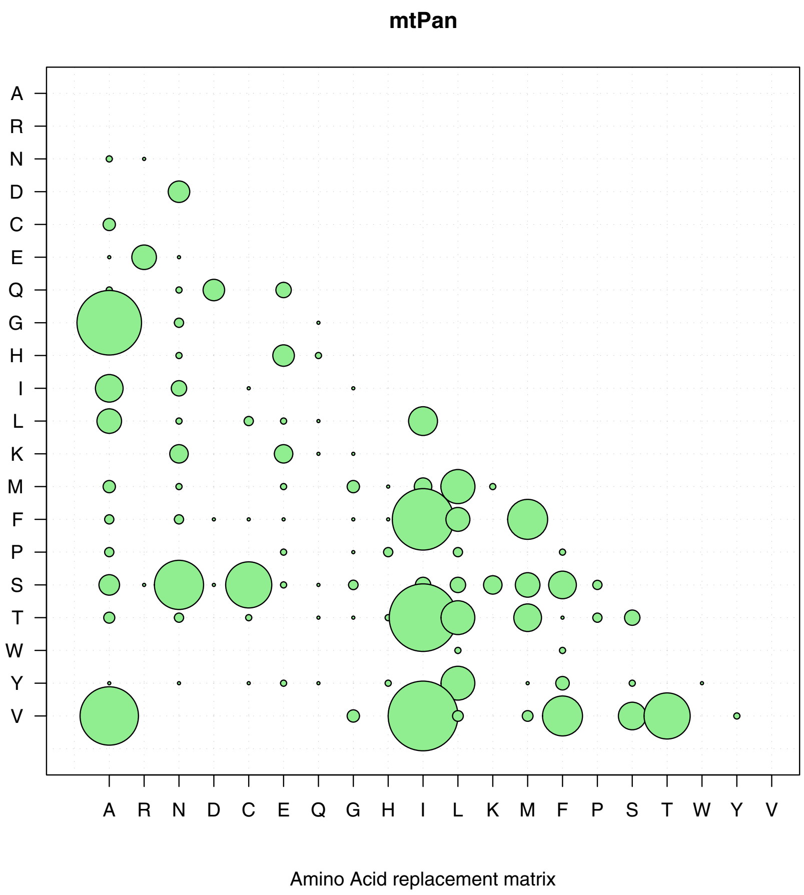


Fig. 1. MtPan, a new model of amino acid replacement in Pancrustacea

sequences in the Pancrustacea dataset, maintained by us, that are 85% or more identical, see Figure 1.

The estimates of the relative rates of amino acid replacement was computed by examining the database and recording the number of times that amino acid type i is observed in one sequence and type j is observed at the corresponding site in a closely related sequence. Interestingly the new model of evolution shows high probability values for transition G-A, V-A, I-V, F-I, T-I with respect all the other possible transitions in mtPan and in the other two models of evolution. These differences reflect the particular environment of and evolutionary trend of mitochondrial proteins in the species we have considered.

2.2 Bayesian Phylogeny

In Bayesian statistics the goal is to obtain a full probability distribution over all possible parameter values. This so-called posterior probability distribution requires the combining of likelihood with the prior probability distribution [9,10,11]. The prior probability distribution shows your beliefs about the parameters before seeing any data. Often, prior information about an unknown parameter may not be available. In such cases, standard non-informative prior distributions, i.e., probability distributions which contain little or no information about the parameters are used, resulting in posterior distributions that are dominated by the likelihood. In phylogeny it is very common that biologists have some strong beliefs about the relationships of some deep branches. In other cases there are theories from shared organs/apparatus or developmental pathways which may suggest alteration of the prior. In Bayesian phylogeny the parameters are of the same kind as in maximum likelihood phylogeny where typical parameters include tree topology, branch lengths, nucleotide frequencies and substitution model parameters [12,13,14,15,16]. The main objective in maximum likelihood however, is to determine the best point estimates of parameter values while Bayesian phylogeny aims to calculate a full probability distribution over all possible parameter values.

If a target distribution has multiple peaks, separated by low valleys, a Markov chain may have difficulty in moving from one peak to another. As a result, the chain might get stuck on one peak and the resulting samples will not approximate the posterior density correctly. This is a serious practical concern for phylogeny reconstruction as multiple local peaks are known to exist in the tree space during heuristic tree search under maximum parsimony, maximum likelihood and minimum evolution criteria. The same can be expected for stochastic tree search using MCMC. Many strategies have been proposed to improve mixing of Markov chains in presence of multiple local peaks in the posterior density. One of the most successful algorithms is the Metropolis-coupled MCMC; [11,31]. Parallel MrBayes the program implements this variant of MCMC called "Metropolis-Coupled Markov Chain Monte Carlo"(MCMCMC)[17]. In this algorithm, m chains are run in parallel, on as many or less processors with different stationary distributions where all but one of them are heated. Heating increases the state acceptance probability of chains resulting in the more eager crossing of valleys in a landscape of probability trees. State swapping is attempted among randomly appointed chains which provides better integration and if such swapping is successful it results in the exploration of other peaks and limits the seclusion to local maximums.

2.3 State of Art, Potentialities of Grid Computing for Phylogeny

The European Commission-funded "Enabling Grids for E-sciencE" (EGEE)[19] project brings together scientists and engineers from more than 240 institutions in 45 countries world-wide to provide a seamless international Grid infrastructure for e-Science that is available to scientists 24 hours-a-day. Here we give some details in order to favour the usage of this important resource. The EGEE Grid consists of 41,000 CPUs in addition to about 5 PB disk (5 million Gigabytes) + tape MSS of storage, and maintains 100,000 concurrent jobs. Having such resources available changes the way scientific research can take place, and could significantly increase our possibility to analyse complex datasets using methodologically more correct, though computationally more intense, methodologies. Middleware is a key component to any grid computing effort for it serves as the communication layer enabling interaction across hardware and network environments. Early in the project the LCG(Large Hadron Collider Computing Project) middleware stack was used on the EGEE infrastructure. Most of this stack was later developed and re-engineered into the current middleware solution, gLite. The gLite Grid services follow a Service Oriented Architecture facilitating compliance with upcoming Grid standards and is currently widely deployed on hundreds of sites as part of the EGEE project, enabling global science in a great number of disciplines [26].

WMProxy is a new component, implemented as a Web service, to access the gLite Workload Management System (WMS) and efficiently handles large number of job submission requests and controls.

Job submission requires a description of the job to be executed and a description of the needed resources. These descriptions are provided with a high-level language called JDL. The Job Description Language (JDL) is based on Condor classified advertisements (classads) for describing jobs and aggregates of jobs such as MPICH which is a high-performance and widely portable implementation of MPI(Message Passing Interface).

Actual job submission is done by calling in sequence the two service operations jobRegister and jobStart. When a jobRegister request arrives at the WMProxy, if the client has the rights to proceed, a set of specific attributes needed by the WMS for handling the request appropriately together with a generated unnique Job identifier are inserted in to the job description. The requesting user is then mapped to a local user by means of LCMAP, which provides authorization functionalities based on VOMS (Virtual Organization Membership Services),resulting in the job, local directories and files being created with appropriate ownership and permissions. When all the aforementioned steps have been successfully completed, the job with the generated job identifier and the enriched JDL description is registered to the RB(Resource Broker). From that point on the job is uniquely identified and can be monitored throughout the system and the various job states with its identifier [25].

Bioinformatics usually entails the execution of very complex workflow analysis. Some applications can perform and scale very well in a Grid environment while others are instead better suited for a dedicated cluster especially when bound to certain license agreements or when specialized supporting software is required. Current results suggests the grid also better suited for more computationally expensive jobs as job submission times often exceeds execution times of small jobs.

An obvious disadvantage of the MCMCMC algorithm is that m chains are run and only one chain is used for inference. For this reason, MCMCMC is ideally suited for implementation on parallel machines, since each chain will in general require the same amount of computation per iteration. The EGEE grid infrastructure with its uncontested parallel capacity provides a very favorable environment for Bayesian MCMCMC analysis allowing a user to combine several simultaneous simulations varying crucial variables such as models of evolution, number of generations and amount of chains. We embedded Parallel MrBayes into the BioMed virtual organisation of the EGEE grid infrastructure as part of the BioinfoGrid project [18].

We utilised this framework to infer phylogenetic analysis on the Pancrustacea dataset by submitting numerous jobs testing three different models of evolution. Firstly the general matrix available for mitochondrial genomes, but based on vertebrate taxa, MtRev [24] secondly MtArt [23] and lastly our specifically developed matrix, MtPan. One million generations were run, varying between 8 or 4 MC-chains, while trees were sampled every 100 generations. Three nexus files was constructed for the batch processing of MrBayes, two stating the GTR model as the rate matrix and setting the prior for the substitution rates in accordance with the MtArt and MtPan matrices and the third, by setting the amino acid model prior variable to MtRev.

Many BioMed VO Grid WNs(worker nodes) are not currently correctly configured to execute MPI(parallel) jobs successfully even though the jdl requirements variable specifies the desired prerequisites to which the selected CEs (Computing Elements) should adhear. This is due to the fact that the Grid infrastructure and maintanance of contributed resources are still evolving. We randomly submitted 12 similar parallel MrBayes jobs (not to specific CEs) and disappointingly 9 of these jobs failed. For our main analysis we submitted only to selected CEs, which consists of hundreds of WNs, proven to have very good MPICH success rates.

3 Results and Discussion

As mentioned, we submitted only to a handful of CEs which has proven to be able to successfully execute MPICH job types. The Resource Broker algorithmically selects among this list for the most suited CE. Of our 45 MtArt 8-chain jobs submitted, 15 failed to reach the running state. The running state refers to the status of a job which has passed successfully through the queueing system to being actually processed by the amount of worker nodes specified.

After several concurrent submissions we achieved a total of 30 8-chain jobs and 20 4-chain jobs for each of the three evolutionary models MtArt, MtRev and MtPan. The average mrbayes execution time was around 29 hours while the average submission time (the time it takes the job to successfully reach the actual running state from the moment the jobs was submitted to the Resource Broker) was around 17 hours. This resulted in total average execution times of around 45 hours for one 8-chain analysis. Thirty 8-chain and twenty 4-chain jobs for each of the three models were run in parallel. This amounts to a total of 960 worker nodes being occupied, in parallel, for an average period of two days. One 4-chain sequential execution, with the same parameters as used in the main analysis, failed to complete within the 7 day grid proxy validity period. This implies that sequentially done, taking the average running time of one chain as 45 hours, this analysis would have taken up to 960*2 days. See Figure 2 for a

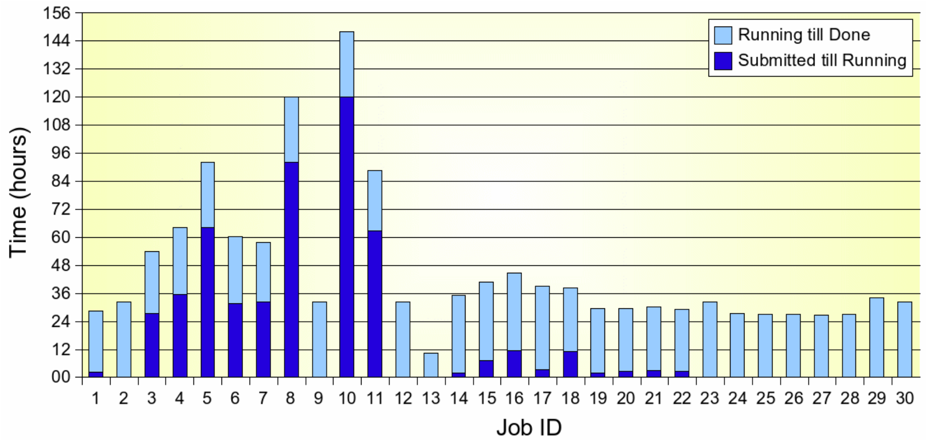


Fig. 2. Grid job submission and execution times for MtArt 8-chain

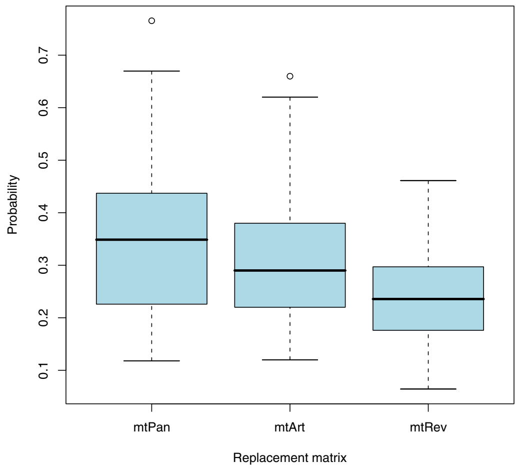


Fig. 3. Box-and-whisker plot of the posterior probabilities obtained using the three different models

depicted summary of the execution and submission times for the 30 successful Mtart 8-chain jobs.

In the analysis of our Pancrustacean mitochondrial data set, plots of likelihood versus generations, together with the value of the likelihood towards which each run converges, were used to assess the efficiency of the analysis to explore the likelihood space and reach the best maximum, and the relative performance of the three amino acid substitution matrices.

Most of the runs, regardless of the matrix used, converged to slightly different maxima. This indicates, on one hand, that the resulting topology for each run is highly dependent on the performance of the algorithm to explore the likelihood surface and the starting point of the search, thus suggesting prudence when interpreting the results. On the other hand, this underlines the importance of conducting different parallel runs and comparing the results in order to have a global outlook on these aspects of the analysis.

Furthermore, comparing the actual topologies to which each run converges, it becomes evident that while most of the shallow nodes are common to most resulting trees, the deepest nodes tend to vary, and the difference in likelihood observed across runs, little as they are, depend on rearrangements at the deepest nodes. This behaviour is most likely due to the intrinsic signal of the data, rather than to limitations of the analysis. It also has a detrimental effect on the biological interpretation of the results, as the most interesting nodes are in fact the ones that connect the major lineages of the Pancrustacea, and these are in turn the less stable. Nevertheless, the tree with the highest posterior probabilities selected (Figure 5) is largely congruent with that obtained in a previous analysis [3], and has a precise taxonomic meaning. It shows, in fact, that two hexapod lineages, Collembola and Diplura, do not cluster with the remaining insects, whose closest relatives appear to be in certain lineages of crustaceans (Stomatopoda, Decapoda, Cephalocarida).

The only difference between this tree and the tree obtained in [3] is the relative position of a cluster of four taxa (Pachypsilla, Trialeurodes, Xenos, Armillifer) which was embedded in the clade of Insecta in [3], while is here joined with the basalmost lineages of the whole tree, and clustered with the Remipedia and the Maxillopoda. This cluster indeed joins species from distant taxonomic lineages: Homoptera (insects), Strepsiptera (insects) and Pentastomida. Their association must therefore be considered an artifact of the analysis, possibly related with an accelerated rate of evolution of these sequences.

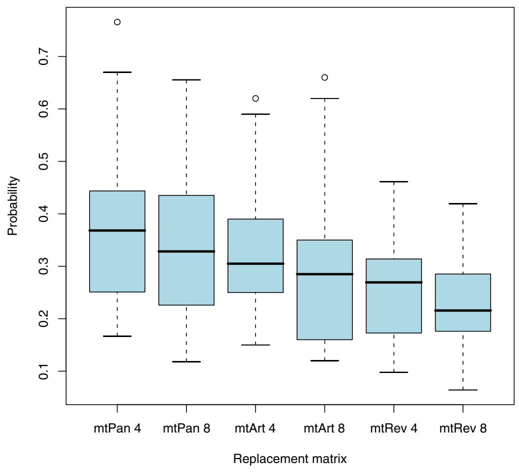


Fig. 4. Box-and-whisker plot of the posterior probabilities obtained using the three different models, showing the two MCMCMC implementations of each model separately. The 4 chain versions tended to slightly outperform the 8 chain versions.

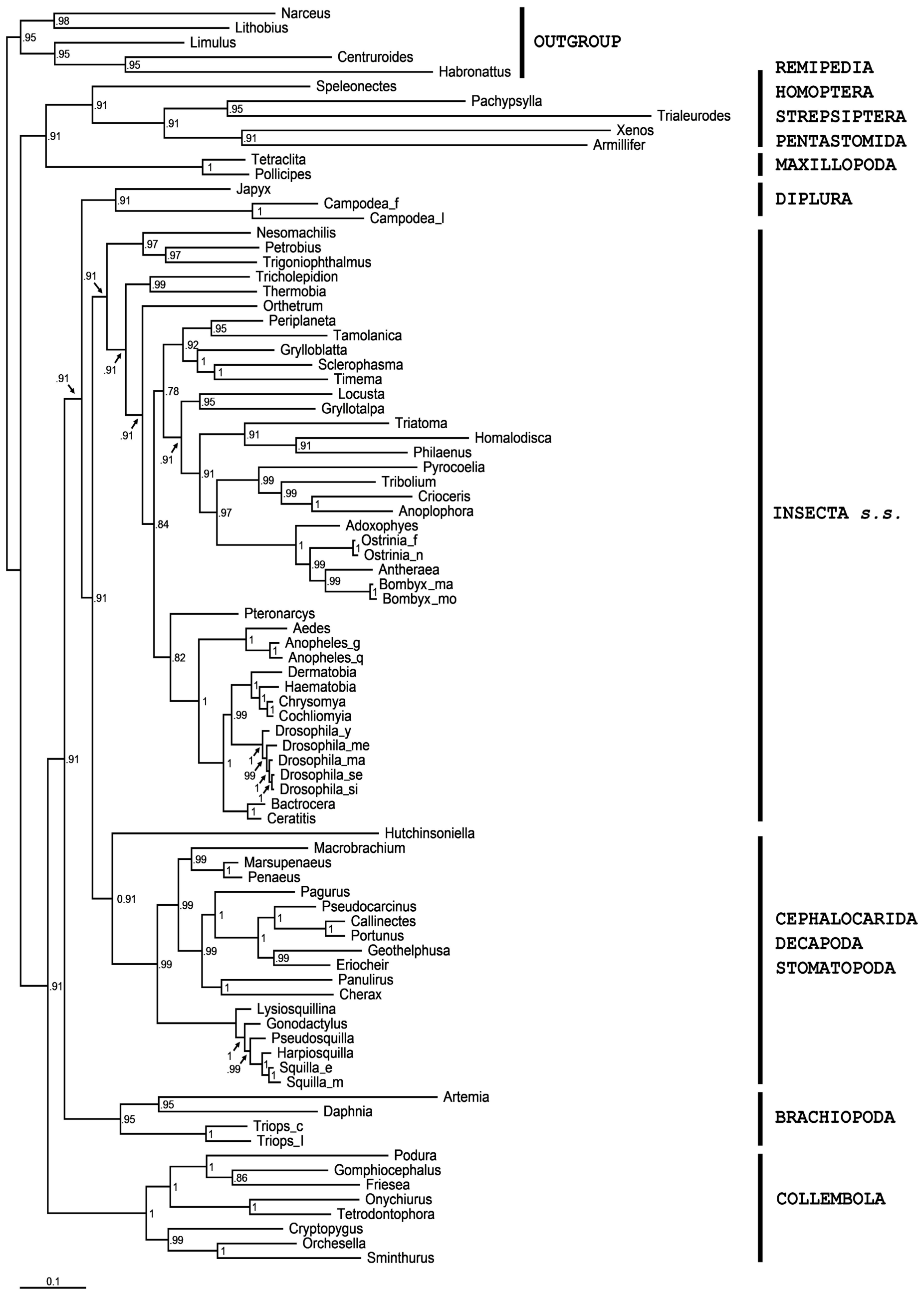


Fig. 5. The tree with the highest posterior probability of all the simulations

The highest posterior probabilities from the 1 million runs (excluding burn-in) of each job was analysed separately for each matrix. From this multiple analysis it was possible to show (Figure 3 and 4) that the MtPan matrix generally converge to higher likelihood values than MtRev and MtArt. This does not come unexpected, as this matrix was directly derived from a collection of pancrustacean mitochondrial genes, and is likely to describe the mechanism of sequence evolution in these groups better than matrices developed for different taxonomic groups [23]. From Figure 4 it can be seen that the tree with the highest overall posterior was found during an MtPan 4-chain run. The consensus tree as produced by MrBayes for this run is shown in Figure 5 and is considered as our best estimate of phylogenetic relationships among Pancrustacea based on the data set analysed.

4 Conclusion

Here we present a new model of evolution for pancrustacean mitochondrial PCGs which is performing in average better than currently available models. From a biological standpoint, this analysis has confirmed that mitochondrial genome sequences unambiguously indicate the reciprocal paraphyly of the formal taxa Hexapoda and Crustacea, as traditionally defined [6,21,22,3], therefore implying a new interpretation of the evolution of the most successful lineage of Metazoa. We found that inching towards better models of evolution may require intensive computation. Models may be improved by adding more taxa from which parameters are estimated, and methods of phylogenetic reconstruction may be improved by taking into account the inevitable presence of different rates of evolution in different lineages of the same phylogenetic tree. The Grid is becoming "the" resource for solving large-scale computing applications in Bioinformatics, system biology and computational medicine. The distributed high performance computer (HPC) and GRID are "de facto" considered as the way to realize the concept of virtual places where scientists and researchers work together to solve complex scientific problems, despite their geographic and organizational boundaries. In phylogenetic inference the grid will become a computational laboratory to test models of evolution and phylogenetic hypothesis on large trees which may provide an effective boost in our investigation of the evolutionary process.

References

1. Whelan, S., Lió, P., Goldman, N.: Molecular phylogenetics: State-of-art methods for looking into the past. Trends Genet. 17, 262–272 (2001)
2. Lió, P., Goldman, N.: Models of molecular evolution and phylogeny. Genome Res. 8, 1233–1244 (1998)
3. Carapelli, A., Lió, P., Nardi, F., van der Wath, E., Frati, F.: Phylogenetic analysis of mitochondrial protein coding genes confirms the reciprocal paraphyly of Hexapoda and Crustacea. BMC Evolutionary Biology 7 (2007), doi:10.1186/1471-2148-7-S2-S8

4. Russo, C.A., Takezaki, N., Nei, M.: Efficiencies of different genes and different tree-building methods in recovering a known vertebrate phylogeny. Mol. Biol. Evol. 13, 933–942 (1996)
5. Zardoya, R., Meyer, A.: Phylogenetic performance of mitochondrial protein-coding genes in resolving relationships among vertebrates. Molecular Biology and Evolution 13, 525–536 (1996)
6. Pollock, D.D., Eisen, J.A., Doggett, N.A., Cummings, M.P.: A case for the evolutionary genomics and the comprehensive examination of sequence biodiversity. Mol. Biol. Evol. 17, 1776–1778 (2000)
7. Cao, Y., Janke, A., Waddell, P.J., Westerman, M., Takenaka, O., Murata, S., Okada, N., Paabo, S., Hasegawa, M.: Conflict among individual mitochondrial proteins in resolving the phylogeny of eutherian orders. J. Mol. Evol. 47, 307–322 (1998)
8. Lió, P.: Phylogenetic and structural analysis of mitochondrial complex I proteins. Gene 345, 55–64 (1999)
9. Liu, J.S., Lawrence, C.E.: Bayesian inference on biopolymer models. Bioinformatics 15, 38–52 (1999)
10. Shoemaker, J.S., Painter, I.S., Weir, B.: Bayesian statistics in genetics: a guide for the uninitiated. Trends Genet. 15, 354–358 (1999)
11. Larget, B., Simon, D.: Markov chain Monte Carlo algorithms for the Bayesian analysis of phylogenetic trees. Mol. Biol. E 16, 750–759 (1999)
12. Huelsenbeck, J.P., Ronquist, F.: MrBayes: Bayesian inference in phylogenetic trees. Bioinformatics 17, 754–755 (2001)
13. Ronquist, F., Huelsenbeck., J.P.: MrBayes3: Bayesian phylogenetic inference under mixed models. Bioinformatics 19, 1572–1574 (2003)
14. Rannala, B., Yang., Z.: Bayes estimation of species divergence times and ancestral population sizes using DNA sequences from multiple loci. Genetics 164, 1645–1656 (2003)
15. Mau, B., Newton, M.A., Larget, B.: Bayesian phylogenetic inference via Markov chain Monte Carlo methods. Biometrics 55, 1–12 (1999)
16. Yang, Z., Rannala, B.: Bayesian phylogenetic inference using DNA sequences: Markov chain Monte Carlo methods. Mol. Biol. Evol. 14, 717–724 (1997)
17. Altekar1, G., Dwarkadas1, S., Huelsenbeck, J.P., Ronquist3, F.: Parallel Metropolis coupled Markov chain Monte Carlo for Bayesian phylogenetic inference. Bioinformatics 20, 407–415 (2004)
18. http://www.bioinfogrid.eu/
19. http://public.eu-egee.org/
20. Richter, S.: The Tetraconata concept: hexapod-crustacean relationships and the phylogeny of Crustacea. Org. Divers Evol. 2, 217–237 (2002)
21. Nardi, F., Spinsanti, G., Boore, J.L., Carapelli, A., Dallai, R., Frati, F.: Hexapod origins: monophyletic or polyphyletic? Science 299, 1887–1889 (2003)
22. Cook, C.E., Yue, Q., Akam, M.: Mitochondrial genomes suggest that hexapods and crustaceans are mutually paraphyletic. Proc. R Soc. Lond. B 272, 1295–1304 (2005)
23. Abascal, F., Posada, D., Zardoya, R.: MtArt: a new model of amino acid replacement for Arthropoda. Mol. Biol. Evol. 24, 1–5 (2007)
24. Yang, Z., Nielsen, R., Hasegawa: Models of amino acid substitutions and applications to mitochondrial protein evolution. Mol. Biol. Evol. 15, 1600–1611 (1998)
25. http://trinity.datamat.it/projects/EGEE/wiki/
26. http://public.eu-egee.org/industry/ifdocuments/glite-flyer.pdf

27. Boore, J.: Animal mitochondrial genomes. Nucl. Acid Res. 27, 1767–1780 (1999)
28. Cameron, S.L., Miller, K.B., DaHaese, C.A., Whiting, M.F., Barker, S.C.: Mitochondrial genome data alone are not enough to unambiguosly resolve the relationships of Entognatha, Insecta and Crustacea sensu lato (Arthropoda). Cladistics 20, 534–557 (2004)
29. Hassanin, A., Lger, N., Deutsch, J.: Evidence for multiple reversals of asymmetric mutational constraints during the evolution of the mitochondrial genome of Metazoa, and consequences for phylogenetic inferences. Syt. Biol. 54, 277–298 (2005)
30. Chor, B., Hendy, M.D., Holland, B.R., Penny, D.: Multiple maxima of likelihood in phylogenetic trees: an analytic approach. In: RECOMB 2000, pp. 108–117 (2000)
31. Mossel, E., Vigoda, E.: Limitations of Markov chain Monte Carlo algorithms for Bayesian Inference of phylogeny. Ann. Appl. Probab. 16, 2215–2234 (2006)
32. Chor, B., Tuller, T.: Finding a maximum likelihood tree is hard. J. ACM 53, 722–744 (2006)
33. Gascuel, O.: Mathematics of Evolution and Phylogeny. Oxford University Press, USA (2007)
34. Yang, Z.: Computational Molecular Evolution (Oxford Series in Ecology and Evolution). Oxford University Press, USA (2006)
35. Felsenstein, J.: Inferring Phylogenies Sinauer Associates, 2nd edn (2003)
36. Nielsen, R.: Statistical Methods in Molecular Evolution (Statistics for Biology and Health), 1st edn. Springer, Heidelberg (2005)

Comparison of Exact String Matching Algorithms for Biological Sequences[*]

Petri Kalsi, Hannu Peltola[**], and Jorma Tarhio

Department of Computer Science and Engineering
Helsinki University of Technology
P.O. Box 5400, FI-02015 HUT, Finland
petri.kalsi@iki.fi, {hpeltola,tarhio}@cs.hut.fi

Abstract. Exact matching of single patterns in DNA and amino acid sequences is studied. We performed an extensive experimental comparison of algorithms presented in the literature. In addition, we introduce new variations of earlier algorithms. The results of the comparison show that the new algorithms are efficient in practice.

1 Introduction

String matching is used for locating nucleotide or amino acid sequence patterns in the biological sequence databases. Nucleotide or DNA sequences have an alphabet of four characters, and amino acid sequences have an alphabet of 20 characters, respectively. Because the total number of sequences is rapidly increasing, efficient methods are needed. Recently three novel algorithms, namely Lecroq's 'New' [13], SSABS [16], and TVSBS [19], were proposed for searching single exact pattern in DNA and amino acid sequences. This motivated us to compare these algorithms with older methods known to be efficient. Another reason is advances in the processor and memory technology and in compilers. Old algorithms do not necessarily behave with new processors and compilers in the same way they did ten years ago in our extensive comparison [18]. Besides the comparison in the DNA and amino acid alphabets, we developed new variations of our earlier algorithm [18].

It turned out that SSABS and TVSBS were poor for DNA sequences. The new variations of our algorithm were slightly faster than Lecroq's algorithm in most cases in the both alphabets. The clear winner for amino acid patterns of moderate length was SBNDM2 [6], which has not been specifically designed for biological alphabets.

In terms of computer science, a biological sequence is a string, and therefore we use both terms. The aim of string matching is to report all exact occurrences of a pattern string in a longer text string. We use the following notations throughout the paper. The length of the pattern and the text are m and n, respectively. The size of the alphabet is σ.

[*] Supported by the Academy of Finland.

[**] Corresponding author.

The rest of the paper is organized as follows. We review earlier solutions in Section 2 and introduce the new variations of our algorithm in Section 3. Section 4 reports the results of our experiments before the conclusion in Section 5.

2 Known Solutions for DNA Sequences

Most of the efficient string matching algorithms in the DNA alphabet are modifications of the Boyer–Moore algorithm [3]. Most often only the occurrence heuristic (also called the bad character heuristic) is applied for shifting. Algorithms of this type are greedy in the sense that the pattern is moved forward after the first character mismatch of an alignment is observed. Shifts may then be unnecessarily short for the nucleotide alphabet if shifting is based on a single character. Therefore it is advantageous to apply q-grams, strings of q characters, instead of single characters. This technique was already mentioned in the original paper of Boyer and Moore [3, p. 772], and Knuth [12, p. 341] analyzed theoretically its gain. Zhu and Takaoka [21] presented the first algorithm utilizing the idea. Their algorithm uses two characters for indexing a two dimensional array. They gave also another version based on hashing. The size of the hash table was two times the pattern length.

Baeza-Yates [1] introduced an extension to the Boyer–Moore–Horspool algorithm [7] where the shift array is indexed with an integer formed from a q-gram with `shift` and `add` instructions. For this kind of approach the practical upper limit with 8 bit characters is two characters.

For the DNA alphabet Kim and Shawe-Taylor [10] introduced a convenient alphabet compression by masking the three lowest bits of ASCII characters. In addition to the `a`, `c`, `g`, and `t` one gets distinguishable codes also for `n` and `u`. Even important control code \\`n`=LF has distinct value, but \\`r`=CR gets the same code as `u`. With this method they were able to use q-grams of up to six characters. Indexing of the shift array was similar to Baeza-Yates' algorithm.

With a small alphabet the probability of an arbitrary q-gram appearing in long pattern is high. This restricts the average shift length. Kim and Shawe-Taylor [10] introduced also an interesting variation for the cases where the q-gram in the text occurs in the pattern. Then two additional characters are checked one by one to achieve a longer shift.

In most cases the q-gram that is taken from the text does not match with the suffix of the pattern, and the pattern can be shifted forward. For efficiency one should use a *ufast* skip loop [8], where the pattern is moved forward until the last q-gram of the pattern matches with a q-gram in the text. The easiest way to implement this idea is to artificially define the shift of the last q-gram of the pattern to be zero. We must also add to the end of the text a copy of the pattern as a stopper. After the skip loop the pattern is compared with the corresponding text positions. An advantage of skip loop is that we use text characters both for preliminary comparison and for computing the shift. So we fetch the characters only once.

Our earlier algorithm [18] applies a skip loop. The key feature of this algorithm is efficient handling of q-grams. The details are presented in Section 3.1. Recently Lecroq [13] presented a related algorithm. Its implementation is based on the Wu–Manber algorithm [20] for multiple string matching, but as suggested above, the idea is older [3,21].

SSABS [16] and TVSBS [19] were developed for biological sequences. SSABS is a Boyer–Moore type algorithm. In the search phase the algorithm verifies that the first and last character of the pattern matches with the current alignment before checking the rest of the alignment (a.k.a. guard tests). TVSBS uses a 2-gram for calculating the shift, adopted from the Berry–Ravindran algorithm [2], which is a cross of the Zhu–Takaoka algorithm and Sunday's QS algorithm [17]. Instead of the two-dimensional shift table of Berry–Ravindran, TVSBS uses a hash function to compute an index to a one-dimensional table.

In Section 4, we will present an experimental comparison of several algorithms. Experimental results on some earlier algorithms for DNA sequences can be found in [18]. We have included three efficient algorithms, namely FAOSO [4], SBNDM [14,15], and SBNDM2 [6] in the comparison, although they have not been designed for biological sequences.

Recently Kim et al. [11] presented an algorithm for packed DNA. We excluded it from the comparison, because there is no fair way to compare other algorithms with it.

3 Variations of Our Algorithm

3.1 Our Earlier Algorithm

Our algorithm [18] is a modification of the Boyer–Moore–Horspool algorithm [7] for the DNA alphabet. Instead of inspecting a single character at each alignment of the pattern, the algorithm reads a q-gram and computes an integer called fingerprint from it. This idea of applying q-grams was already present in the earlier algorithms [1,10,21], but we introduced an efficient way to handle q-grams.

The ASCII codes of `a`, `c`, `g`, and `t` are 97, 99, 103, and 116, respectively. The ASCII codes are mapped to the range of 4: $0 \leq r[x] \leq 3$, where $r[x]$ is the new code of x, such that characters `a`, `c`, `g`, and `t` get different codes and other possible characters get e.g. code 0. In this way the computation is limited to the effective alphabet of four characters. The fingerprint is simply a reversed number of base 4. A separate transformation table h_i is used for each position i of a q-gram and multiplications are incorporated during preprocessing into the tables: $h_i[x] = r[x] \cdot 4^i$. For $q = 4$, the fingerprint of $x_0 \cdots x_3$ is $\sum_{i=0}^{3} r[x_i] \cdot 4^i$ which is then computed as

$$h_0[x_0] + h_1[x_1] + h_2[x_2] + h_3[x_3].$$

The algorithm is given below as BMHq. It corresponds to Algorithm 4 in [18] without unrolling. In the algorithm, $T = t_1 \cdots t_n$ denotes the text and $P = p_1 \cdots p_m$ the pattern. At each alignment of the pattern the last q-gram of the pattern is compared with the corresponding q-gram in the text by testing the equality of their fingerprints. If the fingerprints match, a potential occurrence has been found, which has to be checked. We need to check only the first $m - q$ characters of the pattern because our fingerprint method does not cause hash collisions, assuming that the searched text contains only DNA characters. The algorithm applies a variation of a skip loop called "unrolled fast" (=*ufast*) by Hume and Sunday [8].

Algorithm 1. BMHq$(P = p_1p_2 \cdots p_m, T = t_1t_2 \cdots t_n)$

```
1: Initialize $D[*]$
2: $t_{n+1} \cdots t_{n+m} \leftarrow P$        /* adding stopper */
3: $p \leftarrow f(P, m, q)$
4: $r \leftarrow D[p]$; $D[p] \leftarrow 0$; $k \leftarrow m$
5: $s \leftarrow D[f(T, k, q)]$
6: loop
7:    while $s > 0$ do
8:       $k \leftarrow k + s$
9:       $s \leftarrow D[f(T, k, q)]$
10:   if $j = 0$ then exit
11:   Check the potential occurrence
12:   $s \leftarrow r$
```

In the algorithm, $f(T, k, q)$[1] denotes the fingerprint of $t_{k-q+1} \cdots t_k$. For computation of the shift table D see [18]. The pattern needs to be copied to the end of the text, so that the *ufast* skip loop will end when the search is complete.

Lecroq's 'New' algorithm [13] is closely related to BMHq. However, Lecroq applies a different method based on hashing for computing fingerprints of q-grams. Moreover, the maximal shift of his algorithm is $m - q + 1$, while that of BMHq is m, because BMHq is able to handle all prefixes of the first q-gram of the pattern.

3.2 Variations for DNA

We tested several ways to compute the fingerprint in order to make our algorithm BMH4 faster. When considering 4-grams, they fit into a 32-bit word in 8-bit characters. Some CPU architectures, notably the x86, allow unaligned memory reads of several bytes. This inspired us to try reading several bytes in one instruction, instead of four separate character reads. Reading several bytes at a time is by no means a new technique. Many researchers have used it [5,9]. But we have not seen any comparison with standard bytewise reading. One may argue that it is not fair to apply multiple reading, because all CPU architectures do not support it. But the x86 architecture is nowadays so dominant that it is reasonable to tune algorithms for it.

We will present two variations of BMH4. BMH4b reads a 32-bit word, and BMH4c reads two consecutive halfwords. Because in BMH4b we have access to

[1] This is f2 in [18].

an integer consisting of four characters, it would be inefficient to use the old character-based fingerprint method. The fingerprint calculation arrays of BMH4 are replaced with hashing expression, where the input is a whole 4-gram as a 4-character long integer. ASCII codes for `a`, `c`, `g` and `t` are distinguishable by the last three bits. The following expression packs the unique bits of the four characters together in a few instructions, to form an integer in the range of 2313...16191 = 00100100001001...11111100111111.

```
FP(x) = ((x >> 13) & 0x3838) | (x & 0x0707)
```

Preprocessing of BMH4b is similar to the earlier algorithm, we just calculate the fingerprints of the 4-grams in the pattern with the new hash function. So the main difference between BMH4b and BMH4 is in the computing of f. In BMH4b this is done with masking, shifting, and bitwise OR. Hashing receives the current text location pointer as an argument, and reads the 32-bit integer from that address with `FP(*(k-3))`. $D[x]$ contains the preprocessed shift values for each hashed q-gram of the pattern.

Based on our tests, unaligned memory reads on x86 processors incur a speed penalty of up to 70% when compared with aligned reads. This unfortunately reduces the speed of BMH4b, because 75% of the reads are unaligned on the average. So we made another variation BMH4c, which reads two consecutive halfwords. In the case of BMH4c, only 25% of the reads are unaligned ones getting the speed penalty (while crossing the border of 4 bytes). In BMH4c, the value of a fingerprint is got as $a_1[x_1] + a_2[x_2]$ where x_i is a halfword and a_i a preprocessed transformation table, for $i = 1, 2$.

3.3 Variations for Amino Acids

The q-gram approach is valid also for larger alphabets, although with a larger alphabet the optimal value of q is smaller. A larger alphabet size required only minor changes to the algorithm. A new variation BMH2 was created based on BMHq. The range of the ASCII code mapping $r[x]$ was increased from 4 to 20, to cover the amino acid alphabet "ACDEF GHIKL MNPQR STVWY" instead of the DNA alphabet. Otherwise the algorithm is the same as BMHq.

We made also BMH2c, which reads a 2-gram as a 16-bit halfword. The shift array is indexed directly with halfwords.

These algorithms can be used with any text, e.g. English text. For this kind of data we mapped each character to a smaller range with a modulo function in preprocessing. The best results for English text were obtained with modulo 25. Because the mapping tables are created in the preprocessing phase, the modulo operation does not affect the search time directly.

4 Experimental Results

Algorithms. Among tested algorithms were SSABS [16], TVSBS [19], BMH4, BMH4b, BMH4c, BMH2, and BMH2c. KS by Kim and Shawe-Taylor [10] uses a trie of reversed q-grams of the pattern. The Fast Average Optimal Shift-Or (FAOSO) [4] was tested with different values of the step size and the unrolling factor. The given run times of FAOSO are based on the best possible parameter combination for each pattern length.

SBNDM [15] is based on the BNDM [14] algorithm, with simplified shift calculations. SBNDM2 [6] is a modification of SBNDM. One of the key points of SBNDM2 is loop unrolling, i.e. handling of a 2-gram before entering to the inner loop.

We also tested LEC, which is the 'New' algorithm of Lecroq [13], and which uses q-grams and hashing. The run times of LEC are given based on the best possible q for each pattern length.

SSABS and TVSBS were implemented as described in the articles [16] and [19]. Because TVSBS is based on Berry–Ravindran algorithm [2], we also implemented two versions of the latter, BR and BRX. In BRX we have modified the algorithm to calculate shifts based on the last character of the current text alignment t_s and the next character t_{s+1}, instead of t_{s+1} and t_{s+2} as in BR which corresponds to the original Berry–Ravindran. The probability of a shift of the pattern by one position is approximately $1/\sigma$ for BR and $1/\sigma^2$ for BRX. Thus BRX produces with small alphabets longer shifts than BR on the average. This is due to the inherent weakness of Sunday's QS algorithm [17]: if the last character of the pattern is common, then the probability of a shift of one is high. In such a case, t_{s+2} is often useless in the computation of shift in BR. Our test results show that this weakness of BR is noticeable even with amino acids.

All algorithms were implemented in C. The codes of KS, SBNDM2, FAOSO, and LEC were got from the original authors. Other codes were implemented by us.

Test Setting. The tests were run on a 2.8GHz Pentium D (dual core) CPU with 1 GB of memory. Both cores have 16 KB L1 data cache and 1024 KB L2 cache. The computer was running Fedora 8 Linux.

All the algorithms were tested in a testing framework of Hume and Sunday [8]. All programs were compiled with Intel's C compiler `icc 10.0` producing x86 "32-bit" code and using the optimization level `-O3`.

The test patterns were generated based on the text files so that roughly half of the patterns have matches in the text. These longer patterns were then cut to shorter patterns. Every pattern set contains 200 patterns for DNA and 100 for amino acids of the same length. For DNA, the test data was taken from the genome of the fruit fly. The amino acid text is the peptide sequences of *Arabidopsis thaliana.* All text files consisted of about $2 \cdot 10^6$ characters. Test runs were executed on otherwise unloaded computer. To achieve accurate timings the search for each pattern was repeated 20 times (with DNA and 50 times with amino acids). Reported results are medians for five successive test runs using corresponding pattern set. The change of the process from one processor core to another empties cache memories with various degree. This would slow down reads from memory and induce annoying variation to the timing of test runs. To avoid it we have used Linux function `sched_setaffinity` to bind the process to only one processor or core.

Table 1. Search times in milliseconds for DNA sequences ($\sigma = 4$)

m	BMH4	BMH4C	SSABS	TVSBS	KS	BR	BRX	SBNDM	SBNDM2	FAOSO	LEC$_{3-5}$
4	1004	806	1991	1559	-	1545	1294	1754	1152	$\boxed{601}$	1390
6	678	$\boxed{540}$	1750	1260	1126	1250	984	1246	885	558	762
8	534	429	1612	1073	592	1063	811	984	743	$\boxed{357}$	551
10	447	355	1570	945	411	936	696	806	657	$\boxed{306}$	450
12	391	305	1586	866	324	860	624	692	582	$\boxed{276}$	379
14	351	$\boxed{270}$	1556	802	281	795	573	603	515	278	327
16	317	250	1551	755	241	747	538	538	462	$\boxed{228}$	295
18	286	236	1526	716	$\boxed{212}$	710	506	482	417	235	272
20	257	224	1516	683	$\boxed{196}$	672	477	440	381	203	256
22	240	210	1536	663	$\boxed{179}$	654	462	406	349	205	244
24	232	199	1519	645	$\boxed{170}$	633	442	375	321	227	$\underline{231}$
26	220	191	1491	620	$\boxed{162}$	612	431	348	299	244	220
28	209	182	1526	602	$\boxed{154}$	593	417	327	280	228	209
30	201	178	1529	589	$\boxed{149}$	580	407	308	263	242	198
40	173	156	1492	545	$\boxed{126}$	536	377	-	-	-	168
50	156	144	1481	521	$\boxed{113}$	511	361	-	-	-	150
60	146	135	1432	501	$\boxed{106}$	491	346	-	-	-	139
70	135	126	1495	498	$\boxed{105}$	489	343	-	-	-	129
80	125	112	1551	498	$\boxed{102}$	489	334	-	-	-	$\underline{121}$
90	117	110	1588	494	$\boxed{102}$	483	334	-	-	-	114
100	111	108	1569	502	$\boxed{107}$	491	335	-	-	-	108

DNA sequences. The search times for DNA sequences are listed in Table 1. The reported times do not include preprocessing. For most of the tested algorithms the preprocessing times were less than 1% of search times. The preprocessing time was slightly more for the BMH2c, BMH4b, BMH4c, TVSBS, KS, BR, and BRX algorithms, because they have large memory structures, which are initialized during preprocessing.

FAOSO, KS, and BMH4c were winners of the DNA test. FAOSO was the fastest for $m \leq 16$ excluding $m = 6$ and $m = 14$ where BMH4c was the best. KS was the fastest for $18 \leq m \leq 100$.

Multiple reading is advantageous, because BMH4c was approximately 20% faster than BMH4 for short patterns. The times of BMH4b (not shown) were between those of BMH4 and BMH4c for patterns longer than 25 nucleotides; for shorter patterns is was a little slower than BMH4. Even BMH2c (not shown) was among the best for patterns shorter than 10.

TVSBS proves to be an improvement on SSABS for long patterns, but its performance still falls in line with the other algorithms of the comparison. However, the test results show that SSABS and TVSBS are not appropriate for DNA sequences. The original Berry–Ravindran algorithm was slightly faster

Table 2. Search times in milliseconds for amino acid sequences ($\sigma = 20$)

m	BMH2	BMH2C	SSABS	TVSBS	BR	BRX	SBNDM	SBNDM2	FAOSO	LEC$_{3-4}$
4	320	277	326	336	316	288	346	241	$\boxed{164}$	622
6	225	195	252	260	246	214	294	138	$\boxed{123}$	340
8	280	156	207	213	206	172	250	104	$\boxed{101}$	245
10	152	133	186	187	177	149	219	$\boxed{84}$	91	201
12	131	118	173	170	161	133	196	$\boxed{74}$	79	166
14	119	106	155	148	142	119	175	$\boxed{67}$	79	142
16	112	98	145	137	132	109	157	$\boxed{62}$	75	128
18	107	91	137	128	121	101	141	$\boxed{59}$	69	118
20	101	85	132	120	115	96	127	$\boxed{55}$	70	112
22	94	81	127	116	109	90	116	$\boxed{53}$	70	105
24	90	77	122	107	103	85	106	$\boxed{51}$	68	99
26	86	74	118	102	98	82	97	$\boxed{50}$	69	93
28	83	71	115	98	93	79	90	$\boxed{49}$	69	89
30	80	69	114	96	93	77	84	$\boxed{48}$	68	86
40	72	$\boxed{62}$	103	81	79	67	-	-	-	74
50	67	$\boxed{58}$	98	74	71	61	-	-	-	67
60	63	$\boxed{55}$	95	69	67	58	-	-	-	62
70	58	$\boxed{51}$	92	63	61	54	-	-	-	59
80	53	$\boxed{48}$	92	58	57	51	-	-	-	55
90	51	$\boxed{48}$	91	56	53	50	-	-	-	52
100	51	$\boxed{49}$	91	54	52	$\boxed{49}$	-	-	-	50

than TVSBS. The results show that the shift modification in BRX is a clear improvement on BR for DNA sequences.

Amino acid sequences. The search times in the amino acid alphabet in Table 2 are more even. BMH2c works efficiently for all tested pattern lengths, but SBNDM2 is the overall winner of this test. Because the bitvectors were 32 bits long, BMH2c and BRX dominate when $m > 32$. Performance of BMH2 is close to BR and BRX for all tested pattern lengths.

Additional notes. Further optimizations might improve the search times of TVSBS, especially the same one we used in BRX. With 64-bit bitvectors FAOSO, SBNDM, and SBNDM2 can handle longer patterns than in these tests. On separate test with "64-bit code", SBDNM2 was the fastest also for pattern lengths up to 64. Currently we have only 32-bit version of FAOSO, so its performance with longer bitvectors remained unknown.

It was a surprise that KS was the winner for long patterns, because it was slower in our earlier test [18] and also in the preliminary test with an older compiler for this paper. The improvement of KS is due to new compilers. With `gcc 4.1` the performance is nearly the same as in Table 1, but when compiled with `gcc 4.0.1`, `3.4.4`, or `3.3.5` the search times are at least 12% longer on several computers tested, and even 100% on the set of the shortest patterns.

We repeated the same tests on another computer having two 2.0GHz AMD Opteron DP 246 processors each having 6KB L1 cache and 1MB L2 cache. Size of main memory is 6GB (400MHz DDR PC3200). Unfortunately there is a unusually large variation in timings on that computer so that it was difficult to get reliable results. On tests with the DNA data, BMH4 was good for short patterns. KS was good for patterns longer than 15. BMH4c was the fastest for $m \leq 22$. On amino acids BMH2c was superior for all tried pattern sets. For most algorithms, the relative speed improvement was clearly smaller than in the tests reported above, when patterns got longer.

Although FAOSO was fast for short patterns, it is rather unpractical. Namely it has two constant parameters and it is a tedious process to find out the best combination of them for each type of input. Without varying both of them, one will likely get slower run times.

5 Conclusion

We demonstrated that it is possible to speed up q-gram calculation by reading several characters at the same time. According to the comparison, the new variations BMH4c and BMH2c were among the fastest ones for DNA and amino acid data, respectively. Our intention is to continue working on multiple reading. We expect that multiple reading would speed up also other algorithms handling continuous q-grams, like BR, LEC, and SBNDM2. KS is promising for searching of long DNA patterns on modern processors while using an up-to-date compiler.

We showed how to fix the inefficiency of the Berry–Ravindran algorithm in the case of DNA data. This modification is advantageous also for amino acid data.

We noticed that the results depend more on the processor and compiler than we expected. Algorithm $\mathcal{A}$ may be faster than algorithm $\mathcal{B}$ on computer $\mathcal{C}$ and the other way around on computer $\mathcal{D}$. For more profound practical evaluation of algorithms, several environments should be tried.

References

1. Baeza-Yates, R.: Improved string searching. Software: Practice and Experience 19(3), 257–271 (1989)
2. Berry, T., Ravindran, S.: A fast string matching algorithm and experimental results. Proc. of the Prague Stringology Club Workshop 1999, Czech Technical University, Prague, Czech Republic, Collaborative Report DC-99-05, pp. 16–28 (1999)
3. Boyer, R.S., Moore, J S.: A fast string searching algorithm. Communications of the ACM 20(10), 762–772 (1977), http://en.wikipedia.org/wiki/J_Strother_Moore
4. Fredriksson, K., Grabowski, Sz.: Practical and optimal string matching. In: Consens, M.P., Navarro, G. (eds.) SPIRE 2005. LNCS, vol. 3772, pp. 376–387. Springer, Heidelberg (2005)
5. Fredriksson, K.: Personal communication

6. Holub, J., Ďurian, B.: Fast variants of bit parallel approach to suffix automata (Unpublished Lecture) University of Haifa 04-05 (2005)
7. Horspool, R.N.: Practical fast searching in strings. Software: Practice and Experience 10(6), 501–506 (1980)
8. Hume, A., Sunday, D.: Fast string searching. Software: Practice and Experience 21(11), 1221–1248 (1991)
9. Hyyrö, H.: Personal communication
10. Kim, J.Y., Shawe-Taylor, J.: Fast string matching using an n-gram algorithm. Software: Practice and Experience 24(1), 79–88 (1994)
11. Kim, J.W., Kim, E., Park, K.: Fast matching method for DNA sequences. In: Chen, B., Paterson, M., Zhang, G. (eds.) ESCAPE 2007. LNCS, vol. 4614, pp. 271–281. Springer, Heidelberg (2007)
12. Knuth, D.E., Morris, J.H., Pratt, V.R.: Fast pattern matching in strings. SIAM Journal on Computing 6(1), 323–350 (1977)
13. Lecroq, T.: Fast exact string matching algorithms. Information Processing Letters 102(6), 229–235 (2007)
14. Navarro, G., Raffinot, M.: Fast and flexible string matching by combining bit-parallelism and suffix automata. ACM Journal of Experimental Algorithms 5(4), 1–36 (2000)
15. Peltola, H., Tarhio, J.: Alternative algorithms for bit-parallel string matching. In: Nascimento, M.A., de Moura, E.S., Oliveira, A.L. (eds.) SPIRE 2003. LNCS, vol. 2857, pp. 80–93. Springer, Heidelberg (2003)
16. Sheik, S.S., Aggarwal, S.K., Poddar, A., Balakrishnan, N., Sekar, K.: A FAST pattern matching algorithm. J. Chem. Inf. Comput. Sci. 44(4), 1251–1256 (2004)
17. Sunday, D.M.: A very fast substring search algorithm. Communications of the ACM 33(8), 132–142 (1990)
18. Tarhio, J., Peltola, H.: String matching in the DNA alphabet. Software: Practice and Experience 27(7), 851–861 (1997)
19. Thathoo, R., Virmani, A., Sai Lakshmi, S., Balakrishnan, N., Sekar, K.: TVSBS: A fast exact pattern matching algorithm for biological sequences. Current Science 91(1), 47–53 (2006)
20. Wu, S., Manber, U.: A fast algorithm for multi-pattern searching, Report TR-94-17, Department of Computer Science, University of Arizona, Tucson, AZ (1994)
21. Zhu, R.F., Takaoka, T.: On improving the average case of the Boyer–Moore string matching algorithm. Journal of Information Processing 10(3), 173–177 (1987)

A New Approximation Algorithm for the Minimum Fragment Removal Problem

Hatem Tahri and Mourad Elloumi

Research Unit of Technologies of Information and Communication,
Higher School of Sciences and Technologies of Tunis
5, Avenue Taha Hussein, Monfleury 1008 Tunis, Tunisia
Hatem.Tahri@isetb.rnu.tn, Mourad.Elloumi@fsegt.rnu.tn

Abstract. In this paper, we present a new approximation *Minimum Fragment Removal* (MFR) algorithm based on a new *Optimization Vertex Bipartization* (OVB) one. Our approximation MFR algorithm is of complexity $O(n^2)$ in computing time, where n is the number of the vertices of the *conflict graph*. Then, we present the results obtained after running the program corresponding to our algorithm on random graphs and on graphs from Computational Molecular Biology.

Keywords: Graphs, haplotype assembly, minimum fragment removal, vertex bipartization, algorithms, complexities.

1 Introduction

A *DesoxyriboNucleic Acid* (DNA) macromolecule is a double-stranded one. Each strand is a sequence of molecules, called *nucleotides*. There are four types of nucleotides : *Adenine*, *Cytosine*, *Guanine* and *Thymine* coded, respectively, by *A*, *C*, *G* and *T*. Each DNA macromolecule is packed separately in *a chromosome*. In *diploid* organisms such as human, there are two almost identical copies of each chromosome in an individual : one copy inherited from the individual's father, called *paternal chromosome* and one copy inherited from the individual's mother, called *maternal chromosome*. And consequently, in diploid organisms, there are two almost identical copies of each DNA macromolecule. Each copy of a DNA macromolecule has regions which are variable from one individual to another. These variations are called *genetic polymorphisms*. Thus, a *Single Nucleotide Polymorphism* (SNP) is a variation of a single nucleotide in a DNA macromolecule. A *haplotype* is a list of SNPs in a copy of a DNA macromolecule. The pairing between the paternal haplotype and the maternal one represents the *genotype* of an individual.

The *Haplotype Assembly Problem* (HAP), also called *single individual haplotyping problem*, is defined as follows: Given a set of strings $F=\{f_1, f_2, \dots, f_n\}$ coding overlapping fragments coming from two copies of a DNA macromolecule and $S=\{1, 2, \dots\}$ a set of integers representing the numbers of SNPs appearing in strings of F, find the errors existing in F and remove/correct these errors such that the obtained strings can be partitioned in two classes, H_1 and H_2, where the elements of each class are not *in conflict* among them selves, i.e., for any couple of strings (f_i, f_j) of $H_1 \times H_1$ or of $H_2 \times H_2$, there does not exist SNPs indexed in S whose value in f_i is different from the one in f_j.

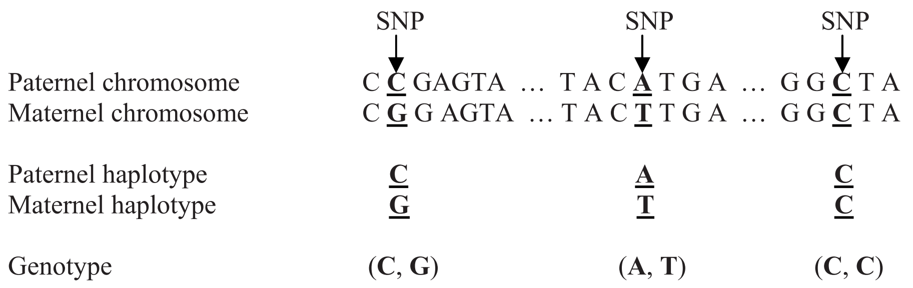

SNP
SNP
SNP
Paternel chromosome C C GAGTA ... T A C A T G A ... G G C T A
Maternel chromosome C G G AGTA ... T A C T T G A ... G G C T A
Paternel haplotype C A C
Maternel haplotype G T C
Genotype (C, G) (A, T) (C, C)

Fig. 1. Haplotypes and Genotype of an individual

When some strings represent *errors* because they code bad fragments, the HAP boils down to the *Minimum Fragment Removal* (MFR) problem [1], [2], [3], [4], [5], [6], [7], [8], [9]:

Find the minimum number of strings of F to remove, so that, the remaining strings can be partitioned in two classes, H_1 and H_2, where the elements of each class are not *in conflict* among them selves.

The MFR problem is polynomial when the *SNP matrix* is *gapless* [6], it is NP-hard when each row (fragment) of the SNP matrix has, at most, one *gap* [1] and it is APX-hard in the general case [6], i.e., there does not exist *good* approximation algorithms for this problem. Among the algorithms that deal with the MFR problem, we mention [6], [7], [10]. The algorithm described in [6] is a *dynamic programming* one [16], [17], it is of complexity $O(2^{2k}m^2n+2^{3k}n^3)$ in computing time, where m and n are respectively the number of rows and the number of columns of the *SNP matrix*, and k is the upper bound of the sum of the gaps lengths in a row in this matrix. Thus, when k=0, i.e., the rows of the *SNP matrix* are *gapless*, the algorithm becomes of complexity $O(m^2n+n^3)$ in computing time. The algorithm described in [7] is a fast approximation one. Finally, the algorithm described in [10] is a *Branch and Bound* (BB) one [18], it is exponential in computing time.

In this paper, we present a new approximation MFR algorithm based on a new *Optimization Vertex Bipartization* (OVB) one. Indeed, the MFR problem boils down to the OVB one [13], [14], [15]. In fact, if G=(V,E) is the *undirected graph* where V is F and E is the set of pairs $\{f_i,f_j\}$, such that $\{f_i,f_j\}\in F\times F$ and f_i is in conflict with f_j, then finding the minimum number of strings of F to remove so that the remaining strings can be partitioned in two classes, H_1 and H_2, where the elements of each class are not in conflict among them selves boils down to finding the minimum number of vertices in G to remove in order to obtain a graph $G'=(V',E')$, $V'\subseteq V$ et $E'\subseteq E'$, such that G' is *bipartite*, i.e., we can partition the set of vertices of V' in two disjoint classes V'_1 and V'_2 such that E' contains neither edges connecting vertices in V'_1 nor edges connecting vertices in V'_2.

Our approximation MFR algorithm is of complexity $O(n^2)$ in computing time, where n is the number of vertices of the conflict graph G associated with the set of strings $F=\{f_1,f_2, \dots ,f_n\}$ and the set of integers $S=\{1, 2, \dots \}$.

The rest of this paper is organized as follows: In the second section, we give some definitions and notations. In the third section, we propose and prove the theorems on which our approximation MFR algorithm is based. In the fourth section, we describe approximation MFR algorithm. In the fifth section, we present the experimental results obtained after running the corresponding program on graphs from Computational Molecular Biology and random graphs. Finally, in the last section, we present our conclusion.

2 Definitions and Notations

Let f_i and f_j be two strings coding overlapping fragments coming from two copies of a DNA macromolecule, we say that f_i and f_j are *in conflict* if and only if there exists a SNP indexed in S whose value in f_i is different from the one in f_j.

An *undirected graph* G is a couple (V,E) made-up by two sets V and E where the elements of V are called *vertices* and those of E are pairs of elements of V called *edges*. If $u_i=\{x_i,y_i\}\in E$ then x_i and y_i are called *endpoints* of u_i and vertices x_i and y_i are *adjacent*. In this case, the *degree* of a vertex x_i is the number of the vertices which are adjacent to it. An *isolated vertex* in G is a vertex with a null degree. The set of isolated vertices of G will be denoted by $\zeta(G)$. A graph $G'=(V',E')$ is a *subgraph* of a graph $G=(V,E)$, if and only if, we have $V'\subseteq V$ and $E'\subseteq E$. We denote by $G\backslash V'$ the subgraph of G obtained by removing from G the vertices of V' and the edges that have endpoints in V'. A graph $G=(V,E)$ is *bipartite*, if and only if, we can partition the set of vertices V in two disjoints classes V_1 and V_2 such that E contains neither edges connecting vertices in V_1 nor edges connecting vertices in V_2. The classes V_1 and V_2 are called *partitions* of the bipartite graph. A *stable set* associated with a graph $G=(V,E)$ is a subset of V, whose elements are pairwise nonadjacent.

Let $F=\{f_1, f_2, \dots, f_n\}$ be a set of strings coding overlapping fragments coming from two copies of a DNA macromolecule and $S=\{1, 2, \dots\}$ a set of integers representing the numbers of SNPs appearing in strings of F the *conflict graph* $G=(V,E)$ associated with F and S is an undirected graph where a vertex i, $1\leq i\leq n$, of V represents a fragment f_i of F and an edge $\{i, j\}$, $1\leq i, j\leq n$, expresses the fact that f_i is in conflict with f_j.

In the following section, we propose and prove the theorems on which our approximation MFR algorithm is based.

3 Fundamental Aspects

Our approximation MFR algorithm is based on the following theorems.

Theorem 1: Let $G=(V,E)$ be an undirected graph and:

(i) $V_1 = \zeta(G)$.

(ii) V_2 be a stable set of $G\backslash V_1$.

(iii) $V_3 = \zeta(G\backslash(V_1\cup V_2))$

(iv) And $V_4 = V\backslash(V_1\cup V_2\cup V_3)$

for any couple (i,j), $1\leq i,j\leq 4$, we have $V_i\cap V_j = \emptyset$ and we have $V = V_1\cup V_2\cup V_3\cup V_4$.

Proof:

(*i*) Let us prove first that for any couple (i,j), $1\leq i,j\leq 4$, we have $V_i \cap V_j = \varnothing$:

(*i.i*) Let's suppose that there exists a vertex $v \in V_4$. Then, $v \in V\backslash(V_1\cup V_2\cup V_3)$, hence $v\notin V_1\cup V_2\cup V_3$. Therefore, for any i, $1\leq i\leq 3$, we have $V_i\cap V_4=\varnothing$.

(*i.ii*) Let's suppose that there exists a vertex $v \in V_3$. Then, $v\in \zeta(G\backslash(V_1\cup V_2))$, hence $v\notin V_1\cup V_2$ and therefore, for any i, $1\leq i\leq 2$, we have $V_i\cap V_3=\varnothing$.

(*i.iii*) Finally; let's suppose that there exists a vertex $v\in V_2$. Then, v is a vertex in $G\backslash V_1$, hence $v\notin V_1$ and therefore, we have $V_1\cap V_2=\varnothing$.

Hence, for any couple (i,j), $1\leq i,j\leq 4$, we have $V_i\cap V_j=\varnothing$.

(*ii*) According to the definitions of V_1, V_2, V_3 and V_4, it is trivial that we have $V=V_1\cup V_2\cup V_3\cup V_4$ •

Corollary 1: Let $G=(V,E)$ be an undirected graph and V_1, V_2 and V_3 be the subsets of V verifying properties (*i*), (*ii*) and (*iii*) defined in Theorem 1, If $V_2 = \varnothing$ then $V_3 = \varnothing$.

Proof: Indeed, **l**et's suppose that $V_2 = \varnothing$ and $V_3 \neq \varnothing$. According to the definition of V_1, the graph $G\backslash V_1$ is without isolated vertices. Thus, if V_2 is a stable set of $G\backslash V_1$ and $V_3=\zeta(G\backslash(V_1\cup V_2))$, then for any vertex $u\in V_3$, there exists at least a vertex $v\in V_2$, such that $(u,v)\in E$. Hence $V_2\neq\varnothing$, this contradicts the hypothesis. •

Theorem 2: Let $G=(V,E)$ be an undirected graph, $U = V_1\cup V_2$ and F be a subset of E such that $F=\{(u_i,v_{3,j})$ such that $(u_i,v_{3,j})\in(U\times V_3)\}$, where V_1, V_2 and V_3 are the subsets of V verifying properties (*i*), (*ii*) and (*iii*) defined in Theorem 1, with $V_2\neq\varnothing$ and $V_3\neq\varnothing$. The graph $G_1=(U\cup V_3,F)$ is bipartite.

Proof: We have $F=\{(u_i\ v_{3,j})$ such that $(u_i,v_{3,j})\in(U\times V_3)\}$, $U=V_1\cup V_2$, $V_2\neq\varnothing$ and $V_3\neq\varnothing\}$. We have also, for any vertex $v_{3,i}$ of V_3 there exists at least one vertex v of V such that $(v_{3,i},v)\in E$ and $v\in V_2$, i.e., V_3 is the subset of vertices of V that are adjacent only to the vertices of V_2. Indeed, since V_2 and V_3 are the subsets of V verifying properties (*ii*) and (*iii*) defined in Theorem 1 then according to Theorem 1 we have $V_2\cap V_3 = \varnothing$. Hence $G_1=(U\cup V_3,F)$ is bipartite. •

Theorem 3: Let $G=(V,E)$ be an undirected graph and V_1, V_2 and V_3 be the subsets of V verifying properties (*i*), (*ii*) and (*iii*) defined in Theorem 1. The graph $G'=(V_1\cup V_2 \cup V_3\cup\sigma(G\backslash(V_1\cup V_2\cup V_3)),E')$ such that $E'=\{(u,v)$ such that $(u,v)\in((V_1\cup V_2)\times(V_3\cup\sigma(G\backslash(V_1\cup V_2\cup V_3))))$, $V_2\neq\varnothing$ and $V_3\cup\sigma(G\backslash(V_1\cup V_2\cup V_3))\neq\varnothing\}$, where $\sigma(G\backslash(V_1\cup V_2\cup V_3))$ is a stable set of $G\backslash(V_1\cup V_2\cup V_3)$, is a bipartite one.

Proof: Indeed, by feeding the subset V_3 by vertices of the subset $\sigma(G\backslash(V_1\cup V_2\cup V_3))$, the subset V_3 remains stable since V_3 is the subset of vertices of V that are adjacent only to vertices of V_2. Therefore the graph $G'=(V_1\cup V_2\cup V_3\cup\sigma(G\backslash(V_1\cup V_2\cup V_3)),E')$ such that $E'=\{(u,v)$ such that $(u,v)\in((V_1\cup V_2)\times(V_3\cup\sigma(G\backslash(V_1\cup V_2\cup V_3))))$, $V_2\neq\varnothing$, $V_3\cup\sigma(G\backslash(V_1\cup V_2\cup V_3))\neq\varnothing\}$ is a bipartite one. •

Corollary 2: Let $G=(V,E)$ be an undirected graph and V_4 be the subset of V verifying property (*iv*) defined in Theorem 1. The minimum number of vertices to be removed to bipartize G is at most equal to $|V_4|$.

Proof: Indeed, according to theorem 3, to bipartize an undirected graph $G=(V,E)$ it suffices to remove vertices from the subset V_4. In the worst case, we remove all the vertices of V_4. •

In the next section, we present our approximation MFR algorithm.

4 Algorithm

Let $F=\{f_1, f_2, \dots, f_n\}$ be a set of strings coding overlapping fragments coming from two copies of a DNA macromolecule and $S=\{1, 2, \dots\}$ be a set of integers representing the numbers of SNPs appearing in strings of F, by adopting our approximation MFR algorithm, *ApproxMFR*, we operate as follows :

During the first step, we construct the conflict graph $G=(V,E)$ associated with F and S.

During the second step, we construct V_1, the subset of isolated vertices of the conflict graph G.

During the third step, we construct V_2, a stable set of the graph $G\backslash V_1$.

During the fourth step, we construct V_3, the subset of isolated vertices of the graph $G\backslash V_1 \cup V_2$.

During the fifth step, we construct $V_4 = V\backslash(V_1 \cup V_2 \cup V_3)$.

Finally, during the last step, we add to the subset V_3 the elements of a stable set of $G\backslash(V_1 \cup V_2 \cup V_3)$, i.e. the subgraph made up by the vertices of V_4. The partitions $V_1 \cup V_2$ and V_3 thus obtained are considered to be a solution to the MFR problem.

In the next subparagraphs, we present our algorithms of construction of the subsets V_1, V_2, V_3 and V_4.

4.1 Construction of V_1, V_2, V_3 and V_4

The subset V_1 is the subset of isolated vertices of G. Thus, to construct it suffices to visit all the vertices of G.

The subset V_2 is a stable set of $G\backslash V_1$. The construction of V_2 can be done *via* a simple algorithm, that we call *Stable_Set*, described in a number of papers [11], [12]: let $G=(V,E)$ be an undirected graph, by using *Stable_Set*, we operate as fellows:

First, we set $V_2=\varnothing$ and sort the vertices according to their decreasing degrees by using *QuickSort* algorithm [19]. Then, during each iteration, we choose a vertex v of minimum degree in G, we add v to V_2 and delete v and all of its neighbours from G. We repeat this process until no vertices remain in G.

Once the subset V_2 has been constructed, the constructions of the subsets V_3 and V_4 are trivial. Indeed, the subset V_3 is the subset of isolated vertices of $G\backslash(V_1 \cup V_2)$ and $V_4=V\backslash(V_1 \cup V_2 \cup V_3)$.

4.2 Construction of a Bipartite Graph

Our approximation MFR algorithm, *ApproxMFR*, takes as input $F=\{f_1, f_2, \dots, f_n\}$ a set of strings coding overlapping fragments coming from two copies of a DNA macromolecule and $S=\{1, 2, \dots\}$ a set of integers representing the numbers of SNPs appearing in strings of F and gives as output a subgraph $G_b=(V_1\cup V_2\cup V_3,E_b)$ of G, where E_b is the set of edges that remain after removing $V\backslash(V_1\cup V_2\cup V_3)$, that is bipartite.

Algorithm. *ApproxMFR* (input : $F=\{f_1, f_2, \dots, f_n\}$, $S=\{1, 2, \dots\}$, output : $G_b=(V_1\cup V_2\cup V_3,E_b)$)

```
(i) Construct the conflict graph $G = (V,E)$ associated with $F$ and $S$
(ii) $V_1 := \zeta(G)$                      // Construction of the set of isolated vertices of $G$
(iii) $V_2 :=$ Stable_Set $(G\backslash V_1)$   // Construction of a stable set of $G\backslash V_1$
(iv) $V_3 := \zeta(G\backslash V_1\cup V_2)$     // Construction of the set of isolated vertices of $G\backslash V_1\cup V_2$
(v) $V_4 := V\backslash(V_1\cup V_2\cup V_3)$              // Construction of $V_4$
(vi) $U :=$ Stable_Set $(G\backslash(V_1\cup V_2\cup V_3))$   // Construction of a stable set of $G\backslash(V_1\cup V_2\cup V_3)$.
     $V_3 := V_3 \cup U$                       // Feeding $V_3$ by the vertices of a stable set of $G\backslash(V_1\cup V_2\cup V_3)$
```

Proposition 1: Let $F=\{f_1,f_2, \dots ,f_n\}$ be a set of strings coding overlapping fragments coming from two copies of a DNA macromolecule and $S=\{1, 2, \dots\}$ be a set of integers representing the numbers of SNPs appearing in strings of F, algorithm *ApproxMFR* constructs a subgraph $G_b=(V_1\cup V_2\cup V_3,E_b)$ of G that is bipartite.

Proof: Indeed:
During step (i), we construct the conflict graph $G = (V,E)$ associated with F and S.

During steps (ii), (iii), (iv) and (v), we construct respectively the subsets V_1, V_2, V_3 and V_4 of V verifying properties (i), (ii), (iii) and (iv) defined in Theorem 1. And during step (vi), we construct a stable set U of the graph $G\backslash(V_1\cup V_2\cup V_3)$. Hence, according to theorem 3, the graph $G(V_1\cup V_2\cup V_3\cup U,E')$ such that $E'=\{(u,v)$ such that $(u,v)\in((V_1\cup V_2)\times(V_3\cup U))$, $V_2\neq\emptyset$, $V_3\cup U\neq\emptyset\}$) is bipartite. •

Proposition 2: Algorithm *ApproxMFR* is of complexity $O(n^2)$ in computing time, where n is the number of vertices of the graph G.

Proof: Steps (i) and (ii) are respectively of complexities $O(n^2)$ and $O(n)$.

Step (iii) is of complexity $O(n \log n)$. Indeed, since *QuickSort* is $O(n \log n)$ in computing time, the first step of *Stable_Set* [11], [12] is also of complexity $O(n \log n)$ in computing time. The second step is $O(n)$. Hence, step (iii) is of complexity $O(n \log n)$. Finally steps (iv), (v) and (vi) are respectively of complexities $O(n)$, $O(1)$ and $O(n \log n)$.

Hence, algorithm *ApproxMFR* is of complexity $O(n^2)$ in computing time. •

5 Experimental Results

The program corresponding to our approximation MFR algorithm, *ApproxMFR*, is coded in Microsoft Visual Basic 6.0 and implemented on a 3 Ghz Pentium-4 machine, running Microsoft Windows XP professional with 2 Gb of RAM.

We have used a rate θ in order to compare the number of the removed vertices n_p to the total number of the vertices n. This rate is defined by:

$$\theta = \frac{n\rho}{n} * 100 \qquad (1)$$

In the histogram and the curves below the *density d'* of the graph is defined by:

$$d' = \left\lceil \frac{\Delta(G)}{(n-1)} * 100 \right\rceil \qquad (2)$$

where $\Delta(G)$ is the maximum degree of a vertex in the graph G.

The curves of figure Fig. 2 represent the results obtained for random graphs with parameters n ranging from 10 to 500 and d' ranging from 10 to 50%.

The histogram of figure Fig. 3 represents the results obtained for graphs from Computational Molecular Biology. These graphs were generated using a model of Panconesi and Sozio [7], with the length of a haplotype is equal to 100, the value of the rate between the Hamming distance between H_1 and H_2 and the size of H_1 $(=H_2)$ is equal to 0.2, the number of fragments per haplotype is equal to 20, each bit of every fragment is flipped with a probability equal to 0.02 and the *coverage*, i.e. the number of the fragments that cover the same position of the haplotype, varying from 5 to 10. That is, by using Panconesi and Sozio [7] notation, with parameters n=100, d=0.2, k=20, p=0.02 and c varying from 5 to 10.

All the presented results have been obtained through computing an average on 50 draws.

According to figure Fig. 2, we notice that for random graphs, when the density d' increases the rate θ increases too. We can explain this as follows : When the density d' increases the maximum degree $\Delta(G)$ of a vertex increases too. And hence, the maximum size of a stable set of the graph decreases. This implies that the number of the vertices to be removed increases.

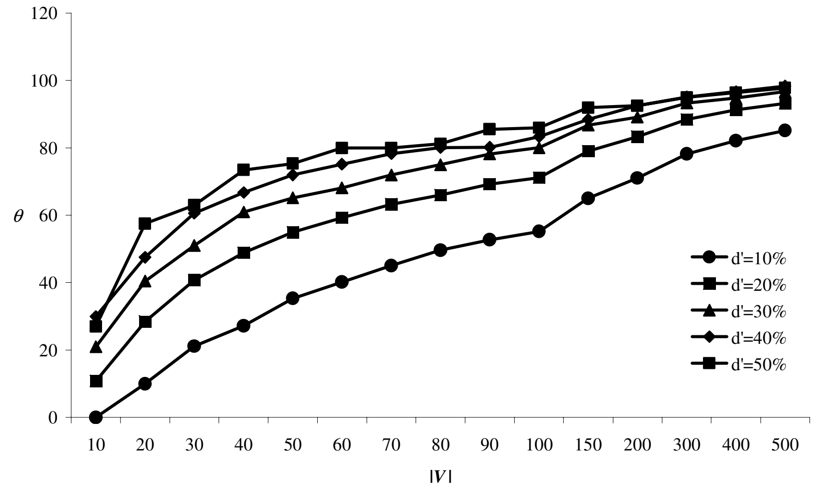


Fig. 2. Variation of θ for random graphs with d' varying from 10 to 50%

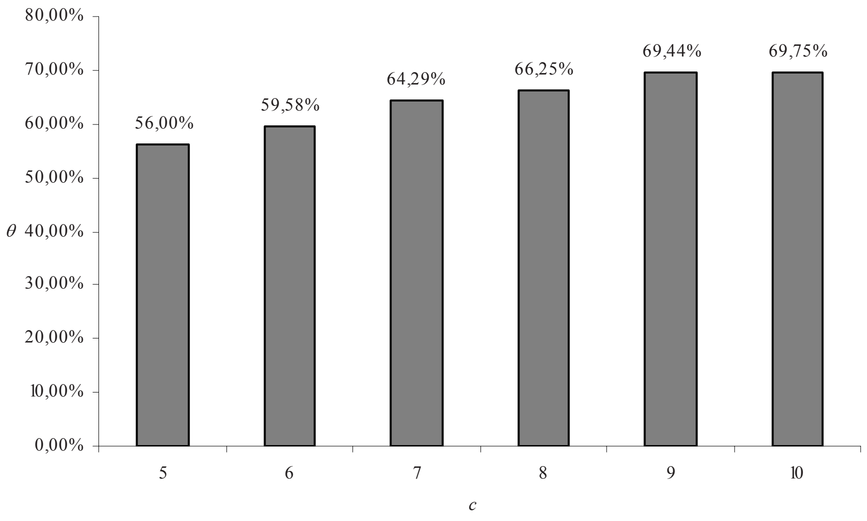


Fig. 3. Variation of θ for graphs from Computational Molecular Biology

According to figure Fig. 3, we notice that for graphs from Computational Molecular Biology, when the coverage c increases the rate θ increases too. We can explain this as follows : When the coverage c increases the number n of the vertices increases too. On the other hand, according to figure Fig. 2, when the number n of the vertices increases the rate θ increases too.

6 Conclusion

In this paper, we have presented a new approximation *Minimum Fragment Removal* (MFR) algorithm based on a new *Optimization Vertex Bipartization* (OVB) one. Our approximation MFR algorithm is of complexity $O(n^2)$ in computing time, where n is the number of the vertices of the *conflict graph*. We have run the program corresponding to our approximation MFR algorithm on random graphs and on graphs from Computational Molecular Biology. We have noticed that the obtained results are interesting. But, we can improve much more these results by improving the last step of our algorithm, to add more vertices to the constructed partitions.

References

1. Lancia, G., Bafna, V., Istrail, S., Lippert, R., Schwartz, R.: SNPs problems, complexity and algorithms. In: Meyer auf der Heide, F. (ed.) ESA 2001. LNCS, vol. 2161, pp. 182–193. Springer, Heidelberg (2001)
2. Rizzi, R., Bafna, V., Istrail, S., Lancia, G.: Practical Algorithms and Fixed-Parameter Tractability for the Single Individual SNP Haplotyping Problem. WABI: 29–43 (2002)
3. Bonizzonni, P., Vedova, G.D., Dondi, R., Li, J.: The haplotyping problem: An overview of computational models and solutions. J. Compu. Sci. Tech. 18(6), 675–688 (2003)

4. Halldorsson, B.V., Bafna, V., Edwards, N., Lippert, R., Yppseph, S., Istrail, S.: A survey of computational methods for determining haplotypes. In: The Proceeding of the 1st RECOMB Satellite Workshop on computational methods for SNPs and Haplotype Inference, pp. 26–47. Springer Verlag GMbH, Heidelberg (2004)
5. Li, L., Kim, M., Waterman, J.H., Haplotype, M.S.: reconstruction from SNP alignment. J. Comput. Biol. 11, 507–518 (2004)
6. Bafna, V., Istrail, S., Lancia, G., Rizzi, R.: Polynomial and APX-hard cases of the individual haplotyping problem. Theoret. Compu. Sci. 335, 109–125 (2005)
7. Panconesi, A., Sozio, M.: Fast hare: A fast heuristic for single individual SNP haplotype reconstruction. In: Jonassen, I., Kim, J. (eds.) WABI 2004. LNCS (LNBI), vol. 3240, pp. 266–277. Springer, Heidelberg (2004)
8. Cilibrasi, R., Iersel, L.V., Kelk, S., Tromp, J.: On the Complexity of the Single Individual SNP Haplotyping Problem. In: Casadio, R., Myers, G. (eds.) WABI 2005. LNCS (LNBI), vol. 3692, Springer, Heidelberg (2005)
9. Zhang, X.-S., Wang, R.-S., Wu, L.-Y., Chen, L.: Models and Algorithms for Haplotyping Problem. Current Bioinformatics 1(1), 105–114 (2006)
10. Lippert, R., Schwartza, R., Lancia, G., Istrail, S.: Algorithmic strategies for the SNPs haplotype assembly problem. Briefings in Bioinformatics 3(1), 23–31 (2002)
11. Kayem, A.V.D.M., Akl, S.G., Martin, P.: An independent set approach to solving the collaborative attack problem. In: Proceedings of the 17th IASTED International Conference, Parallel and Distributed computing and Systems, November 14-16 (2005)
12. Khuller, S., Raghavachari, B., Young, N.E.: Greedy Methods. Technical Report, Departement of Computer Science, University of Maryland (2005)
13. Garg, N., Vazirani, V., Yannakakis, M.: Approximate Max-flow Min-(Multi) Cut theorems and their applications. SIAM Journal on Computing 25(2), 235–251 (1996)
14. Raman, V., Saurabh, S., Sukdar, S.: Improved Exact Exponential Algorithms for Vertex Bipartization and Other Problems. In: The 9th Italian Conference on Theoretical Computer Science, ICTCS (2005)
15. Guo, J., Gramm, J., Hüffner, F., Niedermeier, R., Wernicke, S.: Compression–based fixed–parameter algorithms for Feedback Vertex Set and Edge Bipartization. Journal of Computer and System Sciences 72(8), 1386–1396 (2006)
16. Bellman, R.: Dynamic Programming. Princeton University Press, Princeton (1957)
17. Bellman, R.E., Dreyfus, S.E.: Applied Dynamic Programming. Princeton University Press, Princeton (1962)
18. Sakarovitch, M.: Optimisation combinatoire, graphes et programmation linéaire. Hermann, Paris (1984)
19. Hoare, C.A.R.: Quicksort. Computer Journal 5(1), 10–15 (1962)

Indexing Factors in DNA/RNA Sequences

Tomáš Flouri[1], Costas Iliopoulos[2], M. Sohel Rahman[2,⋆], Ladislav Vagner[1], and Michal Voráček[1,⋆⋆]

[1] Department of Computer Science & Engineering
Czech Technical University in Prague
Czech Republic

[2] Algorithm Design Group
Department of Computer Science
King's College London
Strand, London WC2R 2LS, England

{flourt1,xvagner,voracem}@fel.cvut.cz
{csi,sohel}@dcs.kcl.ac.uk

Abstract. In this paper, we present the Truncated Generalized Suffix Automaton (*TGSA*) and present an efficient on-line algorithm for its construction. *TGSA* is a novel type of finite automaton suitable for indexing DNA and RNA sequences, where the text is degenerate i.e. contains sets of characters. *TGSA* indexes the so called k-factors, the factors of the degenerate text with length not exceeding a given constant k. The presented algorithm works in $\mathcal{O}(n^2)$ time, where n is the length of the input DNA/RNA sequence. The resulting *TGSA* has at most linear number of states with respect to the length of the text. *TGSA* enables us to find the list $occ(u)$ of all occurrences of a given pattern u in degenerate text $\tilde{x}$ in time $|u| + |occ(u)|$.

1 Introduction

Degenerate strings are special strings having a set of symbols instead of one at any particular position, e.g. $\tilde{x} = [a][c, g][g][a, c, t][a, c]$. A normal (non-degenerate) string can be seen as a special kind of degenerate string with singleton set at each position. Degenerate strings are extensively used in molecular biology to express polymorphisms in DNA/RNA sequences, e.g. the polymorphism of protein coding regions caused by redundancy of the genetic code or polymorphism in binding site sequences of a family of genes. Indexing of short factors is a widely used and useful technique in stringology and bioinformatics. Use of k-factors (factors of length k) can be seen in solving diverse text algorithmic problems ranging from different string matching tasks [6] to motif finding [7] and alignment problems [8]. One can further mention the use of k-factors in FASTA and BLAST. In order to efficiently use the k-factors we need an efficient data structure to index them.

⋆ On leave from the Department of CSE, BUET, Dhaka-1000, Bangladesh.

⋆⋆ Supported by the Ministry of Education, Youth and Sports under research program MSM 6840770014 and the Czech Science Foundation as project No. 201/06/1039.

It may be noted here that, in the literature, there exists several recent works on indexing different kinds of k-factors [10,9].

An index over a fixed text $\tilde{x}$ can be defined as an abstract data type based on the set of all factors of $\tilde{x}$, denoted by $Fact(\tilde{x})$. Such data type is equipped with some operations that allow it to answer the following 2 queries. Firstly, given an arbitrary string u, an index can answer the question whether $u \in Fact(\tilde{x})$ (the existence query). Secondly, if $u \in Fact(\tilde{x})$, then it can find the list of all occurrences of u in $\tilde{x}$. In the case of exact string matching in *normal* string, there exist classical data structures for indexing, such as suffix trees, suffix arrays, DAWGs.

In [1], the *Generalized Factor Automaton* (*GFA*) was presented, which, to the best of our knowledge, is the first data structure serving as a full index of a given degenerate string. The precise number of states of *GFA* is not yet known, but, experiments showed that it tends to grow super-quadratically with respect to the length of the degenerate string [1]. As a result, it is not suitable for indexing very long texts. Later, in [3], the *Truncated Generalized Factor Automaton* (*TGFA*) was presented along with an algorithm for its construction based on the so-called "partial determinization". Essentially, *TGFA* is a modification of *GFA* that indexes only factors with length not exceeding a given constant k and it has at most linear number of states. The algorithm in [3] is based on the classical subset construction technique [5] and it inherits its space and time complexity. The space complexity is a bottleneck of this algorithm when indexing very long text since we need to determinize the corresponding large NFA. As a result, we loose the advantage of the much reduced size of TGFA due to its indirect construction from GFA using the partial determinization process and thereby limiting its applicability only for texts not exceeding KBs in length in practice.

In this paper, we present an on-line algorithm to construct *Truncated Generalized Suffix Automaton* (*TGSA*)[1]. This algorithm is space and time efficient, and, therefore, it enables us to construct *TGSA* for degenerate strings of length of tens of MBs. Notably, in [2], algorithms based on *GSA*[1] for searching regularities in degenerate strings were presented. These algorithms can be easily adapted for *TGSA* [4].

The rest of the paper is organized as follows. In Section 2, we present the notations and definitions. In Section 3, we present our main results. In particular, we present the algorithms for the construction of *GSA* and *TGSA*. Finally, we briefly conclude in Section 4.

2 Preliminaries

An *alphabet* Σ is a nonempty finite set of symbols. A *string* over a given alphabet is a finite sequence of symbols. The *empty string* is denoted by ε. The set of all strings over an alphabet Σ (including empty string ε) is denoted by Σ^*. A string $\tilde{x} = \tilde{x}_1\tilde{x}_2 \dots \tilde{x}_n$ is said to be *degenerate*, if it is built over the potential $2^{|\Sigma|} - 1$

[1] Note that *TGFA* and *TGSA* resp. (*GFA* and *GSA*) has the same transition diagram (i.e. number of states) and they differ only in sets of the final states.

non-empty sets of letters belonging to Σ. We say that $\tilde{x}_i \in \{\mathcal{P}(\Sigma) \setminus \emptyset\} = \mathcal{P}^+(\Sigma)$ and $|\tilde{x}_i|$ denotes the cardinality of $\tilde{x}_i$. In what follows, the set containing the letters a and c will be denoted by $[a, c]$ and the singleton $[c]$ will be simply denoted by c for the ease of reading. Also, we use the following convention: we use normal letters like x to denote normal strings. The same letter may be used to denote a degenerate string if used in the following way: $\tilde{x}$. The *Language represented by degenerate string* $\tilde{x}$ is the set $\mathcal{L}(\tilde{x}) = \{\ u \mid u = u_1u_2\dots u_n,\ u_j \in \tilde{x}_j,\ 1 \leq j \leq n,\ u \in \Sigma^*\ \}$. A string u is said to be an *element* of degenerate string $\tilde{x}$, denoted $u \in \tilde{x}$, if it is an element of language represented by $\tilde{x}$. A string u is a *factor* (resp. suffix) of a degenerate string $\tilde{x}$ if $\tilde{x} = \tilde{u}\tilde{v}\tilde{w}$ and $u \in \tilde{v}$ (resp. $u \in \tilde{w}$) and the set of all factors (resp. suffixes) of $\tilde{x}$ is denoted $\mathit{Fact}(\tilde{x})$ (resp. $\mathit{Suff}(\tilde{x})$). A string u is a *k-factor* of $\tilde{x}$ if $u \in \mathit{Fact}(\tilde{x})$ and $|v| = k$ and the set of all k-factors of $\tilde{x}$ is denoted $\mathit{Fact}_k(\tilde{x})$. A string v is a *at-most-k-factor* of $\tilde{x}$ if $v \in \mathit{Fact}(\tilde{x})$ and $|v| \leq k$ and the set of all at-most-k-factors of $\tilde{x}$ is denoted $\mathit{Fact}_k^-(\tilde{x})$. Similarly, we define k-suffix, at-most-k-suffix, $\mathit{Fact}_k(\tilde{x})$ and $\mathit{Fact}_k^-(\tilde{x})$. A (normal) string $u = u_1 \dots u_\ell$ of length ℓ is said to *occur* in a degenerate string $\tilde{x} = \tilde{x}_1\tilde{x}_2\dots\tilde{x}_n$ at position $i, 1 \leq i \leq n$ if and only if $u_j \in \tilde{x}_{i-\ell+j}$ for all $1 \leq j \leq \ell$. The list of all occurrences of u in $\tilde{x}$ is denoted by $\mathit{occ}_{\tilde{x}}(u)$.

Since our algorithms are based on a finite automata approach, we give brief definitions to related concepts below. A nondeterministic finite automaton M is a quintuple $(Q, \Sigma, \delta, q_0, F)$, where: Q is a finite set of states, Σ is an input alphabet, δ is a mapping $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \mapsto \mathcal{P}(Q)$ called a state transition function, $q_0 \in Q$ is the initial state, and $F \subseteq Q$ is a set of final states. A deterministic finite automaton M is a special case of nondeterministic finite automaton such that transition mapping is a function $\delta : Q \times \Sigma \mapsto Q$ and there is only one initial state $q_0 \in Q$. The number of states (resp. transitions) of M we denote by $|M|_Q$ (resp. $|M|_\delta$). The *left language* of state q of finite automaton M, denoted $\overleftarrow{\mathcal{L}}_M(q)$, is a set of strings for which there exists a sequence of transitions from the initial state to state q. The *left language* of finite automaton M, denoted $\overleftarrow{\mathcal{L}}_M$, is a union of left languages of all its states. *Language accepted* by finite automaton M, denoted $\mathcal{L}(M)$, is a set of words for which there exists sequence of transitions from the initial state to some of the final states. The *depth* of state q of acyclic finite automaton M is the length of the shortest path from the initial state to q. d-subset D of a state p of DFA M from NFA M' is the set of states of M' such that for each $q \in D$ it holds $\overleftarrow{\mathcal{L}}_{M'}(q) = \overleftarrow{\mathcal{L}}_M(p)$. A trie *TrieS* constructed for set of string S is deterministic finite automaton accepting S having the transition diagram in the form of a rooted $|\Sigma|$-ary tree. Note that a trie for S has maximum possible number of states among all deterministic finite automata accepting S.

3 Truncated Generalized Factor Automaton

In this section, we first introduce the concept of the *TGSA*. Next, we present an on-line algorithm for the construction of *GSA* followed by an on-line algorithm for the construction of *TGSA*. Notably, the TGSA construction algorithm is an extension of the GSA construction algorithm.

3.1 Generalized Suffix Automaton

Definition 1 (Generalized Suffix Automaton). *Given a degenerate string $\tilde{x}$,* Generalized Suffix Automaton *$GSA(\tilde{x})$ is a minimal deterministic finite automaton accepting the set $Suff(\tilde{x})$.*

GSA can be constructed easily by constructing a nondeterministic *GSA* and then determinizing it [1]. The nondeterministic *GSA* of $\tilde{x} = \tilde{x}_1\tilde{x}_2 \dots \tilde{x}_n$, has $n+1$ states, $q_0, q_1, \dots, q_n$. For each symbol $s \in \tilde{x}_i, 1 \leq i \leq n$, there exists one transition each from q_0 and q_{i-1} to the state q_i. The only final state is q_n.

If we use natural labelling of the states (a number of a state corresponds to a position in the text) then elements of a d-subset of a state corresponds to end-postions of factors from the left language of that state. Thus, *GSA* can be used as a full index of a degenerate string. If, during the determinization of nondeterministic *GSA*, we stop the expansion of states in depth equal to a given constant k (partial determinization, [3]), then we obtain automaton accepting at least the set of all at-most-k suffixes. The resulting automaton is *TGSA*. More formally, *TGSA* can be defined as follows:

Definition 2 (Truncated Generalized Suffix Automaton). *Suppose we are given a degenerate string $\tilde{x}$ and a positive integer k. Assume that $GSA(\tilde{x}) = M = (Q, \Sigma, \delta, q_0, F)$. Then, the* Truncated Generalized Suffix Automaton *for $\tilde{x}$ and k is a deterministic finite automaton $TGSA_k(\tilde{x}) = M_T = (Q_T, \Sigma_T, \delta, q_0, F_T)$, where $Q_T \subseteq Q$, $F_T \subseteq F$ and $\delta_T \subseteq \delta$, such that it holds:*

1. *$Fact_k^-(\tilde{x}) \subseteq \overleftarrow{\mathcal{L}}(M_T) \subseteq Fact(\tilde{x})$,*
2. *$Suff_k^-(\tilde{x}) \subseteq \mathcal{L}(M_T) \subseteq Suff(\tilde{x})$,*
3. *$|M_T|_Q \leq |Trie(Fact_k^-(\tilde{x}))|_Q$,*
4. *for M_T, the value $|\overleftarrow{\mathcal{L}}(M_T) \setminus Fact_k^-(\tilde{x})|$ is minimal among all automata satisfying Conditions 1–3.*

Remark 1. The number of states of $Trie(Fact_k^-(\tilde{x}))$ is given by the size of set $Fact_k^-(\tilde{x})$. The maximum number of all strings over an alphabet Σ of length at most k is limited by value of $\frac{|\Sigma|^{k+1}-1}{|\Sigma|-1}$. This implies that the number of states of $Trie(Fact_k^-(\tilde{x}))$ and hence the size of $TGSA_k(\tilde{x})$ never exceeds this value regardless the size of $\tilde{x}$. Next, the number of not-unique factors of length at most k grows in linear manner with respect to length of $\tilde{x}$ and hence the number of unique factors and hence the size of $TGSA_k(\tilde{x})$ cannot grow faster till it reaches the mentioned value $\frac{|\Sigma|^{k+1}-1}{|\Sigma|-1}$.

Since the construction of *TGSA* using partial determinization is costly, we, in this paper, develope a new space and time efficient online algorithm to construct *TGSA* directly. The algorithm is presented in the following sections.

3.2 On-Line Construction of Generalized Suffix Automaton

In this section we present the online construction algorithm for *GSA*. In the subsequent section, we show how to extend this algorithm to construct *TGSA*.

```
    procedure: CONSTRUCT
1   begin
2       $Q \leftarrow \emptyset$ ; $F \leftarrow$ NEW-LIST() ; $q_0 \leftarrow$ NEW-STATE($\{0\}$) ; $Q \leftarrow Q \bigcup \{q_0\}$ ;
        ADD($F, q_0$)
3       for $i \leftarrow 1$ to $n$ do
4           $p' \leftarrow$ NEW-STATE($\{i\}$) ; $F \leftarrow$ UPDATE()
5           $Q \leftarrow Q \bigcup \{p'\}$ ; $M_i \leftarrow (Q, \Sigma, \delta, q_0, F)$
6   end
```

Algorithm 1. On-line construction of a deterministic suffix automaton accepting a degenerate string $\tilde{x} = \tilde{x}_1\tilde{x}_2 \dots \tilde{x}_n$

The algorithm reads the degenerate string from left to right, one symbol-set at a time. After processing each symbol set $\tilde{x}_i$ $(1 \leq i \leq n)$, a suffix automaton $M_i = (Q, \Sigma, \delta, q_0, F)$ accepting the set *Suff*$(\tilde{x}[1, i])$, is constructed.

```
    procedure: UPDATE
1   begin
2       ADD($F', p'$)
3       while not EMPTY($F$) do
4           $p \leftarrow$ DEQUEUE($F$)                          traverse all final states
5           for each $x \in \tilde{x}_i$ do
6               if $\exists\ \delta(p, x)$ then EXTEND()
7               else $\delta(p, x) \leftarrow p'$
8       ADD($F', q_0$)
9   end
```

Algorithm 2. Procedure *update* traverses the states in the suffix path and updates their transitions

Each suffix $u \in$ *Suff*$(\tilde{x}[1, i + 1]) \setminus \{\varepsilon\}$ can be written in the form $u = vx$ where $v \in$ *Suff*$(\tilde{x}[1, i])$ and $x \in \tilde{x}_{i+1}$. This means that the construction of suffix automaton $M_{i+1} = (Q', \Sigma, \delta, q_0, F')$, accepting the set *Suff*$(\tilde{x})[1, i + 1]$, resides on traversing the set of final states F of suffix automaton M_i. Since the language $\overleftarrow{\mathcal{L}}_{M_i}(p)$, accepted by states $p \in F$, is the set *Suff*$(\tilde{x}[1, i])$, we can construct suffix automaton M_{i+1}, accepting the set *Suff*$(\tilde{x})[1, i + 1]$, by creating new transitions $\delta(p, x) = p'$, where p' is a newly created state $(p' \notin Q \wedge p' \in F')$ and $x \in \tilde{x}_{i+1}$. All newly created transitions lead to the same state p' which has a d-subset $\{i + 1\}$ (it indicates the position in the text where the given suffix ends).

In case a transition $\delta(p, x) = w$ already exists, it means that the automaton M_i already has a transition $\delta^i(q_0, u) = w$, for some $u \in$ *Suff*$(\tilde{x}[1, i + 1])$ and $w \in Q$. In other words, u was already indexed in M_i as a factor of degenerate string $\tilde{x}[1, i]$. In this case, it is necessary to check whether all $v \in \overleftarrow{\mathcal{L}}_{M_i}(w)$ belong to the set *Suff*$(\tilde{x}[1, i + 1])$. Considering that $v = ux$, where $x \in \tilde{x}_{i+1}$, if all u are

```
procedure: Extend
begin
    $w \leftarrow \delta(p, x)$
    $in \leftarrow$ Incoming($w$)                    number of incoming transitions of $w$
    $out \leftarrow$ Outgoing-SP($w$)              transitions from states in $F$ leading to $w$
    if $in = out$ then
        $w \leftarrow w \bigcup \{i\}$                          if equal extend the d-subset
        Add($F', w$)                                      add $w$ to final states of $M_{i+1}$
    else
        $s \leftarrow$ New-State($w \bigcup \{i\}$)                 if not equal, split the state
        for each $q \in F$ and $\delta(q, x) = w$ do $\delta(q, x) \leftarrow s$
        for each $a \in \Sigma$ do $\delta(s, a) \leftarrow \delta(w, a)$
        Add($F', s$)                                      add $s$ to final states of $M_{i+1}$
end
```

Algorithm 3. Procedure *extend* adds states to the new suffix path by extending the d-subsets of existing states or creating new ones

part of the set $Suff(\tilde{x}[1, i])$, then all v are part of the set $Suff(\tilde{x}[1, i+1])$ and thus, it is only needed to extend the d-subset of w with the element $\{i+1\}$, which denotes the new found position in the text.

In order to check whether all u are part of $Suff(\tilde{x}[1, i])$, it is necessary to traverse all final states of M_i, which actually represent the set $Suff(\tilde{x}[1, i])$ and count the x-transitions leading to state w. If the counted number is equal to the number of incoming transitions of state w, then all v are part of $Suff(\tilde{x}[1, i+1])$. If there exists at least one $u \notin Suff(\tilde{x}[1, i])$ (in other words, there exists one incoming transition of state w, which does not originate from a final state of M_i), then the language $\overleftarrow{\mathcal{L}}_{M_i}(w)$ is formed by factors (not only suffixes) of $\tilde{x}[1, i+1]$. Thus, it is required to "split" the state in two states, one representing the factors which are not suffixes and one representing the suffixes.

This is carried out by "cloning" state w (we will denote the new state as $clone(w)$) and changing all transitions $\delta(p, x) = w$, where $p \in F$, to $\delta(p, x) = clone(w)$. By the term "cloning" of state w, we mean the creation of a new state *clone* (w) having the same d-subset as w and the creation of transitions $\delta(clone(w), a) = \delta(w, a)$ for all $a \in \Sigma$. The complete procedure is formally presented in Algorithms 1-3.

In Algorithm 4, we present a simpler version of Algorithm 3. In this case, instead of counting x-transitions and then deciding whether to split state w, the split is performed always. While traversing the rest of the final states, if a transition is found to lead to w, it is redirected to *clone* (w). In case w is left with no incoming transitions, it is removed. Notably, usage of Algorithm 3 or 4 does not change the asymptotic time complexity of the algorithm.

Lemma 1. *Given a degenerate string $\tilde{x} = \tilde{x}_1\tilde{x}_2 \ldots \tilde{x}_n$, Algorithm 1 correctly constructs $GSA(\tilde{x})$ in time $\mathcal{O}(n^2\sigma\varphi)$, where φ is the number of states of the resulting automaton and σ is the size of used alphabet.*

```
procedure: EXTEND-2
1  begin
2      $w \leftarrow \delta(p,x)$
3      if $\nexists$ $clone(w)$ or $clone(w) \bigcap \{i\} = \emptyset$ then
4          $clone(w) \leftarrow$ NEW-STATE($w \bigcup \{i\}$)
5          for each $a \in \Sigma$ do $\delta(clone(w),a) \leftarrow \delta(w,a)$
6          ADD($F', clone(w)$)
7      $\delta(p,x) \leftarrow clone(w)$
8      if INCOMING($w$) = 0 then REMOVE($w$)
9  end
```

Algorithm 4. Optimized EXTEND procedure

3.3 On-Line Construction of Truncated Generalized Suffix Automaton

As before, the algorithm for constructing a *TGSA* reads the degenerate string $\tilde{x} = \tilde{x}_1\tilde{x}_2 \ldots \tilde{x}_n$ from left to right, one symbol set at a time. After processing each symbol set $\tilde{x}_i$ $(1 \leq i \leq n)$, a *TGSA* $M_i = (Q, \Sigma, \delta, q_0, F)$ accepting at least all k-suffixes of $\tilde{x}[1,i]$ is created. The *TGSA* M_{i+1} is constructed in a similar fashion as presented in Section 3.2 (Algorithms 1-4). The main difference is that instead of only traversing the final states of M_i and creating or updating transitions where

```
procedure: CONSTRUCT
input: $\tilde{x} = \tilde{x}_1\tilde{x}_2 \ldots \tilde{x}_n$ - a degenerate string over alphabet $\Sigma$, $k$ - maximum
       length of factors to index
output: $M \leftarrow (Q, \Sigma, \delta, q_0, F)$ - a TGSA over $\tilde{x} = \tilde{x}_1\tilde{x}_2 \ldots \tilde{x}_n$
1  begin
2      $Q \leftarrow \emptyset$ ; $q_0 \leftarrow$ NEW-STATE($\{0\}$) ; $Q \leftarrow Q \bigcup \{q_0\}$ ; $D \leftarrow$ NEW-QUEUE()
3      for $i \leftarrow 1$ to $n$ do
4          $F \leftarrow \{q_0\}$ ; $p' \leftarrow$ NEW-STATE($\{i\}$) ; $Q \leftarrow Q \bigcup \{p'\}$
5          ENQUEUE($D, q_0$) ; $status(q_0) \leftarrow OPEN$ ; $used \leftarrow false$
6          while not EMPTY($D$) do
7              $p \leftarrow$ DEQUEUE($D$)
8              if $level(p) < k$ then
9                  if $i - 1 \in p$ or $p = q_0$ then UPDATE()
10                 ENQUEUE-SUCCESSORS($D, p, p'$)
11             else ELIMINATE-REDUNDANT($p$)
12         if $used = false$ then REMOVE($p'$)            Remove $p'$ if not used
13         $M_i \leftarrow (Q, \Sigma, \delta, q_0, F)$
14         for each $q \in Q$ do $status(q) \leftarrow FRESH$
15 end
```

Algorithm 5. On-line construction of a *TGSA* accepting all k-suffixes of a degenerate string $\tilde{x} = \tilde{x}_1\tilde{x}_2 \ldots \tilde{x}_n$

```
procedure: UPDATE()
begin
    for each $x \in \tilde{x}_i$ do
        if $\exists\delta(p, x)$ then EXTEND($\delta(p, x), p, x$)
        else
            $\delta(p, x) \leftarrow p'$ ; $level(p') \leftarrow$ MIN($level(p'), level(p) + 1$)
            if $used{=}false$ then $F \leftarrow F \bigcup \{p'\}$ ; $used \leftarrow true$        $p'$ was used
end
```

Algorithm 6. As in Algorithm 2, procedure *update* updates the appropriate transitions of final states of automaton M_{i-1}

```
procedure: EXTEND($w, p, x$)
input: $w$ - destination state, $p$ - source state, $x$ - transition symbol
begin
    if $\nexists$ $clone(w)$ or $clone(w) \bigcap \{i\} = \emptyset$ then
        $clone(w) \leftarrow$ NEW-STATE($w \bigcup \{i\}$) ; $status(clone(w)) \leftarrow FRESH$
        for each $a \in \Sigma$ do $\delta(clone(w), a) \leftarrow \delta(w, a)$
        $F \leftarrow F \bigcup \{clone(w)\}$
    $\delta(p, x) \leftarrow clone(w)$ ; $level(clone(w)) \leftarrow$ MIN($level(clone(w)), level(p) + 1$)
    if INCOMING($w$) = 0 then REMOVE($w$)
end
```

Algorithm 7. Optimized EXTEND procedure

necessary, it is required to traverse all states of M_i in a *breadth-first-search* (BFS) manner.

This is because when transforming M_i to M_{i+1}, certain states of M_i can change depth in the new M_{i+1} to a value higher than k, which (states) must be eliminated. The BFS is carried out in order to mark all visited states that have a depth less than k in the resulting M_{i+1}. For each final state of M_i, having a depth less than k, the same steps described in Section 3.2 (Algorithms 1-4) are applied. All states having a depth greater than k are removed, resulting in a new M_{i+1} having at most as many states as $TrieSuff_k(\tilde{x})$.

In step i, to check whether a given state is a final state of M_{i-1}, we need only check whether element $i-1$ is part of its d-subset. This can be done in constant time if the d-subset is implemented as a linked-list of arrays and storing the size of each state's d-subset. Element $i-1$ will then be either on the last position or one position before the last one since the d-subset is always sorted (this is a consequence of reading the degenerate string $\tilde{x} = \tilde{x}_1\tilde{x}_2 \ldots \tilde{x}_n$ from left to right). The whole process is presented in Algorithms 5-10.

Lemma 2. *Given a degenerate string $\tilde{x} = \tilde{x}_1\tilde{x}_2 \ldots \tilde{x}_n$ and positive integer k, Algorithm 5 correctly constructs $TGSA_{\tilde{x}}(k)$ in $\mathcal{O}(n^2\sigma^{k+2})$ time, where σ is the size of used alphabet.*

```
   procedure: ELIMINATE-REDUNDANT($p$)
   input: $p$ - the state whose non-visited successors are to be eliminated
1  begin
2      for each $u \leftarrow \delta(p, a),\ a \in \Sigma$ do
3          if $status(u) = FRESH$ then
4              $\delta(p, a) \leftarrow nil$ ; if INCOMING($u$) $= 0$ then REMOVE($u$)
5  end
```

Algorithm 8. Elimination of redundant states (states with a *level* $> k$)

```
   procedure: ENQUEUE-SUCCESSORS($D, p, p'$)
   input: $D$ - BFS Queue, $p$ - state whose successors are to be enqueued, $p'$ -
          new state
1  begin
2      for each $\delta(p, a) \neq p',\ a \in \Sigma$ do
3          if $status(\delta(p, a)) = FRESH$ then
4              ENQUEUE($D, \delta(p, a)$) ; $level(\delta(p, a)) \leftarrow level(p) + 1$
5              $status(\delta(p, a)) \leftarrow OPEN$
6  end
```

Algorithm 9. Part of the *breadth-first-search* algorithm which enqueues all successors of a state

```
   procedure: REMOVE($p$)
   input: $p$ - the state to remove
1  begin
2      for each $u \leftarrow \delta(p, a),\ a \in \Sigma$ do
3          if $status(u) = FRESH$ then
4              $\delta(p, a) \leftarrow nil$ ; if INCOMING($u$) $= 0$ then REMOVE($u$)
5      $Q \leftarrow Q \setminus \{p\}$ ; DELETE-STATE($p$)
6  end
```

Algorithm 10. Removal of state p and all its, unreachable from other states, successors

Corollary 1. *Given a degenerate DNA/RNA sequence $\tilde{x} = \tilde{x}_1\tilde{x}_2 \ldots \tilde{x}_n$ and a positive integer k ($n \gg k$), Algorithm 5 correctly constructs $TGSA_{\tilde{x}}(k)$ in $\mathcal{O}(n^2)$ time.*

Proof. This follows immediately because σ is constant for DNA/RNA sequences.

Remark 2. We remark that in practical setting we are interested in indexing strings of size in 10's of MBs and the patterns involved are typically small and hence the assumption that $n >> k$ is a valid assumption.

Existence Query. As is mentioned before, *TGSA* has at most linear number of states. However, with each state there is an associated list namely d-subset. Since each such list can have upto n items, the space requirements becomes higher. But if we need only answer the existence queries, we don't need the d-subset lists and in that case we can get a very efficient version of *TGSA*.

4 Conclusion

In this paper, we have presented efficient on-line algorithm to construct *Truncated Generalized Suffix Automaton* (*TGSA*), a novel type of finite automaton serving as a k-factor-index for a degenerate strings. The *TGSA* algorithm works in $\mathcal{O}(n^2)$ time for DNA/RNA sequences and the resulting automaton has at most linear number of states. The already known algorithms for searching regularities in degenerate strings designed originally for *GSA* can be easily modified to work with *TGSA* and hence, *TGSA* can be used for searching regularities in degenerate strings as well.

References

1. Voráček, M., Melichar, B., Christodoulakis, M.: Generalized and weighted strings: Repetitions and pattern matching. In: String Algorithmics, pp. 225–248. KCL Publications, King's College London (2004)
2. Voráček, M., Melichar, B.: Searching for regularities in generalized strings using finite automata. In: Proceedings of International Conference on Numerical Analysis and Applied Mathematics. Wiley-VCH, Weinheim (2005)
3. Voráček, M., Vagner, V., Flouri, T.: Indexing Degenerate Strings. In: Proceedings of International Conference on Computational Methods in Science and Engineering, American Institute of Physics, Melville, New York (2007)
4. Flouri, T.: Indexing Degenerate Strings, Master Thesis, Czech Technical University, Prague (2008)
5. Hopcroft, J.E., Ullman, J.D.: Introduction to automata, languages and computation. Addison-Wesley, Reading (1979)
6. Navarro, G., Sutinen, E., Tanninen, J., Tarhio, J.: Indexing Text with Approximate q-Grams. In: Giancarlo, R., Sankoff, D. (eds.) CPM 2000. LNCS, vol. 1848, pp. 350–363. Springer, Heidelberg (2000)
7. Iliopoulos, C.S., McHugh, J., Peterlongo, P., Pisanti, N., Rytter, W., Sagot, M.: A First Approach to Finding Common Motifs with Gaps. International Journal of Foundations of Computer Science (2004)
8. Ma, B., Tromp, J., Li, M.: Patternhunter: faster and more sensitive homology search. Bioinformatics 18(3), 440–445 (2002)
9. Iliopoulos, C.S., Sohel Rahman, M.: Indexing factors with gaps. Algorithmica (to appear) (DOI: 10.1007/s00453-007-9141-3)
10. Peterlongo, P., Allali, J., Sagot, M.-F.: The gapped-factor tree. In: Holub, J., Zdrek, J. (eds.) Stringology, pp. 182–196. Czech Technical University, Prague (2006)

Implementation of a Swap Matching Algorithm Using a Graph Theoretic Model

Pavlos Antoniou, Costas S. Iliopoulos,
Inuka Jayasekera[⋆], and M. Sohel Rahman[⋆⋆,⋆⋆⋆]

Algorithm Design Group
Department of Computer Science, King's College London,
Strand, London WC2R 2LS, England
{pavlosant,jayasek,csi,sohel}@dcs.kcl.ac.uk
http://www.dcs.kcl.ac.uk/adg

Abstract. The swap matching problem consists if finding a pattern in a text, while allowing for transpositions in the pattern. A new approach using a graph-theoretic model was presented in [6] by Iliopoulos et al. In this paper we present a useful application for this algorithm and provide an analysis of its running time with a naive approach through implementation.

1 Introduction

The problem of the pattern matching with swaps (the Swap Matching problem, for short) is an interesting variant of the classical pattern matching problem motivated by practical applications. In this problem, the pattern P is said to match the text T at a given location i, if adjacent pattern characters can be swapped, if necessary, so as to make the pattern identical to the substring of the text ending (or equivalently, starting) at location i. All the swaps are constrained to be disjoint, i.e., each character is involved in at most one swap. The first non-trivial results for this problem was presented in [1]. In [1], the problem was solved in time $O(nm^{1/3} \log m \log \sigma)$, where $\sigma = \min(|\Sigma|, m)$. Later, in [2], an improved algorithm with $O(n \log m \log \sigma)$ time was presented. Also, certain special but rather restrictive cases were studied in [3], for which, $O(n \log^2 m)$ time solution were presented. Notably, all the above solutions to swap matching problem depend on the fast fourier transform (FFT) technique.

Very recently, Iliopoulos and Rahman [6] presented a new graph theoretic model to solve the swap matching problem. Incorporating their model to the classic shift-or algorithm [4], they presented a simple and efficient algorithm (referred to as the IR algorithm henceforth) for the problem that runs very fast

⋆ Supported by the Engineering and Physical Sciences Research Council (EPSRC) under the Doctoral Training Award (DTA).

⋆⋆ Supported by the Commonwealth Scholarship Commission in the UK under the Commonwealth Scholarship and Fellowship Plan (CSFP).

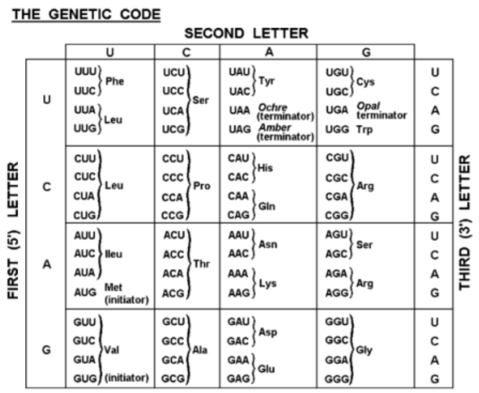
⋆⋆⋆ On Leave from Department of CSE, BUET, Dhaka-1000, Bangladesh.

for sufficiently short patterns. In particular, if the pattern size is similar to the word size of the target machine, then the IR algorithm runs in $O(n \log n)$ time. Moreover, the algorithm is quite simple and lightweight and doesn't use any heavy machinery like the FFT techniques. In this paper, we are interested in the practical performance of the IR algorithm. In particular, we use the algorithm for an interesting problem from biological computing and provide an experimental analysis of its running time and present detailed comparison with the naive approach to solve the problem.

1.1 Biological Motivation

As has already been mentioned, the swap matching problem is motivated by practical applications. We are particularly interested in its application in biological computing. More specifically, an interesting application of the IR algorithm lies in the process of translation in Molecular Biology. The ability of our algorithm to match the pattern not only in its identical form in the text, but also by its swaps has a direct application with the genetic triplets, otherwise called, codons. In the process of translation we have the protein synthesis where the genetic code is translated to proteins. The mRNA, holding the genetic information from the DNA, travels to the cytoplasm in the cell, and there it engages with the ribosomes to start the process of translation. The assembly of a new polypeptide is controlled by a triplet genetic code. Successive groups of three nucleotides (codons) in the mRNA sequence are encoded sequentially in order to specify specific amino acids [7]. Figure 1 presents the encoded amino acids resulted from each possible codon triplet.

Only the central part of the sequence of the mRNA is encoded into proteins. Around both sides of the central part lie flanking sequences. The process begins with DNA polymerase scanning the mRNA, starting with the flanking sequence until it finds the initiation codon. This codon is usually *AUG*, *ACG*, *GUG*, or


THE GENETIC CODE

FIRST (5') LETTER	SECOND LETTER U	SECOND LETTER C	SECOND LETTER A	SECOND LETTER G	THIRD (3') LETTER
U	UUU Phe	UCU Ser	UAU Tyr	UGU Cys	U
	UUC	UCC	UAC	UGC	C
	UUA Leu	UCA	UAA *Ochre* (terminator)	UGA *Opal* terminator	A
	UUG	UCG	UAG *Amber* (terminator)	UGG Trp	G
C	CUU Leu	CCU Pro	CAU His	CGU Arg	U
	CUC	CCC	CAC	CGC	C
	CUA	CCA	CAA Gln	CGA	A
	CUG	CCG	CAG	CGG	G
A	AUU Ileu	ACU Thr	AAU Asn	AGU Ser	U
	AUC	ACC	AAC	AGC	C
	AUA	ACA	AAA Lys	AGA Arg	A
	AUG Met (initiator)	ACG	AAG	AGG	G
G	GUU Val	GCU Ala	GAU Asp	GGU Gly	U
	GUC	GCC	GAC	GGC	C
	GUA	GCA	GAA Glu	GGA	A
	GUG (initiator)	GCG	GAG	GGG	G

Fig. 1. The encoded amino acids of the translation process. Picture taken from http://www.msstate.edu/dept/poultry/pics/gnscht.gif

CUG. Besides starting codons, we also have STOP codons UAA, UAG, UGA; when these codons are encountered the process ends.

As is already mentioned, the genetic code is a three-letter code. Since there are four bases to choose from, logically thinking, one would assume that there are $4^3 = 64$ different amino acids encoded; but in reality, there are only 20. That is because the genetic code is degenerate. Each amino acid is specified on average by about three different codons [7]. So, some different codons specify the same amino acids and some specify different ones. In some cases, swapping the letters of a specific triplet can result in encoding the same amino acid or a different one. These swaps and base substitutions are usually the causes of mutations in genes. There are two different types of mutations, namely, the Silent mutations and the Nonsynonymous mutations.

Now, the IR algorithm [6], can be used to detect in the text the possible positions of the start and stop codons of an mRNA sequence and provide us with the hints as to where the flanking regions are in respect to the translated mRNA region. If our input pattern, does not contain a $STOP$ codon by itself but one of its permutations does, the algorithm by swapping the letters that consist it, will notify us immediately where a possible STOP codon lies in the text so we can avoid it, in order to maintain a larger translated region. Similarly, if we are looking for a starting codon the algorithm does not need the exact triplet as an input pattern to find a possible $START$ codon early on in the text but its permutation will suffice.

The rest of the paper is organized as follows. In Section 2, we give a brief overview of the model and algorithm presented in [6]. In Section 3, we give the details of our implementation. In Section 4, we present our experimental results in the form of graphs along with a detailed analysis of the results. Finally, we briefly conclude in Section 5.

2 Preliminaries

In this section, we first present the basic concepts and then briefly review the model and algorithm presented in [6].

2.1 Basic Definitions

A *string* is a sequence of zero or more symbols from an alphabet Σ. A string X of length n is denoted by $X[1..n] = X_1X_2 \dots X_n$, where $X_i \in \Sigma$ for $1 \leq i \leq n$. The *length* of X is denoted by $|X| = n$. A string w is called a *factor* of X if $X = uwv$ for $u, v \in \Sigma^*$; in this case, the string w occurs at position $|u| + 1$ in X. The factor w is denoted by $X[|u| + 1..|u| + |w|]$. A *k-factor* is a factor of length k. A *prefix (or suffix)* of X is a factor $X[x..y]$ such that $x = 1$ $(y = n)$, $1 \leq y \leq n$ $(1 \leq x \leq n)$. We define i-th prefix to be the prefix ending at position i i.e. $X[1..i], 1 \leq i \leq n$. On the other hand, i-th suffix is the suffix starting at position i i.e. $X[i..n], 1 \leq i \leq n$.

0	1	2	3	4	5	6	7	8	9	10	11	12	13
T	C	A	T	C	A	C	T	C	A	T	C	A	T
	C	**A**	T		A	C	T				**C**	**A**	T
		A	**T**	**C**				**C**	**A**	T			
									A	**T**	**C**		

Fig. 2. Pattern matching with swaps example

Definition 1. *A* swap permutation *for X is a permutation $\pi : \{1, \dots, n\} \to \{1, \dots, n\}$ such that:*

1. *if $\pi(i) = j$ then $\pi(j) = i$ (characters are swapped).*
2. *for all $i, \pi(i) \in \{i-1, i, i+1\}$ (only adjacent characters are swapped).*
3. *if $\pi(i) \neq i$ then $X_{\pi(i)} \neq X_i$ (identical characters are not swapped).*

For a given string X and a swap permutation π for X, we use $\pi(X)$ to denote the *swapped version* of X, where $\pi(X) = X_{\pi(1)}X_{\pi(2)} \dots X_{\pi(n)}$.

Definition 2. *Given a text $T = T_1T_2 \dots T_n$ and a pattern $P = P_1P_2 \dots P_m$, P is said to* swap match *at location i of T if there exists a swapped version P' of P that matches T at location i, i.e. $P'_j = T_{i-m+j}$ for $j \in [1..m]$.*

Problem "SM" (Pattern Matching with Swaps). *Given a text $T = T_1T_2 \dots T_n$ and a pattern $P = P_1P_2 \dots P_m$, we want to find each location $i \in [1..n]$ such that P swap matches with T at location i.*

Example 1. The swapped matches of pattern, $P = ACT$, in text, $T = TCATC$ $ACTC$ $ATCAT$, can be seen in Figure 2 at positions 1, 2, 5, 8, 9 and 11 in T. The positions in which the characters of P had to be swapped to match those in T, are shown in bold.

2.2 Review of the Model and Algorithm of [6]

In this section, we briefly review the model and algorithm presented in [6].

Definition 3. *Given a text $T = T_1 \dots T_n$ of Problem SM, a $\mathcal{T}$-**graph**, denoted by $T^G = (V^T, E^T)$, is a directed graph with n vertices and $n-1$ edges such that $V^T = \{1, 2, \dots n\}$ and $E^T = \{(i, i+1) | 1 \leq i < n\}$. For each $i \in V_T$ we define label$(i) = T_i$ and for each edge $e \equiv (i,j) \in E_T$ we define label$(e) \equiv$ label$((i,j)) \equiv ($label$(i),$ label$(j)) = (T_i, T_j)$.*

Definition 4. *Given a text $P = P_1 \dots P_m$ of Problem SM, a $\mathcal{P}$-**graph**, denoted by $P^G = (V^P, E^P)$, is a directed graph with $3m - 2$ vertices and at most $5m - 9$ edges. The vertex set V^P can be partitioned into three disjoint vertex sets namely $V^P_{(+1)}, V^P_0, V^P_{(-1)}$ such that $|V^P_{(+1)}| = |V^P_{(-1)}| = m - 1$ and $|V^P_{(0)}| = m$. The partition is defined in a $3 \times m$ matrix $M[3, m]$ as follows. For the sake of notational symmetry we use $M[-1], M[0]$ and $M[+1]$ to denote respectively the rows $M[1], M[2]$ and $M[3]$ of the matrix M.*

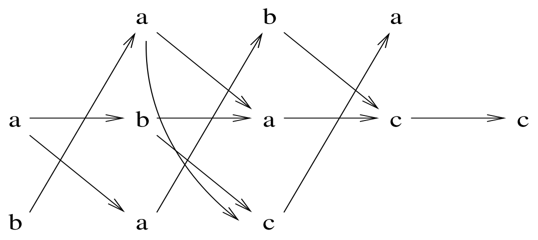


Fig. 3. P-graph for a pattern, P = abacc

1. $V^P_{(-1)} = \{M[-1,2], M[-1,3], \dots M[-1,m]\}$
2. $V^P_{(0)} = \{M[0,1], M[0,2], \dots M[0,m]\}$
3. $V^P_{(+1)} = \{M[+1,1], M[+1,2], \dots M[+1,m-1]\}$

The labels of the vertices are derived from P as follows:

1. *For each vertex* $M[-1,i] \in V^P_{(-1)}, 1 < i \leq m$:

$$label(M[-1,i]) = \begin{cases} P_{i-1} & \text{if } P_{i-1} \neq P_i, \\ \mathcal{X} & \text{if } P_{i-1} = P_i, \text{ where } \mathcal{X} \notin \Sigma \end{cases} \tag{1}$$

2. *For each vertex* $M[0,i] \in V^P_{(0)}, 1 \leq i \leq m, label(M[0,i]) = P_i$
3. *For each vertex* $M[+1,i] \in V^P_{(+1)}, 1 \leq i < m$:

$$label(M[+1,i]) = \begin{cases} P_{i+1} & \text{if } P_i \neq P_{i+1}, \\ \mathcal{X} & \text{if } P_i = P_{i+1}, \text{ where } \mathcal{X} \notin \Sigma \end{cases} \tag{2}$$

The edge set E^P *is defined as the union of the sets* $E^P_{(-1)}, E^P_{(0)}$ *and* $E^P_{(+1)}$ *as follows:*

1. $E^P_{(-1)} = \{(M[-1,i], M[0,i+1]), (M[-1,i], M[+1,i+1]) \mid 2 \leq i \leq m-2 \bigwedge label(M[-1,i]) \neq \mathcal{X}\} \bigcup \{(M[-1,m-1], M[0,m]) \mid label(M[-1,m-1]) \neq \mathcal{X}\}$
2. $E^P_{(0)} = \{(M[0,i], M[0,i+1]) \mid 1 \leq i \leq m-1\} \bigcup \{((M[0,i], M[+1,i+1]) \mid 1 \leq i \leq m-2 \bigwedge label(M[+1,i+1]) \neq \mathcal{X}\}$
3. $E^P_{(+1)} = \{(M[+1,i], M[-1,i+1]) \mid 1 \leq i \leq m-1 \bigwedge label(M[+1,i]) \neq \mathcal{X}\}$[1]

The labels of the edges are derived from using the labels of the vertices in the obvious way.

Example 2. Figure 3 shows the P-Graph for a pattern, P = abacc.

[1] Note that, if $label(M[+1,i]) = \mathcal{X}$ then $label(M[-1,i+1]) = \mathcal{X}$ as well.

$$\begin{tikzcd}
 & \mathbf{a} \arrow[r] & \mathbf{b} \arrow[r] & \mathbf{a} & \\
\mathbf{a} \arrow[ur] & \mathbf{b} \arrow[ur] & \mathbf{a} \arrow[ur] & \mathbf{c} & \mathbf{c} \\
\mathbf{b} \arrow[ur] \arrow[r] & \mathbf{a} \arrow[ur] \arrow[r] & \mathbf{c} \arrow[ur] & &
\end{tikzcd}$$

Fig. 4. F-graph for a pattern, P = abacc

Definition 5. *Given a $\mathcal{P}$-graph $P^G = (V^P, E^P)$, the* forbidden Graph (F-graph) $\overline{P}^G = (\overline{V}^P, \overline{E}^P)$ *is such that* $\overline{V}^P = V^P$ *and* $\overline{E}^P$ *is defined as follows:*
$\overline{E}^P = \{(M[i,j], M[i,j+1]) \mid i \in \{-1,0,+1\}, 1 \leq j < m,$
$(label(M[i,j]) \neq \mathcal{X} \ \bigvee \ label(M[i,j+1]) \neq \mathcal{X}) \ \bigwedge \ (\forall (M[k,j], M[k,j+1]) \in E^P, k \in \{-1,0,+1\},$
$label((M[k,j], M[k,j+1])) \neq label((M[i,j], M[i,j+1])))\}.$

Example 3. Figure 4 shows the corresponding F-Graph for the P-Graph shown in Figure 3 for a pattern, P = abacc.

Definition 6. *For each $c \in \Sigma$ let D_c be a bit array of size m such that for $1 \leq i \leq m$, $D_c[i] = 0$ of and only if $P_i = c$. The array D_c which denotes the position of the character c in p will be referred to as the $D - mask$.*

Once we have computed the D-masks and F-graph, we can now begin to search for swapped occurrences of our pattern in the text. To do this, the IR algorithm uses a modified Shift-Or [4] algorithm to compute the swapped occurrences of the pattern[2]. In particular, the algorithm computes R_{j+1} using the following formula [6]:

$$R_{j+1} = \ SHIFT(R_j) \ OR \ D_{T_{j+1}} \ OR \ F_{(T_j, T_{j+1})}$$

3 Implementation

In this section we describe in detail how the steps of the algorithm were implemented. In the following pseudocodes, p will represent the pattern we will be searching for; D and F will be the D-mask and forbidden graph respectively, in a hash table representation.

1. Constructing the D-mask vector
   Algorithm 1 presents the pseudocodes for the construction of the D-masks.
2. Constructing the F-graph
   Although [6] defines the F-graph using the inverse of the P-graph, it is not necessary to build the P-graph at all. The direct construction of the F-graph can be done easily using Algorithm 2.

[2] Here we assume that the readers are familiar with the Shift-Or algorithm. For further details, we refer to [4] or [5].

Algorithm 1. Algorithm to construct D-Mask

```
buildDMasks ($p$)
for $i = 0$ to $m$ - 1 do
    $x = p[i]$
    $D[x] = (01 \ll m - i - 1) \mid D[x]$
end for
for each $D[x] = y$ do
    $D[x] = y$ XOR $R$
end for
return $D$
```

Algorithm 2. Algorithm to construct F-Graph

```
buildFGraph ($p$)
for $i = 0$ to $m$ - 1 do
    if $p[i]$ != $p[i - 1]$ then
        if $p[i - 1]$ != $p[i + 1]$ then
            $F[p[i - 1]] = F[p[i - 1]]$ OR $(01 \ll (m - i - 1))$
        else if $p[i]$ != $p[1 - 2]$ then
            $F[p[i]] = F[p[i]]$ OR $(01 \ll (m- i - 1))$
        else if $i$ != 1 AND $p[i - 2]$ != $p[i - 1]$ then
            if $p[i]$ != $p[i - 2]$ then
                $F[p[i - 2]] = F[p[i - 2]]$ OR $(01 \ll (m - i - 1))$
            end if
        else
            if $i$ != $(m - 1)$ AND $p[i]$ != $p[i + 1]$ then
                if $p[i - 1]$ != $p[i + 1]$ AND $p[i]$ != $p[i - 2]$ then
                    $F[p[i + 1]] = F[p[i + 1]]$ OR $(01 \ll (m - i - 1))$
                end if
            end if
        end if
    end if
end for
return $F$
```

Algorithm 3. Modified Shift-Or algorithm

```
$R = (R \gg 1)$ OR $D[t[0]]$
for $i = 0$ to $n$ - 1 do
    if $D[t[i]]$ != NULL then
        $R = (R \gg 1) \mid D[t[i]] \mid F[t[i - 1]t[i]]$
    else
        $R = (R \gg 1) \mid D[X] \mid F[t[i - 1]t[i]]$
    end if
    if !($R$ AND 1) then
        print "Match found ending at" $i$
    end if
end for
```

3. Perform modified Shift-Or
   Algorithm 3 shows how the algorithm then uses the D-mask and F-graph to find swapped matches of the pattern in the text.

4 Results

To show the full efficiency of the IR algorithm, we have performed two types of tests. The first test is related directly to the application in codon matching. The

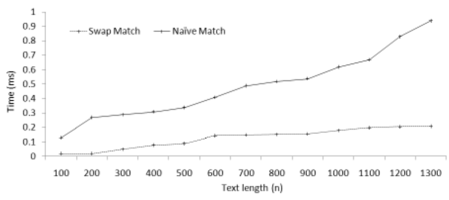


Fig. 5. Single codon matching

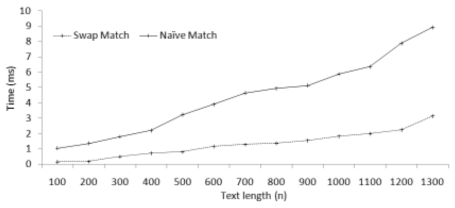


Fig. 6. All codon matching

second consisted of carrying out tests on various generated texts to show how it fairs against the naive approach in general. The tests were run in a Windows environment with a AMD Turion 64 processor of 1.6 GHz and 1GB RAM. The implementation was done in Java (version 1.50).

Notably, we have chosen to compare the IR algorithm against a naive implementation instead of the algorithms employing the FFT techniques (specifically the algorithm of [2]). This is due to the fact that in our problem the patterns involved are of small length and hence the overhead of FFT algorithm is quite high resulting in worse performance even from the naive approach.

4.1 Main Results

For the testing we tried to find the base codons in generated DNA sequences. Since the algorithm searches for the transpositions of consecutive characters, we only have to search for 40 of the 64 codons.

We first ran the algorithms using a single codon and randomly generated biological texts of increasing lengths. The results are shown in Figure 5. The results of the tests using the full set of codons needed for all matches (40 for the Swap match and 64 for the Naive match) are shown in Figure 6. The graphs show in both cases that the swapped matching runs in almost linear time.

4.2 Other Results

In this section, we show the effectiveness of the graph model swap matching algorithm when used to find a fixed pattern in an arbitrary text. The results of Test 1, 2 and 3 are shown in Figures 7, 8 and 9 respectively. They show how the algorithm performs against a naive approach for various patterns and texts.

In Test 1, we used a generated text which contained many swapped occurrences of the pattern in it. It can be seen clearly that in this case, the swap matching algorithm does much better than the naive algorithm.

In Test 2, we used a generated text which contained few swapped occurrences of the pattern in it. The graph clearly shows that in this case also, the swap matching algorithm performs much better than the naive algorithm.

In Test 3, we used a combination of the above texts. We kept the length of the text fixed, and used increasing pattern lengths. It can be seen from Figure 9 that

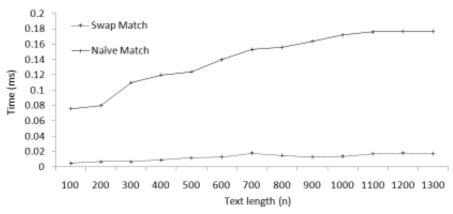


Fig. 7. Matching in text with many occurrences of pattern

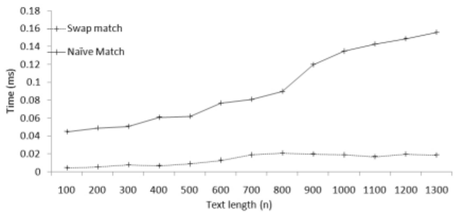


Fig. 8. Matching in text with few occurrences of pattern

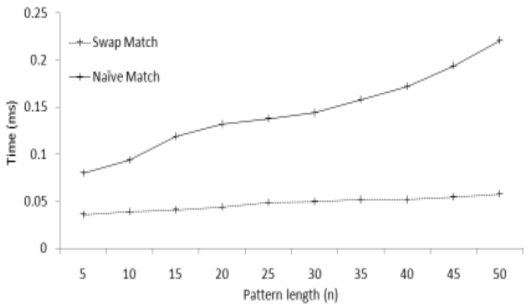


Fig. 9. Matching with fixed length text and increasing pattern length

the swap matching algorithm outperforms the naive algorithm, irrespective of the length of the pattern. This is because when the length of the pattern increases, the swap matching algorithm only increase its running time in the preprocessing stage. However, in the naive approach, increasing the pattern length leads to an overall increase in the running time. We remark that, it might be of interest to see how the FFT algorithms perform for Test 3, specifically for the lengthy patterns, because, the theoretical analysis suggests that, for higher length patterns the algorithm of [2] should outperform the IR algorithm. However, we didn't carry out that experiment because our main focus was to experimentally analyze the IR algorithm which works best when the pattern size is small.

5 Conclusion

In this paper, we have implemented and tested the algorithm to solve the swapped pattern matching presented in [6] and have presented detailed experimental results. We have also compared the running times of it against a naive approach. We have then analyzed the theoretical time complexity bounds with the actual running times achieved by the experiments and compare the results of the two algorithms.

References

1. Amir, A., Aumann, Y., Landau, G.M., Lewenstein, M., Lewenstein, N.: Pattern matching with swaps. J. Algorithms 37(2), 247–266 (2000)
2. Amir, A., Cole, R., Hariharan, R., Lewenstein, M., Porat, E.: Overlap matching. Inf. Comput. 181(1), 57–74 (2003)
3. Amir, A., Landau, G.M., Lewenstein, M., Lewenstein, N.: Efficient special cases of pattern matching with swaps. Inf. Process. Lett. 68(3), 125–132 (1998)
4. Baeza-Yates, R., Gonnet, G.: A new approach to text searching. Communications of the ACM 35, 74–82 (1992)
5. Charras, C., Lecroq, T.: Handbook of Exact String Matching Algorithms. In: Texts in Algorithmics, King's College, London (2004)
6. Iliopoulos, C.S., Rahman, M.S.: A new model to solve the swap matching problem and efficient algorithms for short patterns. In: Geffert, V., Karhumäki, J., Bertoni, A., Preneel, B., Návrat, P., Bieliková, M. (eds.) SOFSEM. LNCS, vol. 4910, pp. 316–327. Springer, Heidelberg (2008)
7. Strachan, T., Read, A.P.: Human Molecular Genetics 3. Garland Science, New York (2003)

Suffix Tree Characterization of Maximal Motifs in Biological Sequences

Maria Federico[1,3] and Nadia Pisanti[2,3]

[1] Dip. di Ingegneria dell'Informazione, Univ.di Modena e Reggio Emilia, Italy
[2] Dip. di Informatica, Univ. di Pisa, Italy
[3] Supported by the MIUR project MAIN STREAM

Abstract. Finding motifs in biological sequences is one of the most intriguing problems for string algorithms designers as it is necessary to deal with approximations and this complicates the problem. Existing algorithms run in time linear with the input size. Nevertheless, the output size can be very large due to the approximation. This makes the output often unreadable, next to slowing down the inference itself. Since only a subset of the motifs, i.e. the *maximal* motifs, could be enough to give the information of all of them, in this paper, we aim at removing such redundancy. We define notions of maximality that we characterize in the suffix tree data structure. Given that this is used by a whole class of motifs extraction tools, we show how these tools can be modified to include the maximality requirement on the fly without changing the asymptotical complexity.

Keywords: Suffix trees, Maximal Motifs, Biological Sequences.

1 Introduction

Finding frequent patterns (motifs) in biological sequences has myriads of applications in molecular biology. Following the hypothesis that sequence similarity is often a necessary condition for function correlations, there have been suggested in literature many versions, for as many various biological applications, of the problem of finding motifs as particularly frequent patterns in a biological sequence, or as patterns surprisingly shared by several distinct sequences. The motifs search is *approximated*, that is, distinct occurrences of the same motif are not necessarily identical, but just *similar*, according to a given similarity notion. From the computational complexity point of view, this makes the task of finding over-represented patterns harder, whatever is the type of approximation one uses. The *Hamming distance* is defined between patterns of the same length and it simply consists of the number of differences that occur between them. One usually sets a maximum allowed distance and then requires that the motifs differ by at most that number of letters substitutions.

Finding approximate motifs is a computationally challenging task because the output itself can be very big, especially with the approximation: its size can be exponential with respect to a parameter that measures the approximation (the maximum distance, the degeneracy degree of the degenerated alphabet, the number of don't care symbols used, etc.). This is a big drawback that, next to make the inference task possibly too slow, often also leads to a poor usability of the results as they are too large to be investigated with a naked eye. The difficulty to make use of the results of some motifs finding tools is often due to the fact that there are many motifs that satisfy the requirements, while only some of them are significant or, more in general, only some of them contain enough information to actually represent all the others. In this paper, we aim at eliminating most of the redundancy that make unreadable the output of existing methods that find approximate motifs. We define some notions of maximality for exact and approximate motifs and give for all of them a characterization on the suffix tree data structure. This allows us to show how to adapt a whole class of algorithms based on suffix tree for which available tools exist, to infer maximal motifs only, without additional complexity. We thank Esko Ukkonen for sending us a copy of [11].

2 Preliminary Definitions

We consider strings that are finite sequences of characters drawn from an alphabet Σ. In particular we will focus out attention on the DNA alphabet $\Sigma = \{A,\ C,\ G,\ T\}$. We denote by $s[i]$ the character at position i in a string s and by $|s|$ the length of s. Consecutive characters of s form a *substring* of s. The substring of s that starts from position i and ends at position j is denoted by $s[i..j]$, where $1 \leq i \leq j \leq |s|$. Given a string x drawn from the same alphabet as s (or from one of its subsets), we say that $s[i..j]$ *exactly occurs* at position i in s if and only if $x = s[i, i+|x|-1]$. In this case, we also say that $s[i, i+|x|-1]$ is an *occurrence* of x in s. The Hamming distance between two strings x and y, denoted as $d_H(x,y)$, is the smallest number of letter substitutions that transform x into y (or vice versa as the distance is symmetric). Given an integer $e \geq 0$, we say that a substring y of a string s is an *e-occurrence* of a string x, if and only if $d_H(x,y) \leq e$. In this case we will also talk about an *approximate occurrence*, or simply an *occurrence*, of x in s. The list of all occurences of a pattern x in s is denoted by $L_{(e,x)}$ and is called *positions set.*

Definition 1. *Given a sequence s, a* quorum *$q \geq 2$, and $e \geq 0$, a pattern m is a* motif *iff $|L_{(e,m)}| \geq q$.*

If $e = 0$ we speak about *exact motifs*, because no differences between motifs and their occurrences are allowed; otherwise, when $e > 0$, we call them *approximate motifs.* The traditional motifs extraction problem gives as input: (i) the string in which one wants to find the repeated motif (or the set of strings in which one wants to find the common motif); (ii) the quorum; (iii) the (minimal) length ℓ required for the motif; (iv) optionally, an approximation measure (e.g. the Hamming distance), and the value of e for the approximation measure. The requested output is simply the set of all patterns of length (at least) ℓ that have at least q (possibly approximated) occurrences in s, that is, the complete set of motifs. Within this traditional framework, the output can be very noisy as it contains redundant data. In this paper, we suggest a way to overcome this drawback by introducing a notion of maximality for motifs, thus identifying a subset of interesting representatives, and an efficient way to detect directly only those. To this purpose, we first introduce the notion of length extension of a motif. With *left extension* (resp. *right extension*) of m, we mean a pattern m' obtained by the concatenation of m with characters at its left (resp. right), and so such that m is a substring of m'. If there exists a right or left extension m' of a motif m which is also a motif, then we will say that m is *included* in m'.

Definition 2. *Let m and α be patterns of s. The pattern $m' = m\alpha$ (resp. $m' = \alpha m$) is a* mandatory right *(resp.* mandatory left*)* extension *of m iff all the occurrences of m in s are followed (resp. preceded) by α. In this case, we call $k = |\alpha|$ the* degree of the extension*.*

Definition 2 above is the same for both exact and approximate motifs. In the latter case, if m' is a mandatory right/left extension of m, then the two motifs m and m' have the same number of occurrences and also the total number of letters mismatches is the same because the right/left extension of m does not introduce further substitutions between the motif and its occurrences. It follows that if m is a motif, then also m' is a motif and vice versa. We will name *mandatory extension* (without specifying whether it is left or right) an extension on possibly both sides. We can observe that for a motif there exists at most one left and one right mandatory extension with a certain degree d. It is intuitive to observe that if the occurrences of a motif m are not all followed (resp. preceded) by the same character, then there can not be a mandatory right (resp. left) extension of degree 1 of m, and hence neither a mandatory right (resp. left) extension of higher degree can exist.

Notions of motif maximality have been defined in [3] for exact motifs. Furthermore, there have been notions of maximality defined for approximate motifs when the approximation is achieved using a degenerate alphabet, the edit distance, and don't care symbols. We give here a notion of maximality for approximate motifs with Hamming distance. Other (different) maximality notions for this type of approximate motifs exist in literature but are not meant for the general case as ours. In [4] the notion is restricted to the case of tandem repeats. The notion of maximality for motifs approximated with Hamming distance given in [5] does not apply to the whole occurrences set of the motif, but rather only for the special case of *repeats*, that are pairs of occurrences.

Definition 3. *A motif m is* right *(resp.* left*)* maximal *iff it has no mandatory right (resp. left) extension of degree 1. A motif m is* maximal *iff it is both right maximal and left maximal.*

For the particular case of exact motifs and $q = 2$, this notion of maximality coincides with that already introduced in [3]. This maximality property may not be enough to significantly bound the number of motifs. It can thus be useful to use an even more strict notion of maximality in extension of a motif (also already introduced in [3] for the special case of $q = 2$ and exact motifs). Moreover, in some applications long patterns with few occurrences can be more interesting of short patterns with a lot of occurrences. For these reasons, we formulate the notion of *supermaximality*[1].

Definition 4. *A motif m is* supermaximal *iff it is not a substring of another motif.*

In other words, a motif is supermaximal if, as one tries to extend it in any way, then the quorum property does not hold anymore. Note that the supermaximal motifs are a subset of the maximal ones.

3 Characterization of Motif Maximality on Suffix Tree

In this section we will give a characterization of maximal and supermaximal motifs of a string s on the suffix tree of the string itself. We will do this both for exact and for approximate motifs. For a formal definition of the suffix tree and its properties, we remind to [3]. We just recall here that it is a tree data structure that indexes a text such that there is a root-leaf path per each suffix of the text, and thus each root-node path for a node u corresponds to a substring of the text, to which we will refer to *the word spelled by* u, or *path-label of* u. Given that for some substrings the path does not end at a node but rather inside an edge, we will also talk about *the word spelled by a path*.

For exact motifs, in [3] there is already a characterization of maximal and supermaximal motifs on suffix trees for the special case in which $q = 2$, that is, when any pattern occurring two times or more is a motif. For the purpose of the suffix tree characterization, setting the quorum equal to two simplifies the task because every internal node corresponds to a pattern that satisfies the quorum and thus it is a motif. In this section, we first describe Gusfiled's result and then show the trivial generalization to the case of $q \geq 2$ that involves a search in a specific area of the suffix tree of the input string. Let s be a sequence and T its suffix tree. It is known that T can be built in linear time and space ([7,10]). Let us start with observing that an exact motif m, not necessarily maximal, labels a single path on suffix tree from the root that can end at an internal node or inside an edge having an internal node as destination. Using the mildest possible repetitiveness constraint ($quorum = 2$) Gusfield in [3] showed that there exists linear time algorithm based on suffix tree to find all maximal and supermaximal motifs. We now briefly describe Gusfield's result for maximal exact motifs. Each internal node of T, has at least two children and the edges out of an internal node are labeled with nonempty substrings of s that start with different characters. Therefore, if m labels an internal node it means that there are at least two occurrences of m in s followed by different characters, and hence m is right maximal. On the contrary, if m labels a path which ends inside an edge, then all its occurrences are followed by the character at depth $(|m| + 1)$ along that edge in T, and hence m has a mandatory right extension and thus it is not maximal.

[1] Being our definition the natural extension of that in [3] to the case of approximate motifs, we keep the same name.

The *left character of a leaf* of T is the character preceding the suffix of s at the position (where starts the suffix) represented by that leaf. A node v is *left diverse* if at least two leaves in the subtree rooted at v have different left characters, so if m labels a left diverse node it means that there are at least two occurrences of m in s preceded by different characters. Recall that, by definition, a leaf cannot be left diverse ([3]). Gusfield indeed proves that maximal motifs label exactly paths on the suffix tree that start from the root and lead to left diverse nodes of T. Let us now show how to extend the idea to the case of maximal exact motifs m occurring at least $q \geq 2$ of times in the input sequence. In the suffix tree T, for any internal node v that spells the pattern m, the positions of occurrences of m in the input sequence are represented by the (starting positions of the suffixes that label the) leaves in the subtree rooted at v. For each internal node v, let $Lv[v]$ denote the number of leaves in the subtree rooted at v. If v is a leaf, then we set $Lv[v] = 1$. It is possible to annotate all the internal nodes of T with the value of $Lv[v]$ within the linear time complexity by a simple traversal of the suffix tree (as shown in [9]). A pattern is a motif if it labels a path on the suffix tree that ends to an internal node v such that $Lv[v] \geq q$, or inside an edge that ends at an internal node for which this is the case. A motif is right maximal if it labels a path on T that ends at an internal node, and it is left maximal if such node is left diverse. Summing up, an exact motif is maximal if and only if it labels an internal node v of T such that $Lv[v] \geq q$ and v is left diverse, because right and left mandatory extensions of degree 1 can not exist for m. Gusfield also provides a characterization on suffix tree of supermaximal exact motifs, for the special case in which $q = 2$. If an exact motif m labels a path ending inside an edge, then it is not supermaximal either, because it has a right extension of degree 1 which is a motif preserving the same occurrences of m. Gusfield furthermore proves that supermaximal exact motifs label internal nodes v of T such that all the children of v are leaves and each has a distinct left character. In such case, indeed, a motif can not be furtherly extended without breaking the quorum constraint, because none of its extensions has two occurrences. This idea can be extended to the case of supermaximal exact motifs occurring at least $q \geq 2$ times in the input sequence. For this purpose, we introduce the notion of *right* and *left q-limited node.*

Definition 5. *Let T be the suffix tree for the sequence s. A node v of T that spells a pattern m is* right q-limited *(resp.* left q-limited*) iff m has no right (resp. left) extension of degree 1 which occurs at least q times in s. We say that a node is* q-limited *if it is right q-limited and left q-limited.*

Note that *not* being right q-limited (or left q-limited) is a property that propagates upward: if an internal node v is not right (resp. left) q-limited, then neither is any of its ancestors in the tree. Clearly, if $Lv[v] < q$ then v is q-limited. A right extension of degree 1 of m can label a child of v or a path ending inside an edge with a child of v as destination. It follows that v is right q-limited if and only if its children are nodes v' such that $Lv[v'] < q$.

The characterization on the suffix tree of the left q-limited property is a bit less immediate and it involves the so-called suffix link [3], that is a pointer that connects a node v spelling ax (with $a \in \Sigma$ and $x \in \Sigma^*$) to the node u that spells x. In other words, this pointer provides a link from node v to node u such that v spells a left extension of degree 1 of the path-label of u. Note that each pattern x has at most $|\Sigma|$ left extensions of degree 1 and thus a node u that spells x is reached by at most as many suffix links. Moreover, the left extensions of x do not always label nodes, but they can also label paths ending inside edges. Node u is reached by as many suffix links as the number of left extensions of degree 1 of x which label a node. If there exists a left extension which labels a path ending inside an edge that leads to node v', then there exists a suffix link from v' to a descendant of u whose left extension of degree 1 is the path-label of v'. It follows that if all the left extensions of degree 1 of the pattern spelled by a node label paths ending inside edges, then this node is not reached by any suffix link. In particular, we can observe that an internal node u, labeled by x, is reached by a suffix link only if at least two of its children are such that both their path-labels are preceded by the same character α at some (possibly all) of their occurrence positions in the input sequence. Moreover, there exists only one suffix link directed to the leaf node representing the suffix at position i in the input sequence s and it starts from the leaf representing the suffix at position $i - 1$ in s.

Due to what we just showed about suffix links and to the fact that not being left q-limited is a property that propagates upward in the tree, we observe that a node v is left q-limited only if $Lv[u] < q$ holds for each node u from which a suffix link to v originates and its children are left q-limited nodes.

Theorem 1. *Given a quorum q, a sequence s and its suffix tree T, the pattern m labeling the path to a node v of T is a supermaximal exact motif iff $Lv[v] \geq q$ and v is q-limited.*

Considering that the distinct occurrences of an approximate motif label different paths on suffix tree, the characterization on suffix tree provided for maximal exact motifs can be simply extended to approximate motifs.

Theorem 2. *Let T be the suffix tree for string s. An approximate motif m is right maximal iff:*

1. *at least an occurrence-path of m labels a node of T, or*
2. *all the occurrence-paths of m end inside edges of T and at least two of them have different characters at depth $(|m| + 1)$.*

Reminding that if a node of T is not left diverse then all the leaves in its subtree have the same left character, we give a characterization also for the left maximality of approximate motifs.

Theorem 3. *Let T be the suffix tree for a string s. An approximate motif m is left maximal iff:*

1. *at least an occurrence-path of m labels a left diverse node or ends inside an edge ending at a left diverse node, or*

2. *there are at least two distinct occurrence-paths of m ending (inside an edge ending) at a node that is not left diverse, and such that the leaves reached following these paths have not the same left character.*

Let us now show the characterization on suffix tree of supermaximal approximate motifs. In the case of exact motifs we used the notion of right and left q-limited node to verify supermaximality of a motif. When we consider approximate motifs, this notion does not suffice because they could have multiple occurrence-paths. Nevertheless, we can observe that if there exists an occurrence-path of an approximate motif m ending at an internal node which is not q-limited or inside an edge with a destination node that is not q-limited, then m is not supermaximal. In order to check whether an approximate motif can be extended keeping on satisfying the quorum constraint, one must take into account, per each occurrence, the number of mismatches with the motif. In details, given the mismatches threshold e, let x be an occurrence of a motif m with positions set $L_{(e,x)} = \{p_{x_1}, \ldots, p_{x_h}\}$ and such that $d_H(m, x) = d$. The positions set of the right (resp. left) extension $m\alpha$ (resp. αm) of m with any character $\alpha \in \Sigma$ includes positions p_{x_i} (resp. $p_{x_i} - 1$) where x is followed (resp. preceded) by:

- α, if $d = e$; the other occurrences are lost because the mismatches threshold e is exceeded, or
- any $\beta \in \Sigma$, if $d < e$; if $\beta \neq \alpha$, an extra mismatch between the extension $m' = m\alpha$ (resp. αm) and its occurrences $x\beta$ (resp. βx) is introduced.

If x labels a path ending inside an edge (u, v), or at an internal node u and let v be any child of u, then $m' = m\alpha$ has an occurrence-path ending inside (u, v) or at v, only if the character at string-depth $(|m| + 1)$ along (u, v) in T is α, or if it is $\beta \neq \alpha$ and $d < e$. Concerning left extensions, we have to follow suffix links directed to u and to its descendants. Actually, not all these suffix links are interesting but only those whose source node is not a child of a node from which a suffix link directed to u, or to a descendant closest to u, comes. Denoted by $sln[u]$ the set of these nodes, $m' = \alpha m$ has an occurrence-path ending at string-depth $(|m| + 1)$ along the edge ending at any node $v \in sln[u]$, only if the label of v starts with α, or if it starts with $\beta \neq \alpha$ and $d < e$.

Theorem 4. *An approximate motif m is supermaximal iff, for every $\alpha \in \Sigma$, $m' = \alpha m$ (resp. $m' = \alpha m$) has occurrence-paths ending at nodes, or inside edges with destination nodes, $u_1{}', \ldots, u_h{}'$ such that $\sum_{i=1}^{h} Lv[u_i{}'] < q$.*

4 Inferring Maximal Motifs with Suffix Tree

The first (exact) algorithm working on suffix trees was introduced for the extraction of motifs with mismatches in [9]. Motifs are considered in lexicographical order starting from the empty word, and they are extended on the right as long as the quorum is satisfied until either a valid motif of maximal length is found (if the required length is reached), or the quorum is no longer satisfied. At each moment, all paths spelling approximated occurrences of the current motif are taken into account. The number of the motif's occurrences is then computed as the sum of $Lv[v]$ for all nodes or destination nodes v of edges at which such occurrence-paths end. The algorithm exploits the property that these occurrence-paths on the suffix tree are such that those of the motif $m\alpha$ (where $m \in \Sigma^*$ and $\alpha \in \Sigma$) are found just going along the occurrence-paths of m and checking whether there is a character α following or a new mismatch can be introduced. Assuming that the required length of the motif is ℓ, and that at most e mismatches are allowed, the algorithm has worst case time complexity in $O(t_\ell \nu(e, \ell))$, where t_ℓ is the number of tree nodes at depth ℓ, and $\nu(e, \ell)$ is the number of words of length ℓ that differ in at most e letters from a word m of length ℓ. Finally, the space complexity is $O(t_\ell)$. The algorithm above was extended in [6] to the case of *structured motifs*, that is motifs composed of two or more parts lying at a certain given distance. The resulting tool, named SMILE, was applied to promoter signals detection in [12]. Moreover, using a data structure which is an enriched version of the suffix tree, basically the same framework has been used in [1]. Finally, the tool presented in [8], which resulted [2] to have very good performances, uses an algorithm that is basically an heuristic version of [9]. The definitions of maximality introduced in this paper and the characterizations on the suffix tree can be used by all the algorithms and tools mentioned above to output directly only (super)maximal motifs, with the consequent obvious improvement in readability and significance. Moreover, our results also apply to the versions of the problem that require motifs *common* to a set of input strings (rather than repeated within the same unique input string).

We now provide a brief description of the operations needed to extend existing motif discovery algorithms in order to extract only (super)maximal motifs and we show that the additional complexity cost due to motif (super)maximality check is negligible. Let us first consider maximal exact motif extraction. In [3], Gusfield presents a linear time algorithm to find left diverse nodes of a suffix tree T. A bottom-up traversal of T is performed, and, for each node v, the algorithm stores either that v is left diverse, or the common left character of every leaf in the subtree rooted at v. Hence, assuming that we have a suffix tree whose nodes are annotated with this information, the additional cost to select only maximal motifs among all exact motifs is constant: for every motif found, it is enough to verify whether it labels a left diverse node of T. If we search for maximal approximate motifs m, the existing extraction algorithms can be simply extended in order to test out the conditions of Theorems 2 and 3. For every occurrence-path of m, if it ends at a node v or else if it ends inside an edge with destination node v and the character at string-depth $(|m| + 1)$ along that edge is different from the one of already identified occurrence-paths of m that end inside edges, then m is right maximal. Moreover, if v is a left diverse node, or else if the left character with which v is annotated is different from the one of already identified occurrence-paths of m, then m is left maximal. In both cases, the condition to check consists of two character comparisons, and then the additional cost to extract only maximal approximate motifs is constant.

Consider now the extraction of supermaximal exact motifs. All nodes v of T can be annotated with right and left q-limited property by a simple bottom-up traversal of T. Therefore, like for maximal motifs, the additional cost to extract only supermaximal exact motifs is constant. The extension of existing motif discovery algorithms in order to extract only supermaximal approximate motifs is a little more complex. It requires that for every node v of T, in addition to right and left q-limited information, we also have the set $sln[v]$ of nodes from which suffix links directed to v or to its descendants originate. This set can be implemented by an array of $|\Sigma|$ positions. At position i of $sln[v]$ there is the node from which a suffix link to v (or to one of its descendant) originates, and whose label starts with the ith character in Σ. Assuming to have at start, for every node v of T, the array $sln[v]$ storing only nodes from which suffix links directed to v come (if any), that can be built within the linear time complexity of suffix tree construction, the complete set $sln[v]$ for all nodes of T can be found with a bottom-up traversal of T with an additional cost of $|\Sigma|^2$ per each node. The existing extraction algorithms can be extended to output only supermaximal approximate motifs m in the following manner. For every occurrence-path of m identified on the suffix tree T, if it ends at a node that is not right or left q-limited or inside an edge with a destination node of this type, then m is not supermaximal. Moreover, m is not supermaximal if the number of its occurrences which are at Hamming distance strictly less than e from m exceeds the quorum q, because in this case all right and left extensions of m preserve such occurrences and so they are motifs as well. These checks introduce a constant additional cost to the extraction algorithms. After finding all the occurrence-paths of m, if none of the described cases is verified, then, if m is a motif, we must also count, for every character α in Σ, how many occurrences of m at Hamming distance equal to e from m are preserved by right and left extensions of m with α. This counting can be made examining occurrence-paths of m labeled by such occurrences as showed in Section 3. If there exists a right or left extension of degree 1 of m which occurs more than q times, then m is not supermaximal. Therefore the operations needed to count the occurrences preserved by every right and left extension of m with a character α have a cost proportional to the number of occurrence-paths of m whose label is at Hamming distance e from m. This is due to the property of the suffix tree such that the edges from a node (at most $|\Sigma|$) are lexicographically sorted and to the implementation of set $sln[v]$ as an array of $|\Sigma|$ positions: the cost for each path is constant because, in the worst case, it simply consists of a comparison between two characters. If ℓ is the motif length, these occurrence-paths are at most $p = \binom{\ell}{e} \cdot (|\Sigma| - 1)^e$ and hence the additional cost to verify if a motif is supermaximal is proportional to $O(p|\Sigma|)$. Moreover, notice that if a motif is supermaximal, then the extraction algorithm can avoid to furtherly extend it, because none of its right extensions can be a motif. Thus, the overhead introduced by the supermaximality check is balanced by a reduction of the number of intermediate length motifs that have to be extended during the extraction process. We expect that the approach we suggested could sensibly reduce the number of output motifs without changing their significance and with a constant or negligible extra time complexity. However, we are aware that in a worst case scenario the improvement could be none: one can always design a string in which all motifs are maximal. Nevertheless, biological sequences contain many more repetitions than randomly generated sequences which, on their turn, averagely would be far from containing only maximal motifs.

5 Conclusions

In order to remove the redundancy in the output of existing algorithms for finding motifs, we defined notions of (super)maximality for exact and approximate motifs. For all of them we gave a characterization on the suffix tree data structure. This allowed us to show how to adapt a whole class of algorithms based on suffix tree for which available tools exist, to infer (super)maximal motifs only. We proved that the additional computational cost due to the on the fly check of (super)maximality requirements is negligible. Therefore, our results suggest a way to improve motifs extraction tools providing outputs which are more readable and usable by biologists.

References

1. Carvalho, A.M., Freitas, A.T., Oliveira, A.L., Sagot, M.-F.: An efficient algorithm for the identification of structured motifs in dna promoter sequences. IEEE/ACM Trans. Comput. Biology Bioinform. 3(2), 126–140 (2006)
2. Tompa, M., et al.: Assessing computational tools for the discovery of transcription factor binding sites. Nature Biotechnology 23(1), 137–144 (2005)
3. Gusfield, D.: Algorithms on strings, trees, and sequences: computer science and computational biology. Cambridge University Press, New York (1997)
4. Kolpakov, R.M., Kucherov, G.: Finding approximate repetitions under hamming distance. In: ESA, pp. 170–181 (2001)
5. Kurtz, S., Ohlebusch, E., Schleiermacher, C., Stoye, J., Giegerich, R.: Computation and visualization of degenerate repeats in complete genomes. In: ISMB, pp. 228–238 (2000)
6. Marsan, L., Sagot, M.-F.: Algorithms for extracting structured motifs using a suffix tree with application to promoter and regulatory consensus identification. J. Comp. Bio. 7, 345–360 (2001)
7. McCreight, E.: A space-economical suffix tree construction algorithm. J. ACM 23(2), 262–272 (1976)
8. Pavesi, G., Mereghetti, P., Mauri, G., Pesole, G.: Weeder web: discovery of transcription factor binding sites in a set of sequences from co-regulated genes. Nucleic Acids Research 32, 199–203 (2004)
9. Sagot, M.-F.: Spelling approximate repeated or common motifs using a suffix tree. In: LATIN, pp. 374–390 (1998)
10. Ukkonen, E.: On-line construction of suffix trees. Algorithmica 14(3), 249–260 (1995)
11. Ukkonen, E.: Structural analysis of gapped motifs of a string. In: MFCS, pp. 681–690 (2007)
12. Vanet, A., Marsan, L., Labigne, A., Sagot, M.-F.: Inferring regulatory elements from a whole genome. an application to the analysis of the genome of helicobacter pylori sigma 80 family of promoter signals. J. of Mol. Biol. 297, 335–353 (2000)

Efficient Seeding Techniques for Protein Similarity Search

Mikhail Roytberg[1], Anna Gambin[2], Laurent Noé[3], Sławomir Lasota[2], Eugenia Furletova[1], Ewa Szczurek[4], and Gregory Kucherov[3]

[1] Institute of Mathematical Problems in Biology, Pushchino, Moscow Region, 142290, Russia
mroytberg@mail.ru, furletova@impb.psn.ru
[2] Institute of Informatics, Warsaw University, Banacha 2, 02-097, Poland
{aniag,S.Lasota}@mimuw.edu.pl
[3] LIFL/CNRS/INRIA, Bât. M3, Campus Scientifique, 59655 Villeneuve d'Ascq Cédex, France
{Gregory.Kucherov,Laurent.Noe}@lifl.fr
[4] Max Planck Institute for Molecular Genetics, Computational Molecular Biology, Ihnestr. 73, 14195 Berlin, Germany
ewa.szczurek@molgen.mpg.de

Abstract. We apply the concept of *subset seeds* proposed in [1] to similarity search in protein sequences. The main question studied is the design of efficient *seed alphabets* to construct seeds with optimal sensitivity/selectivity trade-offs. We propose several different design methods and use them to construct several alphabets. We then perform an analysis of seeds built over those alphabet and compare them with the standard Blastp seeding method [2,3], as well as with the family of vector seeds proposed in [4]. While the formalism of subset seed is less expressive (but less costly to implement) than the accumulative principle used in Blastp and vector seeds, our seeds show a similar or even better performance than Blastp on Bernoulli models of proteins compatible with the common BLOSUM62 matrix.

1 Introduction

Similarity search in protein sequences is probably the most classical bioinformatics problem, and a commonly used algorithmic solution is implemented in the ubiquitous Blast software [2,3]. On the other hand, similarity search algorithms for nucleotide sequences (DNA, RNA) underwent several years ago a significant improvement due to the idea of *spaced seeds* and its various generalizations [5,6,7,8,9,10,11]. This development, however, has little affected protein sequence comparison, although improving the speed/precision trade-off for protein search would be of great value for numerous bioinformatics projects. Due to a bigger alphabet size, protein seeds are much shorter (typically 2-5 letters instead of 10-20 letters in the DNA case) and also letter identity is much less relevant in defining hits than in the DNA case. For these reasons, the spaced seeds technique might seem not to apply directly to protein sequence comparison.

Recall that BLAST applies quite different approaches to protein and DNA sequences to define a hit. In the DNA case, a hit is defined as a short pattern of identically matching nucleotides whereas in the protein case, a hit is defined through an *accumulative* contribution of a few amino acid matches (not necessarily identities) using a given *scoring matrix*. Defining a hit through an additive contribution of several positions is captured by a general formalism of *vector seeds* proposed in [7]. On the other hand, it has been understood [6,12,13,14,15] that using simultaneously a *family* of seeds instead of a single seed can further improve the sensitivity/selectivity ratio. Papers [4,16] both propose solutions using a family of vector seeds to surpass the performance of BLAST.

However, using the principle of accumulative score over several adjacent positions has an algorithmic cost. Defining a hit through a pattern of exact letter matches allows for a *direct hashing* scheme, where each key of the query sequence is associated with a *unique* hash table entry storing positions of the subject sequence (database) where the key can hit. On the other hand, defining a hit through an accumulative contribution of several positions leads to an additional pre-computed table storing, for each key, its *neighborhood* i.e., the list of subject keys (or corresponding hash table entries) with which it can form a hit. For example, in a standard BLASTP setting (Blosum62 scoring matrix with threshold 11 for accumulative score of 3 contiguous positions) a 3-letter key hits on average 19.34 distinct keys, i.e. requires that many accesses to the hash table. For the family of vector seeds from [4] with an equivalent selectivity level (score 18), a (here 4-letter) key hits on average 15.99 keys. For some applications, for example in setting large-scale protein comparisons on a specialized computer architecture (see e.g. [17]) one might need to minimize the number of hash table accesses, and therefore to use another seeding formalism.

In [1], we proposed a new concept of *subset seeds* that can be viewed as an intermediate between ordinary spaced seeds and vector seeds: subset seeds allow one to distinguish between different types of mismatches (or matches) but still treat seed positions independently rather than cumulatively. Distinguishing different mismatches is not done by scoring them, but by extending the seed alphabet such that each seed letter specifies different sets of mismatches. For example, in the DNA case it is beneficial to distinguish between transition mutations ($\mathtt{A} \leftrightarrow \mathtt{G}$, $\mathtt{C} \leftrightarrow \mathtt{T}$) and others (transversions) [18].

Since the protein alphabet is much larger than the one of DNA, subset seeds provide a very attractive seeding option for protein alignment. The present study is then motivated by following general questions: *how far can we go with subset seeds applied to protein sequences? Can we reach the performance of* BLAST *seeds? the one of vector seeds? or maybe even outperform them? ...*

In the paradigm of subset seeds, each seed letter specifies a set of amino acid pairs matched by this letter. Therefore, a crucial question is the design of an appropriate *seed alphabet*, which is one of the main problems we study in this paper. In Section 2, we introduce some probabilistic notions we need to reason about seed efficiency. Section 3 introduces the first simple approach to design a seed alphabet, which, however, does not lead to so-called *transitive* seeds, useful in practice. Section 4 presents three different approaches to designing transitive seed alphabets, based on a pre-defined (Section 4.1) or newly designed (Section 4.2) hierarchical clustering of amino acids, as well as on a non-hierarchical clustering (Section 4.3). Section 5 describes comparative experiments made with the designed seeds on probabilistic models.

2 Preliminaries

Throughout the paper, $\Sigma = \{\mathtt{A}, \mathtt{C}, \mathtt{D}, \mathtt{E}, \mathtt{F}, \mathtt{G}, \mathtt{H}, \mathtt{I}, \mathtt{K}, \mathtt{L}, \mathtt{M}, \mathtt{N}, \mathtt{P}, \mathtt{Q}, \mathtt{R}, \mathtt{S}, \mathtt{T}, \mathtt{V}, \mathtt{W}, \mathtt{Y}\}$ denotes the alphabet of amino acids.

In most general terms, a *(subset) seed letter* α is defined as any symmetric and reflexive binary relation on Σ. Let $\mathcal{B}$ be a *seed alphabet*, i.e. a collection of subset seed letters. Then a *subset seed* $\pi = \alpha_1 \ldots \alpha_k$ is a word over $\mathcal{B}$. π defines a symmetric and reflexive binary relation on words of Σ^k (called *keys*): for $s_1, s_2 \in \Sigma^k$, $s_1 \sim_\pi s_2$ iff $\forall i \in [1..k]$, we have $\langle s_1[i], s_2[i] \rangle \in \alpha_i$.

For practical reasons, we would like seed letters to define a *transitive* relation, in addition. This induces an equivalence relation on keys, which is very convenient and allows for an efficient indexing scheme (see Introduction). In this paper, we will be mainly interested in transitive seed letters, but we will also study the non-transitive case in order to see how restrictive the transitivity condition is.

The quality of a seed letter or of a seed is characterized by two main parameters: *sensitivity* and *selectivity*. They are defined through background and foreground probabilistic models of protein alignments. Foreground probabilities are assumed to represent the distribution of amino acids matches in proteins of interest, when two homologous proteins are aligned together. Background probabilities, on the other hand, represent the distribution of amino acid matches in *random alignments*, when two proteins are randomly aligned together.

In this paper, we restrict ourselves to Bernoulli models of proteins and protein alignments, although some of the results we will present can be extended to Markov models.

Assume that we are given background probabilities $\{b_1, \ldots, b_{20}\}$ of amino acids in protein sequences under interest. The *background probability* of a seed letter α is defined by $b(\alpha) = \sum_{(a_i,a_j)\in\alpha} b_i b_j$. The *selectivity* of α is $1 - b(\alpha)$ and the *weight* of α is defined by

$$w(\alpha) = \frac{\log b(\alpha)}{\log b(\#)}, \tag{1}$$

where $\# = \{\langle a, a \rangle | a \in \Sigma\}$ is the "identity" seed letter. For a seed $\pi = \alpha_1 \ldots \alpha_k$, the background probability of π is $b(\pi) = \prod_{i=1}^{k} b(\alpha_i)$, the selectivity of π is $1 - b(\pi)$ and the weight of π is $w(\pi) = \log_{b(\#)} b(\pi) = \sum_{i=1}^{k} w(\alpha_i)$. Note that the weight here generalizes the weight of of classical spaced seeds [19] defined as the number of "identity" letters it contains.

Let f_{ij} be the probability to see a pair $\langle a_i, a_j \rangle$ aligned in a target alignment. The *foreground probability* of a seed letter α is defined by $f(\alpha) = \sum_{(a_i,a_j)\in\alpha} f_{ij}$.

The *sensitivity* of a seed π is defined as the probability to hit a random target alignment[1]. Assume that target alignments are specified by a length N. Then the sensitivity of a seed $\pi = \alpha_1 \dots \alpha_k$ is the probability that a randomly drawn gapless alignment (i.e. string of pairs $\langle a_i, a_j \rangle$) of length N contains a fragment of length k which is matched by π. In [1] we proposed a general algorithm to efficiently compute the seed sensitivity for a broad class of target alignment models. This algorithm will be used in the experimental part of this work.

The general problem of seed design is to obtain seeds with good sensitivity/selectivity trade-off. Even within a fixed seed formalism, the quality of a seed is dependent on the chosen selectivity value. This is why we will always be interested in computing efficient seeds for a large range of selectivity levels.

3 Dominating Seed Letters

Our main question is how to choose seed letters that form good seeds? Intuitively, "good letters" are those that best distinguish foreground and background letter alignments.

For each letter α, consider its foreground and background probabilities $f(\alpha)$ and $b(\alpha)$ respectively. Intuitively, we would like to have letters α with large $f(\alpha)$ and small $b(\alpha)$. A letter α is said to *dominate* a letter β if $f(\alpha) \geq f(\beta)$ and $b(\alpha) \leq b(\beta)$. Observe that in this case, β can be removed from consideration, as it can be always advantageously replaced by α.

Consider all amino acid pairs (a_i, a_j) ordered by descending *likelihood ratio* $f_{ij}/b_i b_j$. Consider the set of pairs (a_i, a_j) such that $f_{ij}/b_i b_j > s$ for some threshold value s. Then one can show that this set forms a letter that cannot be dominated by any other letter[2] (proof omitted). This observation leads to defining seed letters that consist of those pairs (a_i, a_j) for which the ratio $f_{ij}/b_i b_j$ is above a given threshold.

Resulting Alphabet. We computed the likelihood ratio for all amino acid pairs, based on practical values of background and foreground probabilities computed in accordance with the BLOSUM62 matrix (see Section 5.1). Not surprisingly, amino acid identities (pairs $\langle a, a \rangle$) have highest likelihood scores varying from 38.11 for tryptophan down to 3.69 for valine. Among distinct pairs, only 25 have a score greater than 1 (Figure 1). A quick analysis shows that those do not form transitive relations, and therefore do not verify the transitivity requirement. The alphabet containing those 25 pairs is denoted `Non-transitive`. It will be used in the experimental part of the paper (Section 5) in order to study

[1] Note that our definitions of sensitivity and selectivity are not symmetric: sensitivity is defined with respect to the entire alignment and selectivity with respect to a single alignment position. These definitions capture better the intended parameters we want to measure. However, selectivity could also be defined with respect to the entire alignment. We could suggest the term *specificity* for this latter definition.

[2] It is interesting to point out the relationship to the well-known Neyman-Pearson lemma which is a more general formulation of this statement.

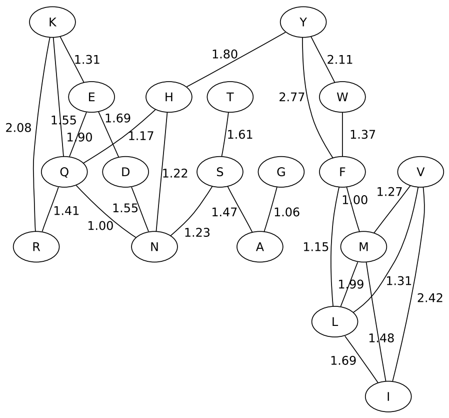


Fig. 1. Alphabet `Non-transitive`: amino acid pairs with likelihood ratio > 1

how restrictive is the requirement of transitive letters, i.e. how much better are general seeds than those obtained with the restriction of transitivity.

4 Transitive Seed Alphabets

In the case of transitive seed alphabets, every letter $\alpha \in \mathcal{B}$ is a partition of the amino acid alphabet Σ. In other words, the binary relation associated with each letter (cf Section 2) is an equivalence relation. Transitive alphabets represent the practical case when each amino acid is uniquely mapped to its equivalence class. This, in turn, allows for an efficient hashing scheme during the stage of seed search, when different entries of the hash table index non-intersecting subsets of keys.

In Sections 4.1,4.2, we explore transitive seed alphabets verifying an additional condition: for any two seed letters $\alpha_1, \alpha_2 \in \mathcal{B}$ corresponding to partitions $P_{\alpha_1}, P_{\alpha_2}$ respectively, one of $P_{\alpha_1}, P_{\alpha_2}$ is a refinement of the other. Formally, for any $\alpha_1, \alpha_2 \in \mathcal{B}$,

$$\text{either every set } \sigma \in P_{\alpha_1} \text{ is a subset of some } \delta \in P_{\alpha_2}, \text{ or vice versa.} \quad (2)$$

The purpose of the above requirement is to define seed letters using a biologically significant hierarchical clustering of amino acids represented by a tree. In Section 4.1, we will use a pre-defined hierarchical clustering to design efficient seed alphabets. Then in Section 4.2, we construct our own clustering based on appropriate background and foreground models of amino acids distribution. Finally, in Section 4.3 we lift condition (2) and study "non-hierarchical" seed alphabets.

4.1 Transitive Alphabets Based on a Pre-defined Clustering

Assume we have a biologically significant hierarchical clustering tree which is a rooted binary tree T with 20 leaves labelled by amino acids. Such trees have

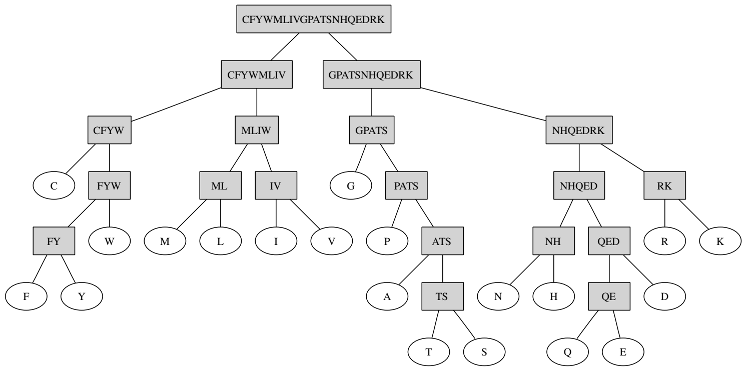


Fig. 2. Hierarchical tree derived from [20]

been proposed in [20,21], based on different similarity relations. The hierarchical tree derived from [20] is shown on Figure 2.

The tree, obtained with a purely bioinformatics analysis, groups together amino acids with similar biochemical properties, such as hydrophobic amino acids `L,M,I,V`, hydrophobic aromatic amino acids `F,Y,W`, alcohols `S,T`, or charged/polar amino acids `E,D,N,Q`. A similar grouping is obtained in [21].

A *seed letter* is defined here as a subset α of nodes of T such that

(i) α contains all leaves,
(ii) for a node v, if $v \in \alpha$, then all descendants of v belong to α too.

In other words, a seed letter can be thought of as a "horizontal cut" of the tree. For example, for the tree of Figure 2 there are 1597 different seed letters. Seed letters are naturally ordered by inclusion. The smallest one is the "identity" seed letter #, containing only the leaves. The largest one is the "joker" seed letter _, containing all the nodes of T. One particular seed letter is obtained by removing from _ the root node. We denote it by @.

Observe that each seed letter α represents naturally an equivalence relation on Σ: a_i and a_j are related iff their common ancestor belongs to α. It is identity relation in case of # and full relation in case of _.

Following condition (2), a *hierarchical seed alphabet* is a family $\mathcal{B}$ of seed letters such that

$$\text{for every } \alpha_1, \alpha_2 \in \mathcal{B},\ \text{either } \alpha_1 \subseteq \alpha_2 \text{ or } \alpha_2 \subseteq \alpha_1. \qquad (3)$$

Hence, a seed alphabet is a chain in the inclusion ordering of seed letters. Let us analyze what are the maximal seed alphabets wrt. inclusion. Clearly each maximal seed alphabet $\mathcal{B}$ always contains the smallest and the largest seed letters # and _. Interestingly, each maximal $\mathcal{B}$ contains also @, as @ is comparable (by inclusion) to any other seed letter.

It can be shown that any maximal seed alphabet contains exactly 20 letters that can be obtained by a stepwise merging of two subtrees rooted at immediate descendants of some node v into the subtree rooted at v. Therefore, since a

$$\begin{array}{c}
\{CFYWMLIVGPATSNHQEDRK\} \\
\{CFYWMLIV\}\ \{GPATSNHQEDRK\} \\
\{CFYWMLIV\}\ \{GPATS\}\ \{NHQEDRK\} \\
\{CFYW\}\ \{MLIV\}\ \{GPATS\}\ \{NHQEDRK\} \\
\{CFYW\}\ \{MLIV\}\ \{G\}\ \{PATS\}\ \{NHQEDRK\} \\
\{C\}\ \{FYW\}\ \{MLIV\}\ \{G\}\ \{PATS\}\ \{NHQEDRK\} \\
\{C\}\ \{FYW\}\ \{MLIV\}\ \{G\}\ \{P\}\ \{ATS\}\ \{NHQEDRK\} \\
\{C\}\ \{FY\}\ \{W\}\ \{MLIV\}\ \{G\}\ \{P\}\ \{ATS\}\ \{NHQEDRK\} \\
\{C\}\ \{F\}\ \{Y\}\ \{W\}\ \{MLIV\}\ \{G\}\ \{P\}\ \{ATS\}\ \{NHQEDRK\} \\
\{C\}\ \{F\}\ \{Y\}\ \{W\}\ \{MLIV\}\ \{G\}\ \{P\}\ \{A\}\ \{TS\}\ \{NHQEDRK\} \\
\{C\}\ \{F\}\ \{Y\}\ \{W\}\ \{MLIV\}\ \{G\}\ \{P\}\ \{A\}\ \{T\}\ \{S\}\ \{NHQEDRK\} \\
\{C\}\ \{F\}\ \{Y\}\ \{W\}\ \{MLIV\}\ \{G\}\ \{P\}\ \{A\}\ \{T\}\ \{S\}\ \{NHQED\}\ \{RK\} \\
\{C\}\ \{F\}\ \{Y\}\ \{W\}\ \{MLIV\}\ \{G\}\ \{P\}\ \{A\}\ \{T\}\ \{S\}\ \{NHQED\}\ \{R\}\ \{K\} \\
\{C\}\ \{F\}\ \{Y\}\ \{W\}\ \{MLIV\}\ \{G\}\ \{P\}\ \{A\}\ \{T\}\ \{S\}\ \{NH\}\ \{QED\}\ \{R\}\ \{K\} \\
\{C\}\ \{F\}\ \{Y\}\ \{W\}\ \{MLIV\}\ \{G\}\ \{P\}\ \{A\}\ \{T\}\ \{S\}\ \{N\}\ \{H\}\ \{QED\}\ \{R\}\ \{K\} \\
\{C\}\ \{F\}\ \{Y\}\ \{W\}\ \{MLIV\}\ \{G\}\ \{P\}\ \{A\}\ \{T\}\ \{S\}\ \{N\}\ \{H\}\ \{QE\}\ \{D\}\ \{R\}\ \{K\} \\
\{C\}\ \{F\}\ \{Y\}\ \{W\}\ \{MLIV\}\ \{G\}\ \{P\}\ \{A\}\ \{T\}\ \{S\}\ \{N\}\ \{H\}\ \{Q\}\ \{E\}\ \{D\}\ \{R\}\ \{K\} \\
\{C\}\ \{F\}\ \{Y\}\ \{W\}\ \{ML\}\ \{IV\}\ \{G\}\ \{P\}\ \{A\}\ \{T\}\ \{S\}\ \{N\}\ \{H\}\ \{Q\}\ \{E\}\ \{D\}\ \{R\}\ \{K\} \\
\{C\}\ \{F\}\ \{Y\}\ \{W\}\ \{M\}\ \{L\}\ \{IV\}\ \{G\}\ \{P\}\ \{A\}\ \{T\}\ \{S\}\ \{N\}\ \{H\}\ \{Q\}\ \{E\}\ \{D\}\ \{R\}\ \{K\} \\
\{C\}\ \{F\}\ \{Y\}\ \{W\}\ \{M\}\ \{L\}\ \{I\}\ \{V\}\ \{G\}\ \{P\}\ \{A\}\ \{T\}\ \{S\}\ \{N\}\ \{H\}\ \{Q\}\ \{E\}\ \{D\}\ \{R\}\ \{K\}
\end{array}$$

Fig. 3. Alphabet `Transitive-predefined` designed using the tree of Figure 2. Each line corresponds to a seed letter (amino acid partition).

binary tree with n leaves contains $n-1$ internal nodes, a maximal seed alphabet contains precisely 20 letters and can be specified by a permutation of internal nodes in tree T.

Resulting Alphabet. Figure 3 shows alphabet `Transitive-predefined` designed through the approach of this Section. The alphabet has been designed from the tree of Figure 2. Each line corresponds to a letter (amino acid partition). Among the alphabets obtained with different parameter values, alphabet `Transitive-predefined` produced better seeds and will be used in the experimental part of this work (Section 5).

4.2 Transitive Alphabets Using an *ab initio* Clustering Method

Hierarchical Clustering of Amino Acids. A prerequisite to the approach of Section 4.1 is a given tree describing a hierarchical clustering of amino acid based on some similarity measure. In this section, we describe an *ab initio* approach that constructs a hierarchical clustering of amino acids from scratch, using a likelihood measure.

As in Section 4, our goal here is to construct a family of seed letters verifying (3). This family will be obtained with a simple greedy neighbor-joining clustering algorithm, starting with the family of twenty amino acid singletons.

We start with the partition of amino acids into 20 singletons. This partition corresponds to the # letter. For a current partition $P = \{C_1, \ldots, C_n\}$, iteratively apply the following procedure.

```
1 For each pair of sets $C_k$, $C_\ell$,
  1.1 consider the set $Bridge(C_k, C_\ell) = \{(a_i, a_j) | a_i \in C_k,\ a_j \in C_\ell\}$.
  1.2 compute $ForeProb(k, \ell) = \sum\{f_{ij} | a_i \in C_k,\ a_j \in C_\ell\}$
      and     $BackProb(k, \ell) = \sum\{b_i b_j | a_i \in C_k,\ a_j \in C_\ell\}$,
  1.3 compute $L(k, \ell) = ForeProb(k, \ell)/BackProb(k, \ell)$
```

$\{CFYWHMLIVPGQERKNDATS\}$

$\{CFYWHMLIV\}\ \{PGQERKNDATS\}$

$\{C\}\ \{FYWHMLIV\}\ \{PGQERKNDATS\}$

$\{C\}\ \{FYWHMLIV\}\ \{P\}\ \{GQERKNDATS\}$

$\{C\}\ \{FYWH\}\ \{MLIV\}\ \{P\}\ \{GQERKNDATS\}$

$\{C\}\ \{FYWH\}\ \{MLIV\}\ \{P\}\ \{GATS\}\ \{QERKND\}$

$\{C\}\ \{FYWH\}\ \{MLIV\}\ \{P\}\ \{G\}\ \{ATS\}\ \{QERKND\}$

$\{C\}\ \{FYWH\}\ \{MLIV\}\ \{P\}\ \{G\}\ \{ATS\}\ \{QERK\}\ \{ND\}$

$\{C\}\ \{FYW\}\ \{H\}\ \{MLIV\}\ \{P\}\ \{G\}\ \{ATS\}\ \{QERK\}\ \{ND\}$

$\{C\}\ \{FYW\}\ \{H\}\ \{MLIV\}\ \{P\}\ \{G\}\ \{A\}\ \{TS\}\ \{QERK\}\ \{ND\}$

$\{C\}\ \{FYW\}\ \{H\}\ \{MLIV\}\ \{P\}\ \{G\}\ \{A\}\ \{TS\}\ \{QE\}\ \{RK\}\ \{ND\}$

$\{C\}\ \{FYW\}\ \{H\}\ \{ML\}\ \{IV\}\ \{P\}\ \{G\}\ \{A\}\ \{TS\}\ \{QE\}\ \{RK\}\ \{ND\}$

$\{C\}\ \{FYW\}\ \{H\}\ \{ML\}\ \{IV\}\ \{P\}\ \{G\}\ \{A\}\ \{TS\}\ \{QE\}\ \{RK\}\ \{N\}\ \{D\}$

$\{C\}\ \{FYW\}\ \{H\}\ \{ML\}\ \{IV\}\ \{P\}\ \{G\}\ \{A\}\ \{T\}\ \{S\}\ \{QE\}\ \{RK\}\ \{N\}\ \{D\}$

$\{C\}\ \{FY\}\ \{W\}\ \{H\}\ \{ML\}\ \{IV\}\ \{P\}\ \{G\}\ \{A\}\ \{T\}\ \{S\}\ \{QE\}\ \{RK\}\ \{N\}\ \{D\}$

$\{C\}\ \{FY\}\ \{W\}\ \{H\}\ \{ML\}\ \{IV\}\ \{P\}\ \{G\}\ \{A\}\ \{T\}\ \{S\}\ \{Q\}\ \{E\}\ \{RK\}\ \{N\}\ \{D\}$

$\{C\}\ \{FY\}\ \{W\}\ \{H\}\ \{M\}\ \{L\}\ \{IV\}\ \{P\}\ \{G\}\ \{A\}\ \{T\}\ \{S\}\ \{Q\}\ \{E\}\ \{RK\}\ \{N\}\ \{D\}$

$\{C\}\ \{FY\}\ \{W\}\ \{H\}\ \{M\}\ \{L\}\ \{I\}\ \{V\}\ \{P\}\ \{G\}\ \{A\}\ \{T\}\ \{S\}\ \{Q\}\ \{E\}\ \{RK\}\ \{N\}\ \{D\}$

$\{C\}\ \{F\}\ \{Y\}\ \{W\}\ \{H\}\ \{M\}\ \{L\}\ \{I\}\ \{V\}\ \{P\}\ \{G\}\ \{A\}\ \{T\}\ \{S\}\ \{Q\}\ \{E\}\ \{RK\}\ \{N\}\ \{D\}$

$\{C\}\ \{F\}\ \{Y\}\ \{W\}\ \{H\}\ \{M\}\ \{L\}\ \{I\}\ \{V\}\ \{P\}\ \{G\}\ \{A\}\ \{T\}\ \{S\}\ \{Q\}\ \{E\}\ \{R\}\ \{K\}\ \{N\}\ \{D\}$

Fig. 4. Alphabet `Transitive-ab-initio` obtained with the method of Section 4.2

2 Find the pair of sets (C_k, C_ℓ) yielding the maximal $L(k, \ell)$,
3 Merge C_k and C_ℓ into a new set, obtaining a new partition.

The rationale behind this simple procedure is that those two sets of amino acid are merged together which produce the maximal increment in the likelihood $f(\alpha)/b(\alpha)$. An alternative method, when the likelihood of the whole resulting set is maximized, yields biased results, as sets with a high likelihood tend to "absorb" other sets.

Resulting Alphabet. An alphabet, called `Transitive-ab-initio`, obtained with this greedy neighbor-joining approach is given in Figure 4. It will be used in experiments presented later in Section 5.

4.3 Non-hierarchical Alphabets

Previous approaches (Sections 4.1 and 4.2) were based on requirement (3) specifying that letters of the seed alphabet should be embedded one into another to form a "nested" hierarchy. This requirement is biologically motivated and, on the other hand, computationally useful as it reduces considerably the space of possible letters. However, this requirement is not necessary to implement the direct indexing (see Introduction). Therefore, we also designed non-hierarchical alphabets in order to compare them to hierarchical ones. We used the following heuristic consisting in generating first a large number of seed candidates, and selecting the ones with (1) high likelihood ratio, (2) a range of different weights.

Resulting Alphabet. An alphabet obtained with the above heuristic, called `Non-tree-transitive`, is shown in Figure 5. This alphabet will be used in the experiments reported in Section 5.

$$
\begin{gathered}
\{ARNDCQEGHILMKFPSTWYV\}\\
\{ARNDQEGHILMKFPSTWYV\}\ \{C\}\\
\{ARNDCQEHILMKFPSTWYV\}\ \{G\}\\
\{ARNDQEHILMKFSTYV\}\ \{CGPW\}\\
\{ARCQEHILMKFSTYV\}\ \{NDGPW\}\\
\{ARNDCQEGHKPST\}\ \{ILMFWYV\}\\
\{ARNDQEGHKST\}\ \{CILMFWYV\}\ \{P\}\\
\{ARNDQEHKPST\}\ \{CW\}\ \{G\}\ \{ILMFYV\}\\
\{ARNDQEKST\}\ \{CP\}\ \{GHW\}\ \{ILMFYV\}\\
\{AGPST\}\ \{RNDQEHK\}\ \{C\}\ \{ILMFWYV\}\\
\{APST\}\ \{RNDQEHK\}\ \{CW\}\ \{G\}\ \{ILMFYV\}\\
\{AGST\}\ \{RNDQEK\}\ \{C\}\ \{HFWY\}\ \{ILMV\}\ \{P\}\\
\{AST\}\ \{RNDQEK\}\ \{CH\}\ \{G\}\ \{ILMV\}\ \{FWY\}\ \{P\}\\
\{AST\}\ \{RQEHK\}\ \{ND\}\ \{CP\}\ \{G\}\ \{ILMV\}\ \{FWY\}\\
\{AST\}\ \{RQK\}\ \{NH\}\ \{DE\}\ \{C\}\ \{G\}\ \{ILMV\}\ \{FWY\}\ \{P\}\\
\{A\}\ \{RQK\}\ \{N\}\ \{DE\}\ \{C\}\ \{G\}\ \{H\}\ \{ILMV\}\ \{FY\}\ \{P\}\ \{ST\}\ \{W\}\\
\{A\}\ \{RK\}\ \{N\}\ \{DE\}\ \{C\}\ \{QH\}\ \{G\}\ \{ILV\}\ \{M\}\ \{FY\}\ \{P\}\ \{ST\}\ \{W\}\\
\{A\}\ \{RQK\}\ \{ND\}\ \{C\}\ \{E\}\ \{G\}\ \{H\}\ \{IV\}\ \{LM\}\ \{FWY\}\ \{P\}\ \{ST\}\\
\{A\}\ \{RK\}\ \{ND\}\ \{C\}\ \{Q\}\ \{E\}\ \{G\}\ \{H\}\ \{IV\}\ \{LM\}\ \{FWY\}\ \{P\}\ \{S\}\ \{T\}\\
\{A\}\ \{RK\}\ \{N\}\ \{D\}\ \{C\}\ \{Q\}\ \{E\}\ \{G\}\ \{H\}\ \{IV\}\ \{L\}\ \{M\}\ \{FY\}\ \{P\}\ \{S\}\ \{T\}\ \{W\}\\
\{A\}\ \{R\}\ \{N\}\ \{D\}\ \{C\}\ \{QE\}\ \{G\}\ \{H\}\ \{I\}\ \{L\}\ \{K\}\ \{M\}\ \{FWY\}\ \{P\}\ \{S\}\ \{T\}\ \{V\}\\
\{A\}\ \{R\}\ \{N\}\ \{D\}\ \{C\}\ \{Q\}\ \{E\}\ \{G\}\ \{H\}\ \{I\}\ \{L\}\ \{K\}\ \{M\}\ \{F\}\ \{P\}\ \{S\}\ \{T\}\ \{W\}\ \{V\}
\end{gathered}
$$

Fig. 5. Non-hierarchical alphabet `Non-tree-transitive`

5 Experiments

This section describes the experiments we made to test the efficiency of seeds we designed with different methods of previous sections. Sections 5.1-5.3 describe the experimental protocol, from the assignment of background and foreground probabilities, to the seed design. In Section 5.4, we analyze the power of different seed models proposed in Sections 3-4 with respect to probabilistic models.

5.1 Probability Assignment and Alphabet Generation

First of all, we derived probabilistic models in accordance with the BLOSUM62 data from the original paper [22]. We obtained the BLOCKS database (version 5) [23] and the software of [22] to infer Bernoulli probabilities for the background and foreground alignment models. These probabilities have been used throughout the whole pipeline of experiments.

Different seed alphabets have then been generated by the methods presented in Section 3 (alphabet `Non-transitive`), Section 4.1 (alphabet `Transitive-predefined`), Section 4.2 (alphabet `Transitive-ab-initio`) and Section 4.3 (alphabet `Non-tree-transitive`).

5.2 Seed Design

To each alphabet, we applied a seed design procedure that we briefly describe now. Since each seed (or seed family) is characterized by two parameters – sensitivity and selectivity – it can be associated with a point on a 2-dimensional plot. Best seeds are then defined to be those which belong to the *Pareto* set among all seeds, i.e. those than cannot be strictly improved by increasing sensitivity, selectivity, or both.

For different selectivity levels, we designed good seed families containing one to six individual seeds, among which the best family was selected. In each seed family, each individual seed has been assumed to have approximately the same weight, within 5% tolerance. This requirement is natural as in the case of divergent weights, seeds with lower weight would dominantly affect the performance. In practice, having individual seeds of similar weight allows an efficient parallel implementation (see e.g. [17]).

Estimation of sensitivity of individual seeds or seed families has been done with the algorithm described in [1] and implemented in the Iedera software, available at `http://bioinfo.lifl.fr/yass/iedera.php`. The selectivity of an individual seed has been computed according to the definition (Section 2). For a seed family, its selectivity has been lower-estimated by summing the background probabilities of individual seeds.

Seed family design has been done using a hill climbing heuristics (see [24,25,11]) alternating seed generation and seed estimation steps. All experiments were conducted for alignment lengths 16 and 32.

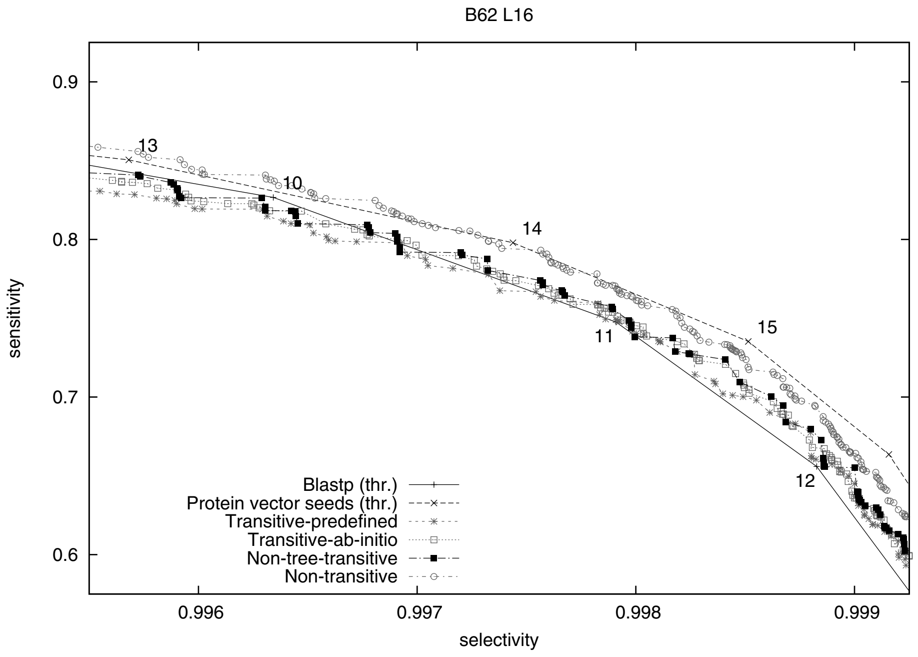
5.3 Blastp and the Vector Seed Family from [4]

Our goal is to compare between different seed design approaches proposed in this paper, but also to benchmark them against other reference seeding methods. We used two references: the Blastp seeding method and the family of vector seeds proposed in [4]. Both of them use a score (or weight) resulting from the accumulative contribution of several neighboring positions to define a hit (see Introduction). Therefore, they use a more powerful (and also more costly to implement) formalism of seeding.

To estimate the sensitivity and selectivity of those seeds, we modified our methods described in the previous section by representing an alignment by a sequence of possible individual scores. Foreground and background probability of each score is easily computed from those for amino acid pairs. After that, sensitivity and selectivity is computed similarly to the previous case.

5.4 Results

We compare the performance of the different approaches by plotting ROC curves of Pareto-optimal sets of seeds on the selectivity/sensitivity graph. The two plots in Figure 6 show the results for alignment length 16 and 32 respectively. The two first polylines show the performance of Blastp with word size 3 and the vector seed family from [4], for different score thresholds. The other curves show the performances of different seed alphabets from Sections 3-4 represented by the Pareto-optimal seeds (seed families) that we were able to construct over those alphabets. As mentioned earlier in Section 5.2, each time we selected the best seed family among those with different number of individual seeds. Typically (but not exclusively), points on the plots correspond to seed families with 3 to 5 seeds. Typically, the seed span ranges between 3 and 5 (respectively, 3 and 6)



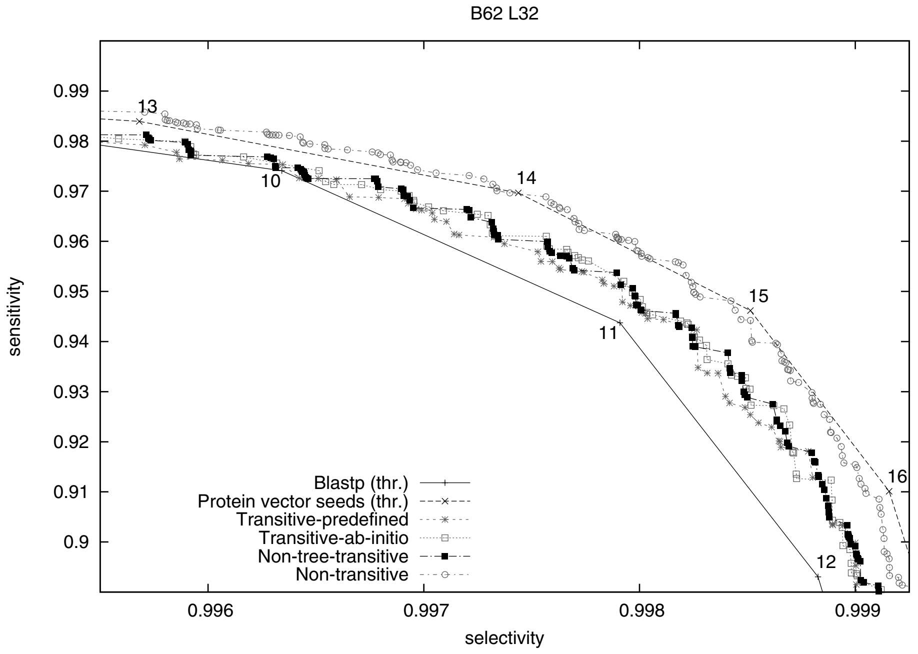


Fig. 6. ROC curves of seed performance measured on the probabilistic model

for alignment length 16 (respectively, 32). Seeds with langer span (> 4) tend to occur in seed families with larger number of seeds (> 3).

We observe that non-transitive seeds over the alphabet of Section 3 are comparable in performance with the vector seed family from [4] and clearly outperforms seeds over other alphabets. This result is interesting in itself, although this alphabet is unpractical in many cases, due to its incompatibility with the transitivity condition.

As for the other alphabets, they roughly show a comparable performance among them. Note that using non-hierarchical alphabet (Section 4.3) does not bring much of improvement, which justifies condition (3). For the alignment length 16, our seeds perform comparably to Blastp, with a slightly better performance for high thresholds and a slightly worse performance for low thresholds. On the other hand, for alignments of length 32, our seeds clearly outperform Blastp.

6 Conclusion

The main conclusion of our work is that although the subset seed model is less expressive than the method of accumulative score used in Blastp, carefully designed subset seeds can reach the same or even a higher performance. To put it informally, the use of the accumulative score in defining a hit can, without loss of performance, be replaced by a careful distinction between different amino acid matches without using any scoring system. From a practical point of view, subset seeds can provide a more efficient implementation, especially for large-scale protein comparisons, due to a much smaller number of accesses to the hash table. In particular, this can be very useful for parallel implementations or specialized hardware (see e.g. [17]).

Note that the seed design heuristic sketched in Section 5.2 does not guarantee to compute optimal seeds, and therefore our seeds could potentially be further improved by a more advanced design procedure, possibly bringing a further increase in performance. This is especially true for seeds of large weight (due to a bigger number of those), for which our seed design procedure could produce non-optimal seeds, thus explaining some “drop-offs” in high-selectivity parts of plots of Figure 5.4.

As far as further research is concerned, the question of efficient seed design remains an open issue. Improvements of the hill climbing heuristics used in this work are likely to be possible.

Acknowledgements. Parts of this work have been done during visits to LIFL of Ewa Szczurek (June-August 2006), Anna Gambin and Sławomir Lasota (August 2006) and Mikhail Roytberg (October-December 2006). Those visits were supported by the ECO-NET and Polonium programs of the French Ministry of Foreign Affairs.

References

1. Kucherov, G., Noé, L., Roytberg, M.: A unifying framework for seed sensitivity and its application to subset seeds. JBCB 4(2), 553–570 (2006)
2. Altschul, S., Gish, W., Miller, W., Myers, E., Lipman, D.: Basic Local Alignment Search Tool. Journal of Molecular Biology 215, 403–410 (1990)

3. Altschul, S., et al.: Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic Acids Res. 25(17), 3389–3402 (1997)
4. Brown, D.: Optimizing multiple seed for protein homology search. IEEE/ACM TCBB 2(1), 29–38 (2004) (earlier version in WABI 2004)
5. Ma, B., Tromp, J., Li, M.: PatternHunter: Faster and more sensitive homology search. Bioinformatics 18(3), 440–445 (2002)
6. Li, M., Ma, B., Kisman, D., Tromp, J.: PatternHunter II: Highly sensitive and fast homology search. JBCB 2(3), 417–439 (2004) (earlier version in GIW 2003)
7. Brejova, B., Brown, D., Vinar, T.: Vector seeds: an extension to spaced seeds. Journal of Computer and System Sciences 70(3), 364–380 (2005)
8. Noé, L., Kucherov, G.: YASS: enhancing the sensitivity of DNA similarity search. Nucleic Acid Res. 33, W540–W543 (2005)
9. Mak, D., Gelfand, Y., Benson, G.: Indel seeds for homology search. Bioinformatics 22(14), e341–e349 (2006)
10. Csürös, M., Ma, B.: Rapid homology search with neighbor seeds. Algorithmica 48(2), 187–202 (2007)
11. Zhou, L., Stanton, J., Florea, L.: Universal seeds for cDNA-to-genome comparison. BMC Bioinformatics 9(36) (2008)
12. Sun, Y., Buhler, J.: Designing multiple simultaneous seeds for DNA similarity search. In: RECOMB, pp. 76–84 (2004)
13. Kucherov, G., Noé, L., Roytberg, M.: Multi-seed lossless filtration. In: Sahinalp, S.C., Muthukrishnan, S.M., Dogrusoz, U. (eds.) CPM 2004. LNCS, vol. 3109, pp. 297–310. Springer, Heidelberg (2004)
14. Yang, I.H., et al.: Efficient methods for generating optimal single and multiple spaced seeds. In: IEEE BIBE, pp. 411–416 (2004)
15. Xu, J., Brown, D., Li, M., Ma, B.: Optimizing multiple spaced seeds for homology search. In: Sahinalp, S.C., Muthukrishnan, S.M., Dogrusoz, U. (eds.) CPM 2004. LNCS, vol. 3109, pp. 47–58. Springer, Heidelberg (2004)
16. Kisman, D., Li, M., Ma, B., Wang, L.: tPatternHunter: gapped, fast and sensitive translated homology search. Bioinformatics 21(4), 542–544 (2005)
17. Peterlongo, P., et al.: Protein similarity search with subset seeds on a dedicated reconfigurable hardware. In: PBC. LNCS, vol. 4967 (2007)
18. Noé, L., Kucherov, G.: Improved hit criteria for DNA local alignment. BMC Bioinformatics 5(149) (2004)
19. Keich, U., Li, M., Ma, B., Tromp, J.: On spaced seeds for similarity search. Discrete Applied Mathematics 138(3), 253–263 (2004) (earlier version in 2002)
20. Li, T., Fan, K., Wang, J., Wang, W.: Reduction of protein sequence complexity by residue grouping. Journal of Protein Engineering 16, 323–330 (2003)
21. Murphy, L., Wallqvist, A., Levy, R.: Simplified amino acid alphabets for protein fold recognition and implications for folding. J. of Prot. Eng. 13, 149–152 (2000)
22. Henikoff, S., Henikoff, J.: Amino acid substitution matrices from protein blocks. Proc. Natl. Acad. Sci. USA 89, 10915–10919 (1992)
23. Henikoff, S., Henikoff, J.: Automated assembly of protein blocks for database searching. Nucleic Acids Res. 19(23), 6565–6572 (1991)
24. Buhler, J., Keich, U., Sun, Y.: Designing seeds for similarity search in genomic DNA. In: RECOMB, pp. 67–75 (2003)
25. Ilie, L., Ilie, S.: Long spaced seeds for finding similarities between biological sequences. In: BIOCOMP, pp. 3–8 (2007)

An Algorithm for Multiple and Global Alignments

Mourad Elloumi and Ahmed Mokaddem

Research Unit of Technologies of Information and Communication,
Higher School of Sciences and Technologies of Tunis
5, Avenue Taha Hussein, Monfleury 1008 Tunis, Tunisia
Mourad.Elloumi@fsegt.rnu.tn

Abstract. In this paper, we develop a new algorithm to construct *Multiple and Global Alignments* (MGA) of *primary structures*, i.e., strings coding biological macromolecules. The construction of such alignments is based on the one of the (longest) *Approximate Common Subsequences* (ACS), made up by longer *approximate* substrings appearing, approximately, in the same positions in all the strings. This ACS represents a MGA. Constructing such alignments is a way to find homologies between biological macromolecules. Our algorithm is of complexity $O(N^2*L^2*(log(L))^2)$ in computing time, where N is the number of the strings and L is the length of the longest string.

Keywords: Strings, multiple and global alignments, common subsequence, divide-and-conquer strategy, algorithms, complexities.

1 Introduction

The comparison of the *primary structures*, i.e., strings coding biological macromolecules consists in identifying similar substrings of these strings. We distinguish two strategies of comparison of primary structures [1]: Either comparison by *alignment* or comparison by *rearrangement* [2], [3], [4]. In this paper we are interested only in the strategy of comparison by *alignment*. There are four basic ways to align primary structures:

(*i*) The first way is to construct a *global alignment*: It consists in aligning the totality of the primary structures to be compared. Among global alignment algorithms, we mention [5], [6], [7], [8], [9], [10], [11], [12].

(*ii*) The second way is to construct a *local alignment:* It consists in aligning portions of the primary structures to be compared, in order to observe local similarities between these structures. Among local alignment algorithms, we mention [13], [14], [15], [16] [17], [18], [19], [20].

(*iii*) The third way is to construct a *pairwise alignment:* It consists in aligning only two primary structures. Among pairwise alignment algorithms, we mention [21], [22], [23], [24], [25], [26], [27], [28]

(*iv*) Finally, the fourth way is to construct a *multiple alignment*: It consists in aligning, more than two primary structures. Among multiple alignment algorithms, we mention [5], [6], [8], [9], [17], [29], [30], [18], [19], [31].

The rest of this paper is organized as follows: In the second section, we give some definitions and notations. In the third section, we describe our algorithm for constructing a MGA to a family of primary structures. In the fourth section, we present the experimental results obtained after running the corresponding program on primary structures of proteins and RNA. Finally, in the last section, we present our conclusion.

2 Definitions and Notations

Let A be a finite alphabet, a *string* is an element of A^*, it is a concatenation of elements of A. The *Levenshtein distance* [32], denoted by $d_{\sigma,\gamma,\delta}$ is the minimum cost of a sequence of *edit operations*, i.e., substitution of cost σ , insertion of cost γ and deletion of cost δ, that transform a string w into another w':

$$d_{\sigma,\gamma,\delta} = min\ \{\sigma^* m_i + \gamma^* n_i + \delta^* l_i\ \}\ . \tag{1}$$

Where m_i, n_i and l_i are, respectively, the numbers of substitutions, insertions and deletions to transform w into w'. Let $f = \{w_1, w_2, \ldots, w_N\}$ be a family of strings, we say that a string x is an *approximate common substring* to the strings of f, if and only if, for each string w_i, $1 \le i \le N$, of f there exists an exact substring x' of w_i, such that $\frac{d_{\sigma,\gamma,\delta}(x,x')}{|x|} \le \varepsilon$, where $\varepsilon > 0$ is an error rate. The exact substring x' is called *image* of x in w_i, $1 \le i \le N$. An *Approximate Common Subsequence* (ACS) to the strings of f is a list of approximate common substrings to the strings of f that appear in the same order, and without overlappings, in all the strings of f. The *length* of an ACS s, denoted by $|s|$, is the sum of the lengths of the approximate common substrings that make up s. An ACS s, $s = [sw_1, sw_2 \ldots sw_n]$, will be denoted by $sw_1 \to sw_2 \to \ldots \to sw_n$. A portion of s beginning at sw_i and ending at sw_j, $0 < i \le j \le n$, is called *sub-ACS* of s and will be denoted by $sw_i \to sw_{i+1} \to \ldots \to sw_j$. Let y_1 and z_1 be two exact substrings of a string w_1, y_2 and z_2 be two exact substrings of a string w_2 and let x be a common approximate substring to w_1 and w_2 such that $y_1 x z_1$ is an approximate substring of w_1 and $y_2 x z_2$ is an approximate substring of w_2, with $|y_1|=|y_2|=|z_1|=|z_2|=l$. The *contextual distance* $CD_{x,l}(w_1,w_2)$, associated with x and l, between w and w' is defined by:

$$CD_{x,l}\,(w1,w2) = d_{\sigma,\gamma,\delta}(y_1,y_2)\ +\ d_{\sigma,\gamma,\delta}(z_1,z_2)\ . \tag{2}$$

Let $f = \{w_1, w_2, \ldots, w_N\}$ be a family of strings and x be a common approximate substring to f. The *norm* $N_{x,l}(f)$, associated with $CD_{x,l}$, between the strings of f is defined by:

$$N_{x,l}\,(f) = \textstyle\sum_{i,j \in [1..N]}\ CD_{x,l}\,(w_i,w_j)\ . \tag{3}$$

Let $x_1, x_2, \ldots, x_N$ be images of x, respectively, in $w_1, w_2, \ldots, w_N$. A list $p_x = [p_1, p_2, \ldots, p_N]_x$ made up by the positions of the first characters of $x_1, x_2, \ldots, x_N$, respectively, in the strings $w_1, w_2, \dots, w_N$, is called *alignment* of x. The *width* of an alignment p_x, denoted by $\delta(p_x)$, is defined by:

$$\delta(p_x) = max_{i,j \in [1..N]} \{p_i\} - min_{i,j \in [1..N]} \{p_i\} \qquad (4)$$

3 Construction of a MGA

Let $f = \{w_1, w_2, \dots, w_N\}$ be a family of strings and $\varepsilon>0$ be an error rate, Our algorithm of construction of a MGA of the strings of f is based on the construction of the (longest) ACS. This ACS represents a MGA. Let $s = sw_1 \rightarrow sw_2 \rightarrow \dots \rightarrow sw_n$ be an ACS to f. The more s is made up by longer approximate common substrings appearing, approximately, in the same positions in all the strings of f, the more s is interesting, i.e., reflects in a better way structural similarities between the strings of f. It is within this scope that we deal with the problem of the ACS to a family of strings. Let $f = \{w_1, w_2, \dots, w_N\}$ be a family of strings and $\varepsilon>0$ be an error rate, to construct an ACS to the family f, our algorithm operates by a *divide-and-conquer* strategy [33]: first, we locate a longest approximate common substring, let us call it sw, appearing, approximately, in the same position in all the strings of f. This approximate common substring partitions each string w_i into three smaller strings: w_{il}, sw_i and w_{ir} such that $w_i=w_{il}sw_iw_{ir}$, where sw_i is *the* image of sw in w_i. This partition gives rise to two new families: $f_1 = \{w_{1l}, w_{2l}, \dots, w_{Nl}\}$ and $f_2 = \{w_{1r}, w_{2r}, \dots, w_{Nr}\}$. Then, we process, recursively, f_1 and f_2 in the same way as f. Let us call s_1 and s_2, respectively, the ACS to f_1 and f_2. The ACS to f is then $s = s_1 \rightarrow sw \rightarrow s_2$. The construction of the longest approximate common substring to the strings of f can be made *via* an adaptation of the algorithm (KMR) [34]. This construction is achieved in two steps: During the first step, we concatenate the strings of f into a single string t, then, at each step, we filter the vector representing the repeated substrings in t, such that, we get exact common substrings with equal lengths. Then, during the second step, we measure the distances d between the different longest exact common substrings, found during the previous step, taken pairwise. Two exact common substrings w and w' such that $\frac{d(w,w')}{|w|} \leq \varepsilon$ represent then the same longest approximate common substring to the strings of f. If we have more than one longest approximate common substring, we make a *filtering*. The filtering is achieved in two steps: During the first step, we a make a *horizontal filtering*: It consists in filtering the different images, in a string, of a same approximate common substring. We operate as follows: with each approximate common substring, we associate a *pivot string*. This string can be the one that contains the maximum number of images of this approximate common substring. At the level of this string, we align each image of this approximate common substring with the (N-1) nearest images appearing in the (N-1) other strings. The alignment kept is the one that has the smallest width. Then, for an approximate common substring, we consider, at the level of each string, only the image taking part of the alignment that has the smallest width. Hence, for each approximate common substring, we will have one image per string. During the second step, we make a *vertical filtering*: It consists in filtering the alignments associated with the different approximate common substrings. We have considered different criteria to filter alignments:

(*i*) *Vertical filtering by norms*: we consider only the alignment with the smallest norm. In the computation of the contextual distances and, hence, in the one of the norms, we set the costs of the edit operations as follows: $\sigma = 2$, $\gamma = \delta = 1$.

(*ii*) *Vertical filtering by widths of alignments*: we consider only the alignment with the smallest width.

(*iii*) And *vertical filtering by frequencies of appearance*: we consider only the alignment for which the associated longest approximate common substring has the maximum number images in *f*.

Time complexity of our algorithm is $O\ (N^2 * L^2 * (log\ (L))^2)$.

4 Experimental Results

The program corresponding to our MGA algorithm is called *μAlign*. It is coded in C++ and implemented on a 850 Mhz Pentium-3 machine, running Windows XP with 320 Mb of RAM. We have run our program *μAlign* on the following families of primary of structures of proteins and families of primary of structures of RNA:

Proteins: *esterase, lipase, lyase, dehydrogenase, asnc family, oxidase family, Phospholase, gntr family, lipocalin family, lysr family, sam domain family, YjgP_YjgQ.*

RNA: *cobalmin, toga family, RNCO family, QUAD family, t_box family, 6S family, intron family, L20 family, U3 family*

We have obtained these data from the biological databases RFAM [35], PFAM [36] and *Entrez* [37]. We have used a rate $\lambda_{(\varepsilon,prog)}$ in order to compare the lengths of the ACS obtained by our program and those of the ACS obtained by other programs. The rate $\lambda_{(\varepsilon,prog)}$ is defined as follows:

$$\lambda_{(\varepsilon,prog)} = \frac{\left|ACS_{\mu Align}\right|}{\left|ACS_{prog}\right|} * 100 \tag{5}$$

where:

- *prog* is the name of the program used
- ε s the error rate.

The histograms below represent the results obtained for families of proteins and families of RNA. We have introduced the notion of *quorum* into our program *μAlign*. The *quorum* is defined as follows: an approximate substring that appears in at least *q* strings of the family of strings is regarded as being common to this family. The following results were obtained with a *quorum* q=75% and thresholds of error rates ε=25%, ε=50% and ε=75%.

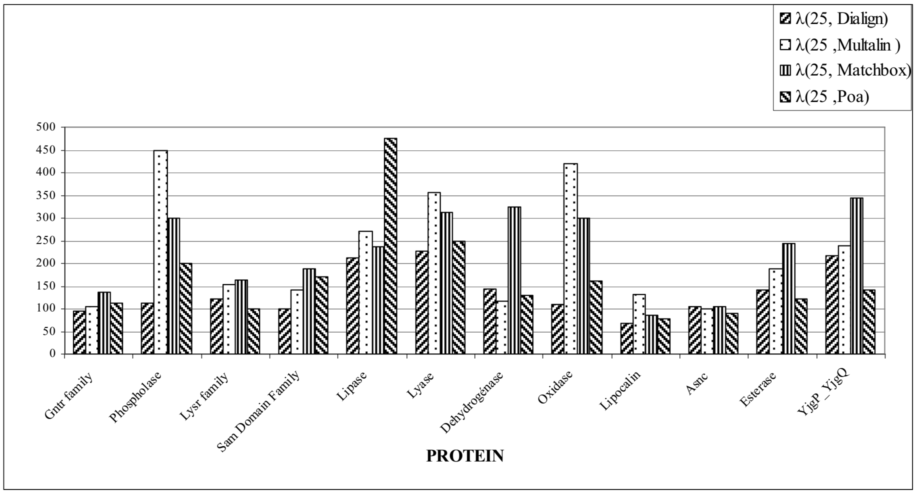


Fig. 1. Results obtained for families of proteins for ε=25%

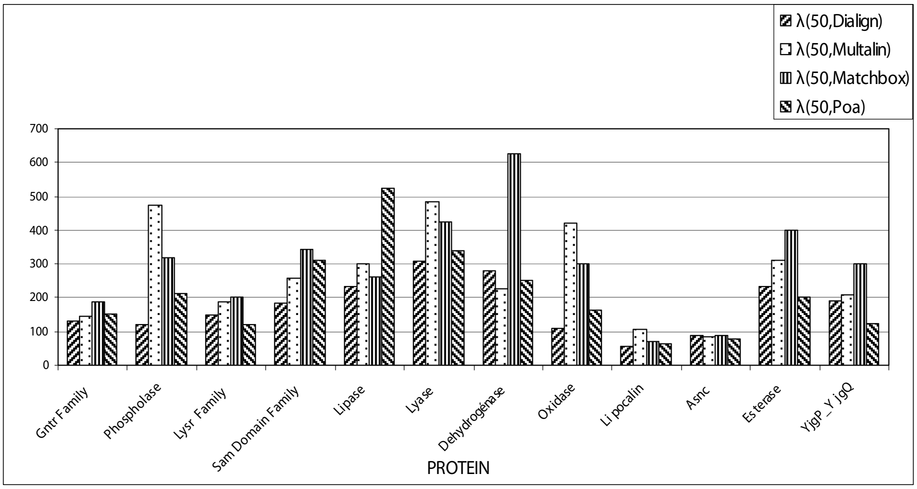


Fig. 2. Results obtained for families of proteins for ε=500025

We notice that the results obtained with families of proteins in the case where ε=50% and the case where ε=75% are almost identical. This is due to the fact that the size of the repeated substrings in the majority of the cases is equal to two characters thus the classes of equivalence and those of pseudo-equivalence formed in both cases are almost the same ones. According to the histograms, we notice that the results obtained with our program *µAlign* for families of proteins and for families of RNA are good, compared to the results obtained with the other programs. But the results obtained with families of proteins are better than those obtained with families of RNA. We can explain this as follows: Our program *µAlign* is based on the identification of repeated substrings.

However, more the size of the alphabet decreases more the probability of having repeated substrings in a string coded starting from this alphabet increases. Since the size of the alphabet associated with the primary structures of RNA is equal to 4 and the one of the alphabet associated with the primary structures of proteins is equal to 20 thus the probability of having repeated substrings in primary structures of RNA is higher than the one of having repeated substrings in primary structures of proteins. On the other hand, more there are occurrences representing the same repeated substrings in a string more our program $\mu Align$ is likely to be induced in error, because of the choice of the

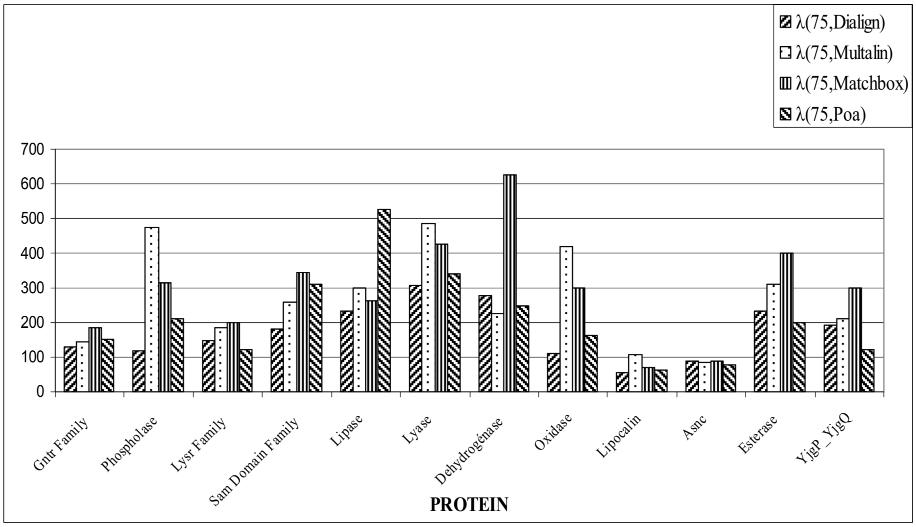


Fig. 3. Results obtained for families of proteins for ε=75%

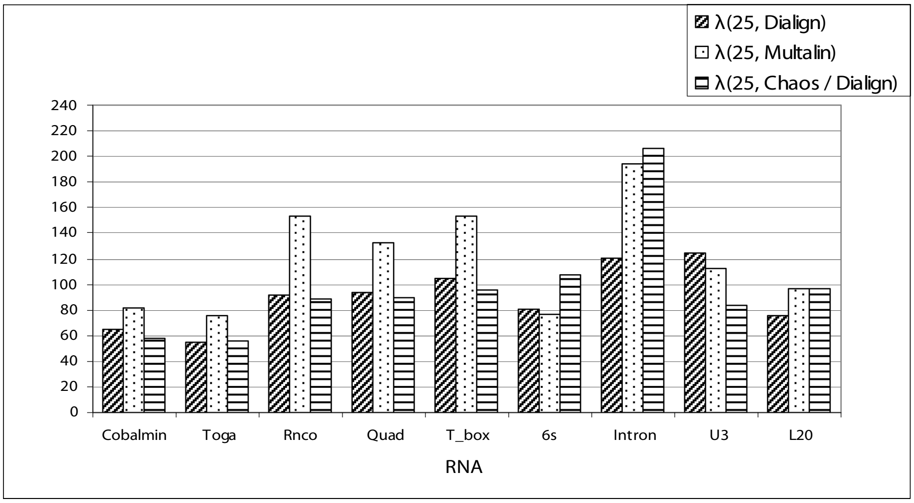


Fig. 4. Results obtained for families of RNA for ε=25%

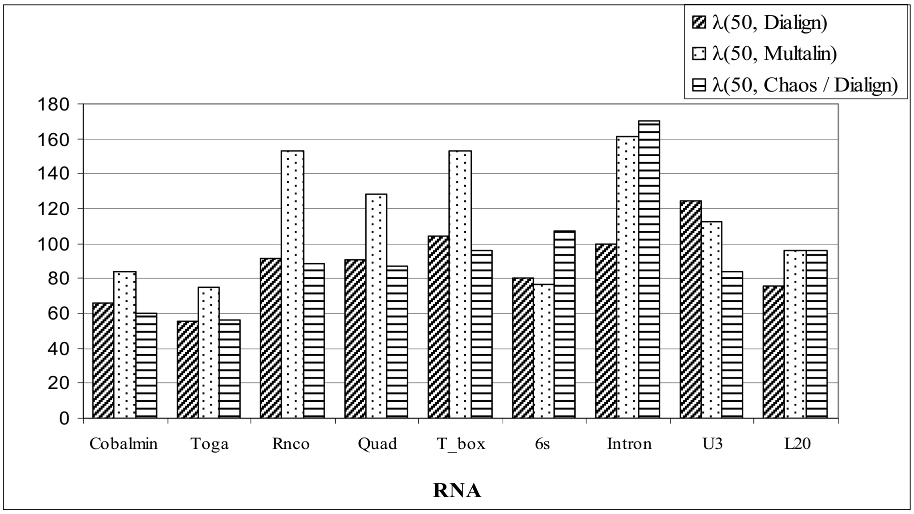


Fig. 5. Results obtained for families of RNA for ε=50%

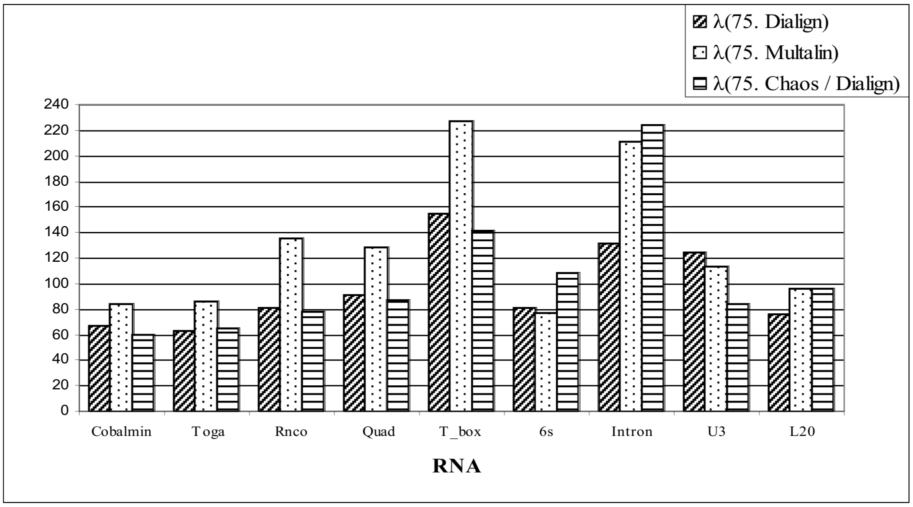


Fig. 6. Results obtained for families of RNA for ε=75%

occurrence associated with the repeated substring. This is why, by using our program *μAlign*, we obtain better results with proteins compared to RNA.

5 Conclusion

In this paper, we have developed a new algorithm to construct *Multiple and Global Alignments* (MGA) of *primary structures*, i.e. The construction of such alignments is based on the one of the (longest) *Approximate Common Subsequences* (ACS), made up by longer approximate substrings appearing, approximately, in the same positions in all the strings. This ACS represents a MGA. Constructing such alignments is a way to find homologies between biological macromolecules. Our algorithm is of complexity $O(N^2{*}L^2{*}(log(L))^2)$ in computing time, where N is the number of the strings and L is the length of the longest string. We have run the program *μAlign*, corresponding to our MGA algorithm, on families of primary of structures of proteins and families of primary of structures of RNA. We have compared the lengths of the ACS obtained by our program *μAlign* with those of the ACS obtained by the programs presented in [5], [6], [8], [9] and [19]. We have noticed that the results obtained with our program *μAlign* for families of proteins and for families of RNA are good, compared to the results obtained with other programs. But the results obtained with families of proteins are better than those obtained with families of RNA. We think that we can improve much more the results obtained with our program *μAlign* by using substitution matrices, like PAM [38] and BLOSUM [39], to measure the similarities between substrings.

References

1. Sagot, M.F.: Ressemblance Lexicale et Structurale Entre Macromolécules -Formalisation et Approches Combinatoires, Thèse de Doctorat, Université de Marne-La-Vallée, France (1996)
2. Hannenhalli, S.: Transforming Men Into Mice a Computationnal Theory of Genome Rearrangements, PhD Thesis, The Pennsylvania State University (1995)
3. Hannenhalli, S., Pevzner, P.: Transforming Cabbage Into Turnip (Polynomial Algorithm for Srting Signed Permutations By Reversals). In: Proc. 27th Annual ACM Symposium on the Theory of Computing, pp. 178–189 (1995)
4. Christie, D.: Genome Rearrangement Problems Ph.D Thesis, University of Glasgow (1998)
5. Corpet, F.: Multiple Sequence Alignment With Hierarchical Clustering. Nucleic Acids Research 16(22), 10881–10890 (1988)
6. Depiereux, E., Feytmans, E.: MATCH-BOX - A fundamentally new Algorithm for The Simultaneous Alignment of Several Protein Sequences. Comput. Appl. Biosci. 8(5), 501–509 (1992)
7. Delcher, A.L., Phillippy, A., Carlton, J., Salzberg, S.L.: Fast Algorithms for Large-ScaleGenome Alignment and Comparison. Nucleic Acids Research 30(11), 2478–2483 (2002)
8. Lee, C., Grasso, C., Sharlow, M.: Multiple Sequence Alignment UingPpartial Order Graphs. Bioinformatics (18), 452–464 (2002)
9. Brudno, M., Chapman, M., Göttgens, B., Batzoglou, S., Morgenstern, B.: Fast and Sensitive Multiple Alignment of Large Genomic Sequences. BMC Bioinformatics 4(66), 1–11 (2003)
10. Bray, N., Pachter, L.: MAVID: Constrained Ancestral Alignment of Multiple Sequences. Genome Research (14), 693–699 (2004)
11. Lassmann, T., Sonnhammer, E.: Kalign: An Accurate and Fast Multiple Sequence Alignment Algorithm. BMC Bioinformatics (6), 298 (2005)
12. Schwartz, A., Pachter, L.: Multiple Alignment by Sequence Annealing. Bioinformatics (2006)

13. Morgenstern, B., Dress, A., Werner, T.: Multiple DNA and Protein Sequence Alignment Based on Segment-to-Segment Comparison. Proc. Natl. Acad. Sci. U.S.A. (93), 2098–12103 (1996)
14. Morgenstern, B., Frech, K., Dress, A., Werner, T.: DIALIGN: Finding Local Similarities by Multiple Sequence Alignment. Bioinformatics 14(3), 290–294 (1998)
15. Lenhof, H.P., Morgenstern, B., Reinert, K.: An Exact Solution for The Segment-to-Segment Multiple Sequence Alignment Problem. Bioinformatics (15), 203–210 (1999)
16. Schwartz, S., Kent, W.J., Smit, A., Zhang, Z.: Human-Mouse Alignments with Blastz. Genome Research, 103–107 (2003)
17. Brudno, M., Do, C.B., Cooper, G.M., Kim, M.F., Davydov, E., Green, E.D., Sidow, A., Batzoglou, S.: LAGAN and Multi-LAGAN: Efficient Tools for Large-Scale Multiple Alignment of Genomic DNA. Genome Research (13), 721–731 (2003)
18. Frith, M.C., Hansen, U., Spouge, J.L.: Finding Functional Sequence Elements by Multiple Local Alignments. Nucleic Acids Research 32(1), 189–200 (2004)
19. Morgenstern, B.: DIALIGN: Multiple DNA and Protein Sequence Alignment at BiBiServ. Nucleic Acids Research 32(Web Server Issue) (2004)
20. Ovcharenko, I., Loots, G.G., Giardine, B.M., Hou, M., Ma, J., Hardison, R.C., Stubbs, L., Millers, W.: Mulan: Multiple-Sequence Local Alignment and Visualization for Studying Function and Evolution. Genome Research (15), 184–194 (2005)
21. Needleman, S.B., Wunsch, C.D.: A General Method Applicable to the Search for Similarities in The Amino-Acid Sequence of two Proteins. Journal of Molecular Biolog (48), 443–453 (1970)
22. Byers, T.H., Waterman, M.S.: Determining All Optimal and Near-Optimal Solutions when Solving Shortest Path Problems by Dynamic Programming. Operations Research 32(6), 1381–1384 (1984) Operations Research Society of America (Eds.)
23. Waterman, M.S., Byers, T.H.: A Dynamic Programming Algorithm to Find All Solutions in a Neighborhood of The Optimum. Mathematical Biosciences (77), 179–188 (1985)
24. Zuker, M.: Suboptimal Sequence Alignment in Molecular biology: 1nalysis with Errors. J. Mol. Biol. 221, 403–420 (1991)
25. Naor, D., Brutlag, D.L.: On Near-Optimal Alignments of Biological Sequences. J. Comp. Biol. (4), 349–366 (1994)
26. Kurtz, S., Ohlebusch, E., Schleiermacher, C., Stoye, J.: Reputer: The Manifold Applications of Repeat Analysis. Nucleic Acids Research 29(22), 4633–4642 (2001)
27. Noé, L.: Recherche de Similarités dans Les Séquences d'ADN: Modèles et Algorithmes pour la Conception de Graines Efficaces, Thése de Doctorat, Université Henri Poincaré (2005)
28. Huang, W., Umbach, D.M., Leping, L.: Accurate Anchoring Alignment of Divergent Sequences. Bioinformatics, 22(1), 29–34 (2006)
29. Zhang, X., Kahveci, T.: A New Approach for Alignment of Multiple Proteins. In: Pacific Symposium on Biocomputing, vol. 11, pp. 339–350 (2006)
30. Edgar, R.C.: MUSCLE: Multiple Sequence Alignment with High Accuracy and High throughput. Nucleic Acids Research 32(5), 1792–1797 (2004)
31. Liang Ye, Y., Huang, X.: MAP2: Multiple Alignments of Syntenic Genomic sequences. Nucleic Acids Research 33(1), 162–170 (2005)
32. Levenshtein, V.I.: Binary Codes Capable of Correcting Deletions, Insertions and Reversals. Cybernetics and Control Theory 10(8), 707–710 (1966)
33. Aho, A.V., Hopcroft, J.E., Ullman, J.D.: The Design and Analysis of Computer Algorithms, pp. 60–65. Addison-Wesley Publishing Company, Reading (1974)

34. Karp, R., Miller, R.E., Rosenberg, A.L.: Rapid Identification of Repeated Patterns in Strings, Trees and Arrays. In: 4th symposium of theory of Computing, pp. 125–136 (1972)
35. RNA Families Database of Alignments and CMs http://www.sanger.ac.uk/Software/Rfam
36. Protein Families Database, http://www.sanger.ac.uk/Software/Pfam
37. National Center for Biotechnology Information, http://www.ncbi.nlm.nih.gov
38. Dayhoff, M.O., Schwartz, R.M., Orcutt, B.C.: A Model for Evolutionary Change. Atlas of Protein Sequence and Structure 5(3), 345–352 (1979)
39. Henikoff, S., Henikoff, J.G.: Amino Acid Substitution Matrices From Protein Blocks. Proc. Natl. Acad. Sci. U.S.A. 89, 10915–10919 (1992)

LASA: A Tool for Non-heuristic Alignment of Multiple Sequences

Ernst Althaus and Stefan Canzar

Max-Planck Institut für Informatik,
Campus E1 4, D-66123 Saarbrücken, Germany

Abstract. We have developed a non-heuristic tool (LASA) for the multiple sequence alignment problem (MSA), one of the most important problems in computational molecular biology. It is based on a dynamic programming algorithm for solving a Lagrangian relaxation of an integer linear programming (ILP) formulation for MSA. The objective function that is optimized by LASA models the sum-of-pairs scoring scheme and "truly" affine gap costs. Due to a reformulation w.r.t. additionally introduced variables prior to relaxation we improve the convergence rate dramatically while at the same time being able to solve the Lagrangian problem efficiently. Our experiments show that our implementation LASA outperforms all exact algorithms for the multiple sequence alignment problem. Furthermore, the quality of the alignments ranks among the best computed so far.

1 Introduction

The importance of *multiple string comparison* (of DNA or protein sequences) in computational molecular biology is evidenced by the large number of programs that have been developed for the multiple *alignment* problem, one common formalization (see next section) of the multiple string comparison.

Multiple alignment programs may detect biologically important, yet faint, similarities from a set of strings, that might not be apparent when comparing two strings alone. From these commonalities one might be able to infer evolutionary trees or group proteins into structurally or functionally related families.

On the other hand, multiple alignment can be viewed as solving problems that are inverse to the ones addressed by pairwise string comparisons [9]. Pairwise alignments are mostly used to find strings in a database that share certain patterns with a query sequence but which might not be known to be biologically related. The inverse problem is to deduce common patterns from known biological relationships.

Finding an alignment of k sequences that is optimal in the *sum-of-pairs (SP)* scoring scheme (defined in the next section) becomes quickly computationally intractable as k increases. For example, dynamic programming algorithms find an optimal alignment of sequences of length n with (quasi)-affine gap costs in time and space $\mathcal{O}\left(n^k\right)$ [8]. More complex gap cost functions add a polylog factor to this complexity [7]. If the number k of sequences is not fixed, it has been proved by Elias [6] that multiple alignment with SP score is $\mathcal{NP}$-complete by a reduction from Independent Set in 3-regular graphs . Hence it is unlikely that polynomial time algorithms exist and, depending on the problem size, various heuristics are applied to solve the problem approximately (see, e.g., [17,16]).

The remainder of this paper is organized as follows. It first introduces preliminary definitions and reviews the ILP formulation of the multiple sequence alignment problem in Section 2. Section 3 gives a broad overview of our approach and assesses the practical relevance of our improvements in asymptotic running time. Section 4 describes the approximation of the Lagrangian dual problem. Finally, computational experiments on a set of real-world instances are reported in Section 5, Section 6 concludes the paper.

Preliminaries. Let us formally state the multiple alignment problem. Let $\mathcal{S} = \{s^1, s^2, \ldots, s^k\}$ be a set of k strings over an alphabet Σ and let $\bar{\Sigma} = \Sigma \cup \{-\}$. Given a string s, we let $\|s\|$ denote the number of characters in the string and s_l the lth character of s, for $l = 1, \ldots, \|s\|$. We will assume that $\|s^i\| \geq 4$ for all strings s^i and let $n := \sum_{i=1}^{k} \|s^i\|$.

A (global) *multiple alignment* of $\mathcal{S}$ is a set $\mathcal{A} = \{\bar{s}^1, \bar{s}^2, \cdots, \bar{s}^k\}$ of strings over the alphabet $\bar{\Sigma}$ where each string can be interpreted as a row of a two dimensional *alignment matrix*. $\mathcal{A}$ has to satisfy the following properties: (1) the strings in $\mathcal{A}$ all have the same length, (2) ignoring dashes ("–"), string $\bar{s}^i$ is identical to string s^i, and (3) there is no column of the alignment matrix consisting entirely of dashes (see figure 1(a)).

For a given pairwise scoring function w on letters from alphabet $\bar{\Sigma}$, the *sum of pairs (SP)* score for a multiple alignment $\mathcal{A}$ is the sum of the scores of all pairwise projections [9], namely $c(\mathcal{A}) = \sum_{h=1}^{l} \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} w(\bar{s}^i_h, \bar{s}^j_h)$, where l denotes the (equal) length of strings in $\mathcal{A}$ and $w(-,-) := 0$.

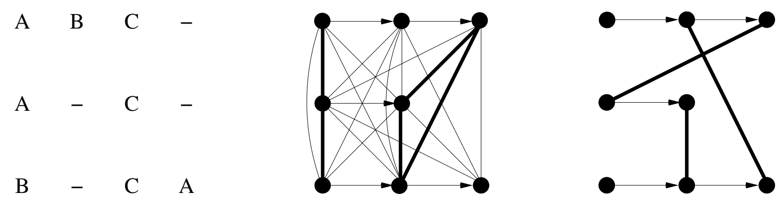
Depending on the definition of the scoring function w, the "score" of an alignment can be naturally interpreted as distances between or similarities of sequences. In this paper we will formally cast the multiple alignment problem as a maximization ("of weights") problem.

To create biologically meaningful alignments, our objective function includes a term that reflects the notion of a *gap*. For a given alignment $\mathcal{A}$, a gap in $\bar{s}^i$ with respect to $\bar{s}^j$ is a maximal, consecutive run of dashes in $\bar{s}^i$ in the projection of $\mathcal{A}$ to strings $\bar{s}^i$ and $\bar{s}^j$. Associated with each of these gaps is a cost. In the *affine gap cost* model the cost of a single gap of length q is given by the affine function $c_{\text{open}} + qc_{\text{ext}}$, i.e. such a gap contributes a weight of $-c_{\text{open}} - qc_{\text{ext}} = w_{\text{open}} + qw_{\text{ext}}$ to the total weight of the alignment, while $w(x,-) = w(-,x) := 0$ for any $x \in \Sigma$. The problem calls for an alignment $\mathcal{A}$ whose overall weight is maximized.

2 Previous Work

In [2] Althaus et al. use a formulation for the multiple sequence alignment problem as an ILP given by Reinert in [18].

For ease of notation, they define the *gapped alignment graph*, a mixed graph whose node set corresponds to the characters of the strings and whose edge set



(a) Alignment of Input Sequences (b) Gapped Alignment Graph (c) Ordering Conflict

Fig. 1. (a) A possible alignment $\mathcal{M}$ of the input sequences $\mathcal{S} = \{ABC, AC, BCA\}$. (b) The gapped alignment graph for the sequences in $\mathcal{S}$. The thick edges specify alignment $\mathcal{M}$. (c) The alignment edges can not be realized at the same time in an alignment. Together with appropriate arcs of A_P, they form a mixed cycle.

is partitioned into undirected alignment edges and directed positioning arcs as follows: $G = (V, E_A \cup A_P)$ with $V = V^i \cup \cdots \cup V^k$ and $V^i = \{u^i_j \mid 1 \leq j \leq \|s^i\|\}$, $E_A = \{uv \mid u \in V^i, v \in V^j, i \neq j\}$ and $A_P = \{(u^i_l, u^i_{l+1}) \mid 1 \leq i \leq k \text{ and } 1 \leq l < \|s^i\|\}$ (see Figure 1(b)). Furthermore, we denote with $\mathcal{G} = \{(u, v, j) \mid u, v \in V^i, j \neq i\}$ the set of all possible gaps.

The ILP formulation uses a variable for every possible alignment edge $e \in E_A$, denoted by x_e, and one variable for every possible gap $g \in \mathcal{G}$, denoted by y_g. Reinert [18] showed that solutions to the alignment problem are exactly the $\{0, 1\}$-assignments to the variables such that

(PaiwAl) we have pairwise alignments between every pair of strings,

(MixedCy) there are no *mixed cycles*, i.e. in the subgraph of the gapped alignment graph consisting of the positioning arcs A_P and the realized edges $\{e \in E_A \mid x_e = 1\}$ there is no cycle that respects the direction of the arcs of A_p (and uses the edges of E_A in either direction) and contains at least one arc of A_P (see Figure 1(c)),

(Trans) transitivity is preserved, i.e. if u is aligned with v and v with w then u is aligned with w, for $u, v, w \in V$.

These three conditions are easily formulated as linear constraints (see [2]).

We refrain from explicitly specifying the inequalities enforcing (PaiwAl) and (Trans), as they are not crucial for the understanding of our approach.

Similarly as in the cutting plane approach in [2], we observed in [1] that the number of iterations of our subgradient optimization (see section 4) can be considerably reduced, if we use additional variables $z_{(u,v)}$ for $u \in V^i, v \in V^j, i \neq j$, with the property that $z_{(u,v)} = 1$ iff at least one character of the string of u lying (not strictly) right of u is aligned to a character of the string of v lying (not strictly) left of v, i.e. $z_{(u^i_l, u^j_m)} = 1$, iff there is $l' \geq l$ and $m' \leq m$ with $x_{u^i_{l'}, u^j_{m'}} = 1$. This condition is captured by the inequalities

$$0 \le z \le 1, \quad \begin{aligned} & z_{(u^i_{\|s^i\|}, u^j_1)} = x_{u^i_{\|s^i\|} u^j_1}, \\ & z_{(u^i_l, u^j_m)} \ge z_{(u^i_{l+1}, u^j_m)} + x_{u^i_l u^j_m} \quad \text{and} \\ & z_{(u^i_l, u^j_m)} \ge z_{(u^i_l, u^j_{m-1})} + x_{u^i_l u^j_m}. \end{aligned} \tag{1}$$

Notice that indicator variables x_e are associated with undirected edges $e = uv$, whereas variables z_e are associated with directed edges $e = (u, v)$.

Using these additional variables, we can define facets that guarantee (MixedCy) as follows. We model the mixed cycles as introduced above by letting $A_A = \{(u, v) \mid u \in V^i, v \in V^j, i \ne j\}$, i.e. for each undirected edge $uv \in E_A$, we have the two directed arcs (u, v) and (v, u) in A_A. Then a cycle $M \subseteq A_A \cup A_P$ in $(V, A_A \cup A_P)$ that contains at least one arc of A_P uniquely defines a mixed cycle. Where it is clear from the context, we therefore resign to distinguish between the term *mixed cycle* in its original meaning, namely cycles as defined in (MixedCy) having both undirected and directed edges, and their corresponding cycles in $(V, A_A \cup A_P)$, consisting exclusively of directed arcs. The set of all mixed cycles is denoted by $\mathcal{M}$.

In [2] the authors show, that for a mixed cycle $M \in \mathcal{M}$ the inequality

$$\sum_{e \in M \cap A_A} z_e \le |M \cap A_A| - 1 \tag{2}$$

is valid and that we can restrict the attention to mixed cycles $\mathcal{M}$ containing exactly one positioning arc.

3 The Extended Pairwise Alignment Problem

Our Lagrangian approach is based on the integer linear program outlined above. In order to score the alignment, we assign each edge $u^i_l u^j_m \in E_A$ a weight $w_{u^i_l u^j_m} := w(s^i_l, s^j_m)$ and a gap (u^i_l, u^i_m, j) the weight $w_{(u^i_l, u^i_m, j)} := w_{\text{open}} + (m - l + 1) \cdot w_{\text{ext}}$, which represents the benefit of realizing that edge or gap.

Since a single variable x_{uv}, $y_{(u,v,j)}$, or $z_{(u,v)}$ involves exactly two sequences, we can partition the three classes of variables, X, Y, and Z, into sets $X^{i,j}$, $Y^{i,j}$, and $Z^{i,j}$, of variables involving sequences i and j.

If we restrict our attention to the variables in $X^{i,j}$, $Y^{i,j}$ and $Z^{i,j}$, for a specific pair of sequences i, j, a solution of the ILP yields a description of a pairwise alignment between sequences i and j, along with appropriate values for the variables in $Z^{i,j}$. The constraints *(MixedCy)* and *(Trans)* are used to guarantee that all pairwise alignments together form a multiple sequence alignment. We call an assignment of $\{0, 1\}$-values to variables in $(X^{i,j}, Y^{i,j}, Z^{i,j})$ such that $(X^{i,j}, Y^{i,j})$ imposes a pairwise alignment and $Z^{i,j}$ satisfies inequalities (1), an *extended pairwise alignment.* Given weights for the variables in $X^{i,j}$, $Y^{i,j}$ and $Z^{i,j}$, we call the problem of finding an extended pairwise alignment of maximum weight the *extended pairwise alignment problem.*

In our Lagrangian approach we dualize the constraints for condition *(MixedCy)* (i.e. inequalities (2)) and relax conditions *(Trans)* (during experiments it turned out that relaxing condition *(Trans)* is more efficient in practice as dualizing them). Hence our Lagrangian subproblem is an extended pairwise alignment problem. More precisely, if $\lambda_M \geq 0$ is the current multiplier for the mixed cycle inequality of $M \in \mathcal{M}$, we have to solve the Lagrangian relaxation problem

$$\begin{aligned}\sum_{M\in\mathcal{M}} \lambda_M(|M \cap A_A| - 1) \quad &+ \\ &\max \sum_{e\in E_A} w_e x_e + \sum_{g\in\mathcal{G}} w_g y_g - \sum_{M\in\mathcal{M}} \lambda_M \sum_{e\in M\cap A_A} z_e \qquad (LR_\lambda) \\ &\text{s.t.}(X^{i,j}, Y^{i,j}, Z^{i,j}) \text{ forms an extended pairwise alignment for all } i, j.\end{aligned}$$

For a given pair of sequences s^i and s^j let K denote the number of variables in $Z^{i,j}$ that have a non-zero coefficient assigned in the objective function, i.e. $K = |\{z_{(u,v)} \in Z^{i,j} \mid \sum_{M\in\mathcal{M}|(u,v)\in M} \lambda_M \neq 0\}|$ (see LR_λ). In [1] the authors introduced a dynamic programming algorithm that solves the extended pairwise alignment problem between sequences s^i and s^j of lengths n_i and n_j, respectively, in time $\mathcal{O}\left(n_i^2 n_j^2 K\right)$, to which we refer to in the following as "simple algorithm". In a second step the number of arcs in the dynamic programming graph was reduced such that our "improved algorithm" achieves a running time of $\mathcal{O}\left(n_i n_j + K^4\right)$. Finally, the dynamic programming graph was augmented with a so called *bypass graph* $G = (\mathcal{V}, \mathcal{E})$, to achieve a running time of $\mathcal{O}\left(n_i n_j + |\mathcal{V}| + |\mathcal{E}|\right)$, which can be bounded by $\mathcal{O}\left(n_i n_j + K^3\right)$ (by bounding $|\mathcal{V}|$ and $|\mathcal{E}|$ by $\mathcal{O}\left(K^2\right)$, respectively $\mathcal{O}\left(K^3\right)$).

Due to the minor improvement w.r.t asymptotic running time ($K \in \mathcal{O}\left(n_i n_j\right)$), we tried to compare the practical performance of the simple algorithm and both versions of the improved algorithm (with and without bypass graph) by considering the size of the underlying graph structure after the last iteration in the root node of the branch and bound tree. Table 3 indicates that for a bpg $G = (\mathcal{V}, \mathcal{E})$, $\mathcal{O}\left(K^2\right)$ and $\mathcal{O}\left(K^3\right)$ are rather pessimistic estimates for $|\mathcal{V}|$, respectively $|\mathcal{E}|$, and therefore we expect the running time of the simple algorithm to be significantly larger than the running time of the improved algorithm using the bpg. Moreover, the "transitive reduction" obtained by introducing the bypass graph reduces the number of additional arcs considerably.

We could solve the extended pairwise alignment problem at least twice as fast when using an A^*-approach: Roughly speaking, the dynamic programming scores computed during an iteration of the subgradient optimization (see Section 4) can be at most the scores of the first iteration, i.e. when all multipliers λ are set to 0.

4 Improving the Lagrangian Relaxation Bound

Recall that (LR_λ) is the problem of computing all extended pairwise alignments for a given set of multipliers λ and $v(LR_\lambda)$ is its objective function value. Moreover, (P) is the multiple sequence alignment problem itself.

Table 1. For the first four benchmark alignments of each subgroup of short and medium sized instances of the BAliBASE library [21], we give the size of the bpg. To the names of instances an indication (k, n) of the number of sequences and the overall number of characters is added. The last three columns give the running times of tools LASA, COSA and MSA. The CPU time was limited to 12 hours ("–").

Instance	K	#BPG-Nodes	#BPG-Arcs	#Arcs	LASA	COSA	MSA
Reference 1 Short, V3							
1aho (5/320)	32	7	42	205	<1	1:29	-
1csp (5/339)	6	0	0	0	<1	1	<1
1dox (4/374)	18	3	18	56	3	30	<1
1fkj (5/517)	29	6	35	137	13	6:04	-
Reference 1 Short, V2							
1aab (4/291)	18	3	18	51	<1	4	< 1
1csy (5/510)	57	11	58	333	17	3:01	-
1fjlA (6/398)	13	2	13	33	12	34	-
1hfh (5/606)	61	17	91	760	33	-	-
Reference 1 Short, V1							
1aboA (5/297)	63	40	236	4047	9:13:49	-	-
1tvxA (4/242)	82	61	373	8979	1:59:44	-	-
1idy (5/269)	51	21	120	1314	10:27:30	-	-
1r69 (4/277)	74	29	165	1934	58:40	-	-
Reference 1 Medium, V3							
1amk (5/1241)	17	1	7	9	8	-	-
1ar5A (4/794)	39	10	56	258	20	-	-
1ezm (5/1515)	24	5	26	83	23	-	-
1led (4/947)	65	13	72	468	3:54	-	-
Reference 1 Medium, V2							
1ad2 (4/828)	51	11	61	360	42	-	-
1aym3 (4/932)	54	21	122	1286	2:37	-	-
1gdoA (4/988)	104	22	121	1168	2:38:36	-	-
1ldg (4/1240)	67	14	76	491	8:32	-	-

Since the optimal value $v(LR_\lambda)$ is an upper bound on the optimal value of (P) for all multiplier vectors $\lambda \in \mathbb{R}^m_+$, $m = |\mathcal{M}|$, we are interested in solving the problem

$$\min_{\lambda \geq 0} v(LR_\lambda) \qquad (LR)$$

to obtain tighter bounds for our branch-and-bound algorithm.

Subgradient Optimization. It is well known that the *Lagrangian function* $f(\lambda) = v(LR_\lambda)$ (for our case where (P) is a maximization problem) is a convex function of λ, but it is not differentiable at points, where the optimal solution of (LR_λ) is not unique. A commonly used approach to determine near-optimal Lagrangian multipliers efficiently is based on the vector of *subgradients*

$g(\lambda) \in \mathbb{R}^m$, associated with a given λ. The set $\partial f(\lambda^0)$ of all subgradients of $f(\lambda)$ at a point λ^0 is always nonempty, and one can show that the vector

$$g_M(\lambda^0) = r - 1 - \sum_{j=1}^{r} \bar{z}_{(u_j, u_{j+1})}, \quad M \in \mathcal{M} \tag{3}$$

is contained in $\partial f(\lambda^0)$, where $\bar{z}$ is an optimal solution to (LR_{λ^0}). The iterative approach proposed by Held and Karp [10] generates a sequence $\lambda^0, \lambda^1, \ldots$ of Lagrangian multipliers by taking at iteration k a step along a subgradient of $f(\lambda^k)$, projecting the resulting point onto the nonnegative orthant:

$$\lambda_M^{k+1} = \max\left\{0, \lambda_M^k + \theta \frac{v(LR_{\lambda^k}) - LB}{\sum\limits_{M' \in \mathcal{M}} {g_{M'}}^2} g_M(\lambda^k)\right\}, \quad M \in \mathcal{M} \tag{4}$$

where LB is a lower bound on $v(P)$, and θ is a step size parameter assuming values in $(0, 2]$. As to the adaption of scalar step size θ, our approach differs from the classical Held-Karp method, which halves parameter θ when there is no upper bound improvement for a certain number of consecutive iterations. If the best and worst upper bounds computed in the last p iterations differ by more than 1%, we suspect that we are "overshooting" and thus we halve the current value of θ. If, in contrast, the two values are within 0.1% from each other, we overestimate $v(LR_{\lambda^*})$, where λ^* is an optimal solution to (LR), and therefore increase θ by a factor of 1.5. Similarly to [4], we experienced a faster convergence to near optimal multipliers using this strategy, compared to the classical approach.

As (2) involves exponentially many mixed cycle inequalities that would have to be dualized, formula (4) can not be applied in a straightforward way, but we use the relax-and-cut framework outlined below.

Relax-and-Cut. In the traditional case of the subgradient method (SM), when the number of dualized constraints is not too large, Beasley [3] reported good practical convergence to $v(LR)$, when setting $g_i = 0$ whenever $g_i \geq 0$ and $\lambda_i = 0$, for $i \in 1, \ldots, m$, i.e. if an inequality whose multiplier is 0 is not violated. We extend this idea by setting $g_M = 0$ for all M with $\lambda_M = 0$ whose corresponding mixed cycle inequalities are not violated by the Lagrangian solution. These multipliers would remain zero valued at the end of the current iteration and thus would not directly contribute to $v(LR_\lambda)$, at any given SM iteration. We call the corresponding constraints *inactive inequalities*. Conversely, we call inequalities, whose associated multiplier may directly contribute to the Lagrangian objective function, *active inequalities*. These are the constraints (2) that are violated by the Lagrangian solution and those inequalities that have nonzero multipliers associated with them. Otherwise the value $\sum_{M \in \mathcal{M}} g_M$ would be very high, resulting in virtually unchanged multipliers from iteration to iteration. We therefore apply (4) exclusively to active inequalities, as suggested in [14].

This dynamic scheme, where the pool of active inequalities may continuously change, heavily relies on the ability to identify inequalities that are violated by the Lagrangian solution. In order to prevent the set of active inequalities from growing too rapidly we restrict the separation problem to mixed cycle inequalities, that are most violated by the average of the last h solutions. Experiments show, that this modification improves the rate of convergence dramatically.

5 Experiments

We have implemented our Lagrangian approach in C++ using the LEDA-library [15] and have embedded it into a branch-and-bound framework. The lower bounds in each bb node are computed by selecting, in a greedy fashion, edges from the set $\{e \in E_A \mid x_e = 1\}$ that satisfy conditions *(PaiwAl)*, *(MixedCy)*, and *(Trans)*. We set $w_{ext} = 4$ and $w_{open} = 6$, i.e. the gap arcs were assigned a weight that was computed as $4l + 6$, where l is the number of characters in the corresponding gap. The weights for the alignment edges in E_A were obtained by the BLOSUM62 amino acid substitution matrix.

While the purpose of the experiments in [1] was mainly to evaluate the complexity of instances our approach was able to solve in reasonable time, in this work we want to assess the quality of alignments our current implementation, which we will call LASA (LAgrangian Sequence Alignment), produces. Additionally to the set of instances of the BAliBASE library [21] we used reference alignments from three different sets: SABmark [22], PREFAB [5] and artificially created alignments using Rose [19]. We used the original benchmarking measures proposed by its respective database. A score between 0 and 1 indicates the degree of accordance with the reference alignment.

Table 3 compares the performance of our implementation with the exact methods MSA [13] and COSA [2]. Although MSA reduces the complexity of the prob-

Table 2. Average score of the alignments computed by different programs. Only instances that have been solved by LASA in less than 2 hours were considered. In BAliBASE 3.0 full length sequences (full) and instances from the homologous region set (hom) are distinguished.

Group	LASA	T-COFFEE	CLUSTALW	MAFFT	MUSCLE
			BAliBASE 2.0		
Short V1	0.969	0.968	0.984	0.988	0.970
Short V2	0.865	0.819	0.936	0.948	0.811
Short V3	0.512	0.340	0.562	0.882	0.537
Medium V1	0.944	.952	0.943	0.969	0.969
Medium V2	0.933	0.886	0.911	0.901	0.895
Long V1	0.960	0.941	0.933	0.976	0.982
			BAliBASE 3.0		
RV11 full	0.942	0.966	0.935	0.939	0.952
RV11 hom	0.795	0.672	0.764	0.819	0.780
RV12 full	0.918	0.900	0.918	0.919	0.905
RV12 hom	0.894	0.876	0.895	0.882	0.888

Table 3. Rows show the average developer (f_D) score and modeler (f_M) score for the "Superfamily"($\leq$ 50% identity) and "Twilight Zone" ($\leq$ 25% identity) sets in the SABmark database, the quality (Q) score and total column (TC) score for PREFAB instances and reference alignments created by Rose, achieved by each aligner. The latter are grouped according to their average evolutionary distance. The number of sequences in each set is given in parenthesis.

Group/Score	LASA	T-COFFEE	CLUSTALW	MAFFT	MUSCLE	DIALIGN	POA
SABmark (405)							
Superfam. f_D	79.56	80.50	81.04	82.32	80.80	75.50	69.41
Superfam. f_M	61.11	62.50	62.39	62.88	62.28	60.57	63.69
Twilight f_D	47.82	46.84	50.3	48.72	47.91	40.64	29.74
Twilight f_M	33.43	33.13	34.25	34.26	33.43	31.23	34.66
PREFAB (161)							
Q	0.71	0.69	0.69	0.69	0.71	0.63	0.57
TC	0.71	0.69	0.69	0.69	0.71	0.63	0.57
Rose (264)							
Dist100 Q	0.91	0.90	0.88	0.91	0.92	0.87	0.79
Dist100 TC	0.86	0.86	0.82	0.88	0.88	0.81	0.67
Dist150 Q	0.78	0.76	0.80	0.81	0.83	0.72	0.55
Dist150 TC	0.70	0.67	0.72	0.71	0.75	0.59	0.38
Dist200 Q	0.67	0.60	0.62	0.63	0.67	0.51	0.34
Dist200 TC	0.50	0.44	0.46	0.50	0.54	0.32	0.18
Dist250 Q	0.58	0.46	0.55	0.49	0.53	0.39	0.29
Dist250 TC	0.37	0.23	0.36	0.28	0.35	0.18	0.11

lem by incorporating quasi-affine gap costs into the multiple alignment, it could hardly solve instances with a moderate degree of similarity. In contrast, LASA outperforms the CPLEX based approach COSA.

In terms of alignment quality, we compared LASA with the heuristic methods T-COFFEE [17], CLUSTALW [20], MAFFT [11], MUSCLE [5], DIALIGN [16] and POA [12]. Our approach ranks among the best programs implemented so far (see Table 2 and Table 3). The quality could be probably improved by a more careful choice of the objective function. In our current implementation we use a fixed objective for all instances, no matter what their level of identity is.

6 Conclusion

We have presented the fastest algorithm to exactly compute multiple sequence alignments with affine gap costs and showed that the quality ranks among the best alignments computed so far. Nevertheless, we believe that a more careful choice of the underlying biological model can improve the quality of the alignments considerably. Our implementation LASA will be available as part of the software library SEQAN currently developed by the free university of Berlin.

References

1. Althaus, E., Canzar, S.: A lagrangian relaxation approach for the multiple sequence alignment problem. In: Dress, A.W.M., Xu, Y., Zhu, B. (eds.) COCOA. LNCS, vol. 4616. Springer, Heidelberg (2007)
2. Althaus, E., Caprara, A., Lenhof, H.-P., Reinert, K.: Aligning multiple sequences by cutting planes. Mathematical Programming 105, 387–425 (2006)
3. Beasley, J.: Lagrangian Relaxation. In: Modern heuristic techniques for combinatorial problems. Blackwell Scientific Publications (1993)
4. Caprara, A., Fischetti, M., Toth, P.: A heuristic method for the set cover problem. Operations Research 47, 730–743 (1999)
5. Edgar, R.C.: Muscle: multiple sequence alignment with high accuracy and high throughput. Nucleic Acids Research 32(5), 1792–1797 (2004)
6. Elias, I.: Settling the intractability of multiple alignment. In: Ibaraki, T., Katoh, N., Ono, H. (eds.) ISAAC 2003. LNCS, vol. 2906, pp. 352–363. Springer, Heidelberg (2003)
7. Eppstein, D.: Sequence comparison with mixed convex and concave costs. Journal of Algorithms (11), 85–101 (1990)
8. Gupta, S., Kececioglu, J., Schaeffer, A.: Improving the practical space and time efficiency of the shortest-paths approach to sum-of-pairs multiple sequence alignment. J. Comput. Biol. 2, 459–472 (1995)
9. Gusfield, D.: Algorithms on strings, trees and sequences: computer science and computational biology. Cambridge University Press, Cambridge (1997)
10. Held, M., Karp, R.: The traveling salesman problem and minimum spanning trees: part ii. Mathematical Programming 1, 6–25 (1971)
11. Katoh, K., Ichi Kuma, K., Toh, H., Miyata, T.: MAFFT version 5: improvement in accuracy of multiple sequence alignment. Nucleic Acids Research 33, 511 (2005)
12. Lee, C., Grasso, C., Sharlow, M.F.: Multiple sequence alignment using partial order graphs. Bioinformatics 18(3), 452–464 (2002)
13. Lipman, D., Altschul, S., Kececioglu, J.: A tool for multiple sequence alignment. Proc. Natl. Acad. Sci. U.S.A. 86, 4412–4415 (1989)
14. Lucena, A.: Steiner problem in graphs: Lagrangean relaxation and cutting-planes. COAL Bulletin 21, 2–7 (1993)
15. Mehlhorn, K., Näher, S.: The LEDA Platform of Combinatorial and Geometric Computing. Cambridge University Press, Cambridge (1999)
16. Morgenstern, B.: DIALIGN: multiple DNA and protein sequence alignment at BiBiServ. Nucl. Acids Res. 32(2), 33–36 (2004)
17. Notredame, C., Higgins, D.G., Heringa, J.: T-Coffee: a novel method for fast and accurate multiple sequence alignment. J. Mol. Biol. 302(1), 205–217 (2000)
18. Reinert, K.: A Polyhedral Approach to Sequence Alignment Problems. PhD thesis, Universität des Saarlandes (1999)
19. Stoye, J., Evers, D., Meyer, F.: Rose: generating sequence families (1998)
20. Thompson, J.D., Higgins, D.G., Gibson, T.J.: CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. Nucleic Acids Res. 22(22), 4673–4680 (1994)
21. Thompson, J.D., Plewniak, F., Poch, O.: BAliBASE: A benchmark alignment database for the evaluation of multiple alignment programs. Bioinformatics 15(1), 87–88 (1999)
22. Walle, I.V., Lasters, I., Wyns, L.: SABmark - a benchmark for sequence alignment that covers the entire known fold space. Bioinformatics 21, 1267–1268 (2005)

SVM-Based Local Search for Gene Selection and Classification of Microarray Data

Jose Crispin Hernandez Hernandez, Béatrice Duval, and Jin-Kao Hao

LERIA, Université d'Angers,
2 Boulevard Lavoisier, 49045 Angers, France
{josehh,bd,hao}@info.univ-angers.fr

Abstract. This paper presents a SVM-based local search (SVM-LS) approach to the problem of gene selection and classification of microarray data. The proposed approach is highlighted by the use of a SVM classifier both as an essential part of the evaluation function and as a "provider" of useful information for designing effective LS algorithms. The SVM-LS approach is assessed on a set of three well-known data sets and compared with some best algorithms from the literature.

Keywords: Microarray gene expression, Feature selection, Local search, Support vector machines.

1 Introduction

With the fast advances of DNA Microarray technologies, more and more gene expression data are made available for analysis. These data can be used for various purposes, for instance, in classification of tissue samples using gene discriminator between normal and cancer samples [6,2].

Gene expression data are known to be of very high dimensions (thousands of gene expressions at least) with a small number of samples (typically under one hundred). This characteristic, known as the "curse of dimensionality", induces a difficulty for classification and requires special techniques to reduce the data dimensionality (gene selection) in order to obtain reliable predictive results.

Gene selection is a kind of feature selection [10], aiming at identifying a (small) subset of informative genes from the initial data in order to obtain high classification accuracy. In the literature there are two main approaches for feature selection: the filter approach and the wrapper approach.

In the filter approach [5], feature selection is performed without taking into account the classification algorithm that will be applied to the selected features. A filter algorithm generally relies on a relevance measure that evaluates the importance of each feature for the classification task. A typical filter algorithm ranks all the features according to their interestingness for the classification problem and selects the top ranked features. The feature score can be obtained independently for each feature, as it is done in [6] which relies on correlation coefficients between the class and each feature. The drawback of such a method is to score each feature independently and to ignore the relations between the features.

In contrast, the wrapper approach selects a subset of features that is "optimized" for a given classification algorithm. So the classification algorithm, that is considered as a black box, is run many times on different candidate subsets, and each time, the quality of the candidate subset is evaluated by the performance of the classification algorithm trained on this subset. The wrapper approach conducts a search in the space of candidate subsets. For this search problem, genetic algorithms have been used in a number of studies, see e.g. [12,11,8]. Embedded methods, a variant of the wrapper approach, use feature selection as a part of the training process in which the learning algorithm is no more a simple black box. One example of an embedded method is proposed in [7] with recursive feature elimination using support vector machines (SVM-RFE).

In this paper, we present a Local Search approach guided by SVM which can be considered as an embedded method. In this approach, a SVM classifier is used not only to evaluate a candidate gene subset, but also to provide the local search algorithm with useful information for its search operators. As we show in the experimentation section, despite its simplicity, this SVM-based Local Search (SVM-LS) approach allows us to obtain highly competitive results on three well-known data sets when compared with some best algorithms from the literature.

2 SVM Classification and Gene Selection

It is common in wrapper approaches to use a classifier to evaluate the quality of a proposed gene subset. SVM classifiers can be used for such a purpose. In our SVM-based Local Search approach, a SVM classifier is used not only in the evaluation function of gene subsets but also in the design of LS strategies. SVM is thus a key component of our SVM-LS approach. For this reason, this section recalls the main characteristics of SVM and explains how a feature selection process can be guided by useful information provided by a SVM classifier.

2.1 Support Vector Machines

SVMs represent a class of state-of-the-art classifiers [4] that have been successfully used for gene selection and classification [7,13]. SVMs solve a binary classification problem by searching a decision boundary that has the maximum margin with the examples. SVMs handle complex decision boundaries by using linear machines in a high dimensional feature space, implicitly represented by a kernel function. In this work, we only consider linear SVMs because they are known to be well suited to the datasets that we consider.

For a given training set of labeled samples, a linear SVM determines an optimal hyperplane that divides the positively and the negatively labeled samples with the maximum margin of separation. A noteworthy property of SVM is that the hyperplane only depends on a small number of training examples called the support vectors, they are the closest training examples to the decision boundary and they determine the margin.

Formally, we consider a training set of n samples belonging to two classes; each sample is noted $\{X_i, y_i\}$ where $\{X_i\}$ is the vector of attribute values describing the sample and y_i the class label.

A soft-margin linear SVM classifier aims at solving the following optimization problem:

$$\min_{w,b,\xi_i} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{n} \xi_i \tag{1}$$

subject to $y_i (w \cdot X_i + b) \geq 1 - \xi_i$ and $\xi_i \geq 0$, $i = 1, ..., n$.

In this formulation, w is the weight vector that determines the separating hyperplane; C is a given penalty term that controls the cost of misclassification errors. To solve this optimization problem, it is convenient to consider the dual formulation [4]:

$$\min_{\alpha_i} \frac{1}{2} \sum_{i=1}^{n} \sum_{l=1}^{n} \alpha_i \alpha_l y_i y_l X_i \cdot X_l - \sum_{i=1}^{n} \alpha_i \tag{2}$$

$$st\text{: } \textstyle\sum_{i=1}^{n} y_i \alpha_i = 0 \text{ and } 0 \leq \alpha_i \leq C$$

The decision function for the linear SVM classifier with input vector X is given by: $\varphi(X) = w \cdot X + b$ with $w = \sum_{i=1}^{n} \alpha_i y_i X_i$ and $b = y_i - w \cdot X_i$.

The weight vector w is a linear combination of training samples. Most weights α_i are zero and the training samples with non-zero weights are the support vectors. The maximum margin M is given by:

$$M = \frac{2}{\|w\|} \tag{3}$$

2.2 Gene Ranking by SVM

As discussed in [7], the weights of a linear discriminant classifier can be used to rank the genes for selection purposes. More precisely, in a backward selection method, one starts with all the genes and removes iteratively the least informative gene. To determine the feature to be removed at each iteration, one considers the gene that has the least influence on the cost function of the classification process. For a linear SVM, the cost function is defined by $\frac{1}{2}||w||^2$. So given a SVM classifier with weight vector w, one can define the ranking coefficient vector c given by:

$$\forall i, c_i = (w_i)^2 \tag{4}$$

Intuitively, in order to select informative genes, the orientation of the separating hyperplane found by a linear SVM can be used. If the plane is orthogonal to a particular gene dimension, then that gene is informative, and vice versa. As we show in the next section, the coefficient vector c contains very useful ranking information that can be used to design a dedicated LS strategy.

3 SVM-LS for Gene Selection and Classification

In this section, we present our SVM-based LS approach for gene selection and classification of Microarray data. We explain the basic ingredients and their underlying rational. Our method begins by a pre-selection step where we use a filter criterion (in our case, the BW ratio introduced in [5]) to obtain a group G_p of p (typically $p \geq 75$) top ranked genes. Then our SVM-LS approach is applied to select, from G_p a gene subset of smaller size (typically less than 20 genes).

3.1 Representation and Search Space

A candidate solution $s = < s^g, s^c >$ is composed of two parts s^g and s^c called respectively *gene subset vector* and *ranking coefficient vector* [8]. The first part, $s^g = (g_1, g_2 ... g_p)$, is a binary vector of fixed length p. Each $g_i \in \{0, 1\}$ $(i = 1...p)$ corresponds to a particular gene and indicates whether or not the gene is selected. The second part, $s^c = (c_1, c_2 ... c_p)$, is a *positive* real vector of fixed length p and corresponds to the ranking coefficient vector c (Equation 4, Section 2.2) of the linear SVM classifier. s^c indicates thus for each selected gene the interestingness of this gene for the SVM classifier.

Therefore, a solution represents a candidate subset of genes with additional ranking information on each selected gene. The gene subset vector of a solution is evaluated by a linear SVM classifier and the ranking coefficients obtained during this evaluation will be used in our specialized LS strategies.

For the group G_p of p pre-selected genes, the search space is given by the set $\Omega = 2^p$ (i.e. all the possible gene subsets of p genes).

3.2 Evaluation Function

Given a candidate solution $s = < s^g, s^c >$, the quality of s (more precisely, of the gene subset part s^g) is assessed by an evaluation function f according to two criteria: the ability of s to obtain a good classification with this gene subset (C) and the maximum margin (M) given by the SVM classifier (Equation 3). More formally, the evaluation function can be written as follows:

$$f(s) = < f_C(s), f_M(s) > \qquad (5)$$

where

– $f_C(s)$ is the classification accuracy of the SVM classifier using the set of genes and applied to the given training data,
– $f_M(s)$ is simply the maximum margin of the SVM classifier, given by Equation 3 (Section 2.1).

Now given two candidate solutions s and s', it is possible to compare them: $f(s)$ is better than $f(s')$, denoted by $f(s) > f(s')$, if the following condition is satisfied: $f(s) > f(s') \Leftrightarrow f_C(s) > f_C(s')$ or $f_C(s) = f_C(s') \wedge f_M(s) > f_K(s')$.

So the dominating criterion is the classification accuracy, ties are broken by comparing the maximum margins, with a preference for a larger value (a larger margin indicates a better discrimination between the two classes).

3.3 Move and Neighborhood

One of the most important features of a local search algorithm is its neighborhood. In a local search algorithm, applying a move operator mv to a candidate solution s leads to a new solution s', denoted by $s' = s \oplus mv$. Let $\Gamma(s)$ be the set of all possible moves which can be applied to s, then the neighborhood $N(s)$ of s is defined by: $N(s) = \{s \oplus mv | mv \in \Gamma(s)\}$.

In our case, the move is based on the drop/add operation which removes a gene g_i from the solution s and add another gene g_j. Moreover, the move operator is defined in such a way that it integrates semantic knowledges of the gene selection and classification problem. More formally, let $s =< s^g, s^c >$ with $s^g = (g_1, g_2 ... g_p)$ and $s^c = (c_1, c_2 ... c_p)$, define:

- $i = ArgMin_j\{c_j | c_j \in s^c \wedge c_j \neq 0\}$, i.e. i identifies the gene g_i which has the smallest ranking coefficient c_i and thus is the least relevant gene,
- $O = \{j | g_j \in s^g \wedge g_j = 0\}$, i.e. O is the set of non selected genes in the current solution s

Then our move operator drops, from the current solution, g_i (identified by the above index i) which is the least informative gene among the selected genes and adds a non selected gene g_j ($j \in O$). This can be formally written as: $mv(i, j) = (g_i : 1 \rightarrow 0; g_j : 0 \rightarrow 1)$.

Clearly, for two neighbor solutions $s =< s^g, s^c >$ and $s' =< s'^g, s'^c >$, the hamming distance between s^g and s'^g is exactly two. Moreover, one sees that the size of this neighborhood is equal to $|O|$ and bounded by p, the length of s.

3.4 Local Search Algorithms

Local search (LS) is a class of general and powerful heuristics methods [9]. For our SVM-LS approach, we implemented three LS algorithms: steepest descent (SD), Tabu Search (TS) and Iterative Local Search (ILS).

Steepest Descent (SD)**:** Given the current solution s, the steepest descent moves at each iteration to the *best improving* neighboring solution $s' \in N(s)$ such that $f(s') > f(s)$ and $\forall s" \in N(s), f(s") \leq f(s')$. Notice that SD needs no parameter and stops when no improving neighbor can be found in the neighborhood, at which point the last solution is the best solution found and corresponds to a local optimum.

Tabu Search (TS)**:** From the steepest descent SD, one can obtain a basic TS algorithm by adding a tabu list (see below). At each iteration, the current solution s is replaced by the best neighboring solution s' that is not forbidden by tabu list, i.e. $s' \in N(s)$ such that $\forall s" \in N(s), f(s") \leq f(s')$ and $s' \notin \bar{S}$ where $\bar{S}$ is the set of solutions currently forbidden by tabu list. Notice that contrary to the SD algorithm, the selected neighbor s' may or may not be better than s. The TS algorithm stops when a fixed maximum number of iterations is reached or when all the moves become tabu.

The main role of a tabu list is to prevent the search from cycling. In our case, the tabu list is implemented as follows. Each time a move $mv(i, j)$ is carried out, i.e. gene g_i is dropped and gene g_j is selected, g_i is recorded in the tabu list for the next k iterations. Consequently, g_i cannot be reselected during this period. The value of k is determined experimentally and varies typically from k_{min} to k_{max}. Notice that such a tabu list does not forbid a newly selected gene g_j to be removed soon after its selection if its ranking coefficient is very weak.

Iterate Local Search (ILS)**:** ILS uses a local search strategy (e.g. Descent or TS) to reach a local optimum s^*, at which point the search applies a perturbation operator to the local optimum solution to allow further search progress. ILS can be combined with any local search algorithm. Here, we consider the combination with TS, denoted by ILS^{TS} because this is the best combination we have found. More precisely, ILS^{TS} iterates two phases: a TS phase to reach a local optimum s^* and a perturbation to diversify the search. Our perturbation operator changes the best local optimum s^* in a controlled way and is based on the evaluation function; the second to the fifth best neighbors are successively tried in order to continue the search process. Otherwise the search stops.

3.5 Initial Solution

The initial candidate solution can be randomly created with a risk of being of bad quality. For this reason, we devise a simple way to obtain a "not-too-bad" initial solution as follows. We generate randomly l solutions such that the number of genes in each solution varies between $p * 0.9$ and $p * 0.6$ (p being the number of pre-selected genes by a filter, see the beginning of Section 3), from which the best solution according to the evaluation function (see Equation 5) is taken.

3.6 The General SVM-LS Procedure

The general SVM-LS procedure is shown in Algorithm 1. It is composed of two repeated main phases: SVM-LS phase for gene selection (Line 7) and gene reduction phase (Line 8). At line 7, a SVM-LS algorithm (with any of the above LS algorithms) is used to search for the best gene subset of a given size. After each LS phase, gene reduction is achieved by deleting the least relevant gene (i.e., the gene with the least ranking coefficient) from the best gene subset given by the SVM-LS phase, from which point a new SVM-LS search is re-applied. This two-stage process stops when removing the least interesting gene worsens the classification accuracy on the training data.

4 Experimental Results

In this section we present two comparative studies. The first compares the different LS algorithms presented in Section 3: SD, TS, ITS^{TS}. In the second study, we compare the results of our SVM-LS approach with SVM-RFE as well as three other state-of-the-art algorithms from the literature.

Algorithm 1. General SVM-LS Procedure

```
1: Input: $G_p$, i.e. a group of $p$ pre-selected genes with a filter
2: Output: $s^g$, the set of selected (most informative) genes
3: Generate an initial set of genes $s^g$ (section 3.5)
4: repeat
5:    Evaluate $s^g$ using the SVM classifier on the training data (section 2) and fill $s^c$
6:    $s = (s^g, s^c)$ /* $s$ is the current solution */
7:    $s$= SVM-LS($s$) /* LS phase: apply SVM-based local search to improve current
      solution $s = (s^g, s^c)$ */
8:    $s^g = s^g - \{g_i\}$ /* Gene reduction phase: remove the least informative gene from
      the best solution found by SVM-LS phase */
9: until (stop condition is verified)
```

4.1 Data Sets

We applied our approach on three well-known datasets that concern colon cancer, leukemia and lymphoma. These data sets have largely been used for benchmarking feature selection algorithms, for instance in [14,13,11].

The colon cancer data set, first studied in [2], contains 62 tissue samples (22 normal and 40 anomal), each with 2000 gene expression values. The data set is available at http://www.molbio.princeton.edu/colondata

The leukemia data set, first studied in [6], consists of 72 tissue samples, each with 7129 gene expression values. The samples include 47 acute lymphoblastic leukemia (ALL) and 25 acute myeloid leukemia (AML). The original data are divided into a training set of 38 samples and a test set of 34 samples. The data set is available at http://www-genome.wi.mit.edu/cancer/

The lymphoma data set, first analyzed in [1], is based on 4026 variables describing 96 observations (62 and 34 of which are respectively considered as abnormal and normal). The data set is available at http://www.kyb.tuebingen.mpg.de/-bs/people/weston/l0

Notice that prior to running our method, we apply a linear normalization procedure to each data set to transform the gene expressions to mean value 0 and standard deviation 1.

4.2 Protocol for Experimentations and Comparison Criteria

To avoid the problem of selection bias which leads to over-optimistic estimations, we adopt the experimental protocol suggested in [3]. For each SVM-LS algorithm and each data set, 50 independent experiments are carried out. For each of these experiments, the data set samples are first randomly partitioned into a training set L and a testing set T ((L, T) respectively fixed at (50,12), (38,34) and (60,36) for “Colon”, “Leukemia” and “Lymphoma”). The training set L is then used by the SVM-LS algorithm to determine the best gene subset G (smallest size and highest classification accuracy on the samples of L). Finally, the selected gene subset G is evaluated on the testing samples of T using the SVM classifier. The resulting classification accuracy and the size of G are used for calculating the averaged statistics.

For comparison, we use two criteria: *averaged classification accuracy (Acc)* on the testing samples and the *averaged number of selected genes (NG)* over these 50 independent experiments. Computing time is not reported, but let us mention that one experiment on one data set takes about 20 minutes on a typical PC (Pentium Centrino Duo, 1.2MB).

4.3 Results and Comparisons

Comparison of the Three LS Algorithms. Table 1 shows the results of our SVM-LS approach using the three different LS algorithms: SD, TS and ILS^{TS}. One can rank these LS algorithms as follows: ILS^{TS} > TS > SD (> means "better than"). Indeed, ILS^{TS} performs globally the best even if for Leukemia, SD obtains a better prediction accuracy (92.52% against 91.94%), but requires more genes (6.04 against 3.14). The results of TS are also globally good, followed by the simple descent. Comparing these results with those showed in the next two tables will allow us to better assess the interest of the SVM-LS approach.

Table 1. Comparison of SVM-LS algorithms based on the classification accuracy on test set (Acc) with standard deviation and the number of selected genes (NG) with standard deviation

	SD		TS		ILS^{TS}	
Dataset	*Acc*	*NG*	*Acc*	*NG*	*Acc*	*NG*
Colon	84.84%±9.17%	15.32±1.83	85.50%±8.21%	11.16±2.81	**87.00%**±7.36%	**08.20**±2.09
Leukemia	**92.52%**±3.42%	06.04±1.38	92.47%±3.36%	04.74±1.32	91.94%±4.06%	**3.14**±1.08
Lymphome	92.11%±2.20%	17.04±2.44	92.44%±1.86%	14.32±2.21	**95.44%**±2.15%	**12.46**±1.58

Table 2. Results of SVM-RFE algorithm

	Colon		Leukemia		Lymphoma	
	Acc	*NG*	*Acc*	*NG*	*Acc*	*NG*
$SVM - RFE$	85.16%±8.11%	18.32±6.07	92.35%±3.25%	4.82±2.39	92.33%±3.96%	16.40±2.51

Comparison with SVM-RFE. The proposed approach is somewhat related to the well-known SVM-RFE approach [7]. With SVM-RFE, one starts with all features and remove iteratively the "least relevant" feature (according to the SVM classifier). Notice that SVM-RFE is fully greedy; a wrongly eliminated gene can never be reselected afterwards. Table 2 shows the results of SVM-RFE obtained under the same experimental conditions. Comparing Tables 2 and 1, one observes that SVM-RFE performs better than the pure decent algorithm, but is outperformed by TS and ILS^{TS}. This confirms the interest of using LS to explore the search space of a fixed size before gene elimination.

Comparison with State-of-the-art Approaches. Table 3 shows the results of three other best performing selection algorithms [14,13,11]. We have chosen these references because they use the same or similar experimental protocol to

Table 3. Comparison with three other SVM-based based selection methods (the symbol - indicates that the paper gives no information for the concerned dataset)

	[14]		[13]		[11]	
Dataset	*Acc*	*NG*	*Acc*	*NG*	*Acc*	*NG*
Colon	85.83%±2.0%	20	82.33%±9%	20	81.00%±8.00%	4.44±1.74
Leukemia	-	-	-	-	90.00%±6.00%	3.16±1.00
Lymphome	91.57%±0.9%	20	92.28%±4%	20	93.00%±4.00%	4.42±2.46

avoid selection bias. Once again, one observes that the SVM-LS approach (in particular with TS and ILS^{TS}) is very competitive since its results often dominate these reference methods with a higher classification accuracy and smaller set of selected genes.

5 Conclusion

In this paper, we have presented a SVM-based Local Search approach for gene subset selection and classification with two distinguished and original features. First, the evaluation function of our LS algorithms is based not only on the classification accuracy given by the SVM classifier, but also on the its maximum margin. Second, the ranking information provided by the SVM classifier is explicitly exploited in the LS strategies. These two features ensure that the SVM-LS approach is fully dedicated to the targeted problem and constitute its basic foundation.

Using an experimental protocol that avoids the selection bias problem, the SVM-LS approach is experimentally assessed on three well-known data sets (Colon, Leukemia and Lymphoma) and compared with four state-of-the-art gene selection algorithms. The experimental results clearly show that the proposed approach competes very well with the reference methods in terms of the classification accuracy and the number of selected genes. The proposed approach has an additional and important advantage over the filter methods and SVM-RFE. Indeed, SVM-LS allows us to generate multiple gene subsets of high quality, which can be used for further analysis and data mining purpose.

This study shows that local search constitutes a simple, yet powerful approach for gene selection and classification of microarray data. Its effectiveness depends strongly on how semantic information of the given problem is integrated in its basic operators such as neighborhood and evaluation function. Finally, it is clear that the proposed approach can easily be combined with other ranking and classification methods.

Acknowledgments. We acknowledge that the work is partially supported by the French Ouest Genopole® and the Region "Pays de La Loire" via the BIL (Bioinformatique Ligérienne) Project. The first author of the paper is supported by a Mexican PROMEP scholarship. We thank the reviewers of the paper for their useful comments.

References

1. Alizadeh, A., Eisen, M.B., Davis, E., et al.: Distinct types of diffuse large B–cell lymphoma identified by gene expression profiling. Nature 403, 503–511 (2000)
2. Alon, U., Barkai, N., Notterman, D.A., Gish, K., Ybarra, S., Mack, D., Levine, A.J.: Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. Proc. Natl. Acad. Sci. USA 96, 6745–6750 (1999)
3. Ambroise, C., McLachlan, G.J.: Selection bias in gene extraction on the basis of microarray gene-expression data. Proc. Natl. Acad. Sci. USA 99(10), 6562–6566 (2002)
4. Boser, B.E., Guyon, I., Vapnik, V.: A training algorithm for optimal margin classifiers. In: Proceedings of the Fifth Annual Workshop on Computational Learning Theory, pp. 144–152. ACM Press, New York (1992)
5. Dudoit, S., Fridlyand, J., Speed, T.P.: Comparison of discrimination methods for the classification of tumors using gene expression data. Journal of the American Statistical Association 97(457), 77–87 (2002)
6. Golub, T.R., Slonim, D.K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J.P., Coller, H., Loh, M.L., Downing, J.R., Caligiuri, M.A., Bloomfield, C.D., Lander, E.S.: Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. Science 286, 531–537 (1999)
7. Guyon, I., Weston, J., Barnhill, S., Vapnik, V.: Gene selection for cancer classification using support vector machines. Machine Learning 46(1-3), 389–422 (2002)
8. Hernandez Hernandez, J.C., Duval, B., Hao, J.K.: A genetic embedded approach for selection and SVM classification of microarray data. In: Marchiori, E., Moore, J.H., Rajapakse, J.C. (eds.) EvoBIO 2007. LNCS, vol. 4447, pp. 90–101. Springer, Heidelberg (2007)
9. Hoos, H., Stutzle, T.: Stochastic Local Search: Foundations and Applications. Morgan Kaufmann Publishers Inc., San Francisco (2004)
10. Kohavi, R., John, G.H.: Wrappers for feature subset selection. Artificial Intelligence 97(1-2), 273–324 (1997)
11. Paul, T.K., Iba, H.: Selection of the most useful subset of genes for gene expression-based classification. In: Proceedings of the 2004 Congress on Evolutionary Computation, pp. 2076–2083. IEEE Press, Los Alamitos (2004)
12. Peng, S., Xu, Q., Ling, X.B., Peng, X., Du, W., Chen, L.: Molecular classification of cancer types from microarray data using the combination of genetic algorithms and support vector machines. FEBS Letters 555(2), 358–362 (2003)
13. Rakotomamonjy, A.: Variable selection using svm-based criteria. Journal of Machine Learning Research 3, 1357–1370 (2003)
14. Weston, J., Elisseeff, A., Scholkopf, B., Tipping, M.: The use of zero-norm with linear models and kernel methods. Journal of Machine Learning Research 3(7-8), 1439–1461 (2003)

GENFOCS – A Comparative Tool on Gene Finding with Sensitivity and Specificity

M.A. Lijo Anto[1], G. Gopakumar[1], Achuthsankar S. Nair[1], and Indira Ghosh[2]

[1] Centre for Bioinformatics, University of Kerala, Trivandrum, Kerala, India
[2] Bioinformatics Centre, University of Pune, Pune, India
lama@sooryakiran.com

Abstract. This work proposes a versatile tool (GENFOCS) for comparative analysis of predictions of selected gene-finders[1]. The objective of this tool is not necessarily to find better predictions, but to provide a platform for analyzing the predictions of different gene finding tools/techniques. The tool does comparative analysis of individual predictions over the queried DNA sequence. This work significantly extended the analysis to eukaryotes and prokaryotes as well. GENFOCS uses sensitivity and specificity as measure to analyze the predictions of selected organism[2]. This online[3] tool generates various outputs like graphics, charts and tables.

Keywords: Comparative analysis of gene predictions, sensitivity and specificity.

1 Introduction

Recent developments in sequencing technology escalated the size of biological databases dramatically. After the sequencing is done, rummaging for the potential genes in the sequenced genome is one of the important tasks to be carried out. This process known to be gene-finding, paved path to several computational gene-finding techniques/tools. But the most assessing factor in gene prediction relays greatly on positioning of the genes, with its attributes rightly. The interest of this work is to set up a novel approach, by which he/she could analyze the predictions of different gene-finders. This tool facilitates analysis for both eukaryotes and prokaryotes.

1.1 GENFOCS – Introduction

GENFOCS comes with many interesting features. It allows user to do a comparative run of gene-finders for a set of selected organisms (Arabidopsis thaliana, Human, Rice), with the sequence of his/her interest. The tool can run individual gene-finders with all their default options provide. The outputs generated by individual gene-finders will be visualized under a single window. Also, the outputs can visualize in

[1] Gene-finders considered were on the basis of approaches obtained.
[2] Currently limited to *Arabidopsis thaliana*.
[3] Deployed on a local server.

tabular as well as graphical forms. The comparative analysis based on sensitivity and specificity of gene-finders at various levels, for the organism (now limited to *Arabidopsis thaliana*) is the key feature of this tool. All the gene finding tools, GENFOCS selected, are free to use for academic purpose. Genscan (C. Burge et.al, 1997), Glimmerhmm (M. Pertea et al., 2004) and Geneid (M. Burset et al., 1996) are the gene-finders selected under Eukaryotes. And under prokaryotes, Glimmer2 (A.L Delcher, 1999) and Genemark.hmm (Lukashin A. and Borodovsky M, Nucleic Acids Research, 1998) were selected for analysis. The comparison of Glimmer2 output with that of Genemark.hmm is limited, since the offline implementation of Genemark.hmm was not possible.

Before generating the analyzed result, GENFOCS dynamically collects essential information predicted by gene-finders and store it temporarily for future processes. Using the collected information and available GFF[4] file, the tool finds the sensitivity and specificity measures at gene level and exon level.

In the case of eukaryotes, the tool displays the features like number of genes, DNA strand (forward/backward), number of exons per gene, type of exons (Initial/Intermediate/Terminal),type of gene(single exon/multiple exon), starting and ending nucleotide positions of each exon, length of individual exons in nucleotides etc. Under prokaryotes the features displayed are number of ORFs, starting and ending nucleotide positions of ORFs, length of ORFs etc. User can view the individual outputs generated by respective gene-finders during the simultaneous run for a given input sequence. Users can also do individual run of gene-finders, by applying all the options and parameters provided along with the tools.

2 Methodology

This work has divided into two major divisions, for eukaryotes and prokaryotes. Under each division, user is allowed to comparative run as well as individual run of gene-finders. If the user selects the individual run for a particular predictor, then a new window for the selected gene-finder would be displayed. Initially, the user has to select one of the organisms from the list (Human, Arabidopsis or Rice). Then, the user can paste or upload the corresponding DNA sequences. GENFOCS is limited to handle input sequence[5] length of one million base pairs on comparative run. Sensitivity and specificity are calculated for the queried sequence with the help of GFF file information. The predicted output and GFF information are parsed separately, and then specificity and sensitivity will be calculated. Here, the details like starting nucleotide, ending nucleotide and chromosome number are the essential parameters required to supply by the user. Due to various resource constraints, sensitivity and specificity analysis is limited to *Arabidopsis thaliana*. The performance of GENFOCS depends on the computing resources available.

2.1 GENFOCS – Parsing the Predicted Outputs

GENFOCS uses direct and simple steps for parsing the individual prediction results of different gene-finders.

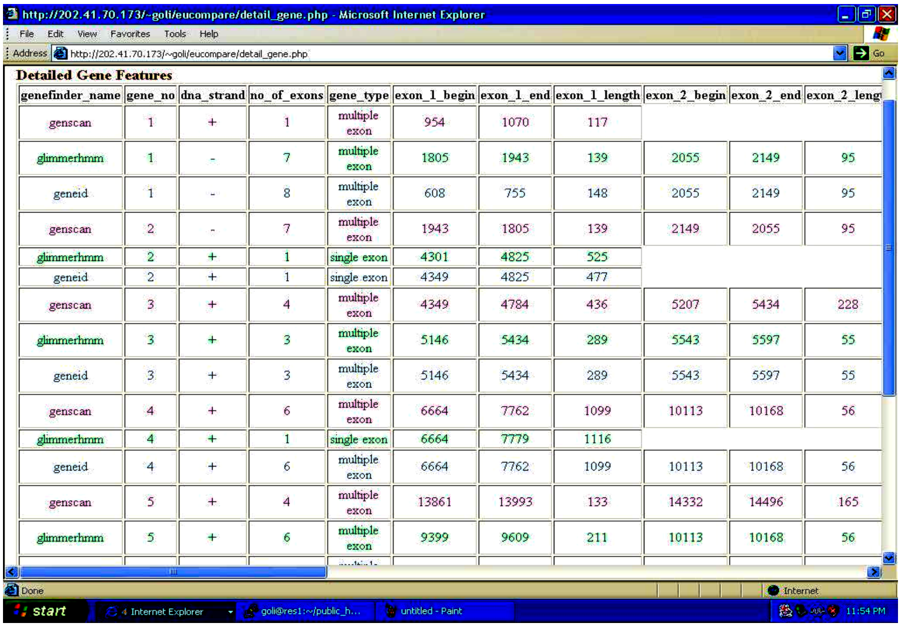
[4] General Feature Format (GFF).

[5] Since Genscan input sequence capacity is limited to one million bas pairs.

1. Open the output file (predicted result).
2. Parse the data line by line and store them in array1 temporarily.
3. Split each line using space as the separator and store each output fields in array2.
4. Search for number of base pairs in array1
5. From array2, using different index values, find the details like gene number, number of exons in that gene, DNA strand on which that gene is predicted, type of each exon, exon begin, exon end and exon length.
6. Store the above details in the database for further use.

2.2 GENFOCS Visual Representations

This tool visualizes the predicted results in various forms, to interpret the predictions in a fast and friendly way. For larger sequences, simultaneous understanding of the separate predictions of each gene finder is a tiresome task. A combined visual representation of each predicted results is more appropriate in this case. The option *GENFOCS Results* under this tool gives an option to visualize the first level information of different predictions, under a common table. This is a table displays a first level information like gene finder name, the number of base pairs, number of genes predicted by each predicting tools etc... Users can also have a detailed tabular view of features with the option named *View Tabular Output* (Fig.1). Under this option output apparently displays all the common features pertaining to each gene predicted. That is, for given sequence, the prediction result of Genscan will be displayed in first row, second row shows Glimmerhmm and third of Genid. Each feature elements beneath the


http://202.41.70.173/~goli/eucompare/detail_gene.php - Microsoft Internet Explorer

Address http://202.41.70.173/~goli/eucompare/detail_gene.php

Detailed Gene Features

genefinder_name	gene_no	dna_strand	no_of_exons	gene_type	exon_1_begin	exon_1_end	exon_1_length	exon_2_begin	exon_2_end	exon_2_leng
genscan	1	+	1	multiple exon	954	1070	117			
glimmerhmm	1	-	7	multiple exon	1805	1943	139	2055	2149	95
geneid	1	-	8	multiple exon	608	755	148	2055	2149	95
genscan	2	-	7	multiple exon	1943	1805	139	2149	2055	95
glimmerhmm	2	+	1	single exon	4301	4825	525			
geneid	2	+	1	single exon	4349	4825	477			
genscan	3	+	4	multiple exon	4349	4784	436	5207	5434	228
glimmerhmm	3	+	3	multiple exon	5146	5434	289	5543	5597	55
geneid	3	+	3	multiple exon	5146	5434	289	5543	5597	55
genscan	4	+	6	multiple exon	6664	7762	1099	10113	10168	56
glimmerhmm	4	+	1	single exon	6664	7779	1116			
geneid	4	+	6	multiple exon	6664	7762	1099	10113	10168	56
genscan	5	+	4	multiple exon	13861	13993	133	14332	14496	165
glimmerhmm	5	+	6	multiple exon	9399	9609	211	10113	10168	56

Fig. 1. Output displayed by selecting *View Tabular Output* option

corresponding column headings being fixed based on the generic characteristics of gene-finders. For example, under the column heading gene_no, number of gene predicted by the first tool will be displayed, and followed by that of second tool, then by that of third one. The other important features included are number of exons, type of gene (multiple/single exon) etc… Separate color schemes are used to represent the details of different gene-finders, for an easy discriminated view of the tabular result. By placing mouse over each element in the tabular view, GENFOCS can show a small description of information belonging to that element. This feature helps users by not scrolling back to the top for checking the feature name.

Another interesting option *View Graphical Comparison* (Fig. 2.) gives a high level graphical presentation of different prediction. This option displays two graphs, the first graph shows a bar diagram of Gene-finders Vs Number of predicted genes. And the second graph shows pie charts. Pie chart is a direct representation of the percentage of coding and non coding nucleotides present in the given DNA sequence. The percentage values are calculated by the following formula:

$$\begin{gathered}\text{Coding\%} = (\text{total exon length}/\text{total number of base pairs}) * 100 \\ \text{Non coding\%} = 100 - \text{Coding\%}\end{gathered} \tag{1}$$

Below, the pie charts show the result for Genscan, Glimmerhmm and Geneid. These charts give highg level information about the amount of coding/non coding nucleotides identified by individual gene-finders. The following figure shows the graphs generated by GENFOCS.

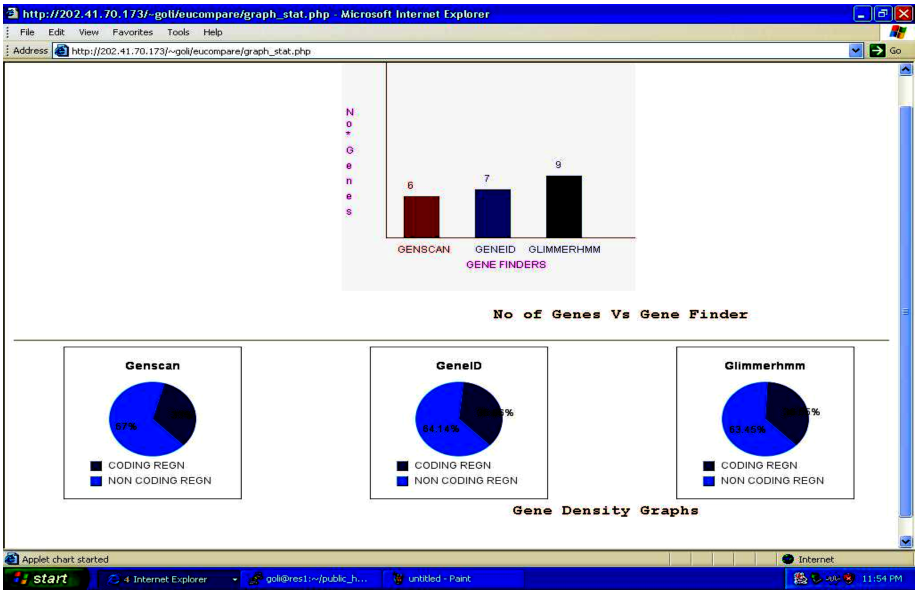


Fig. 2. Diagrams inside the window show the number of gene predicted and gene density

2.3 GENFOCS – Sensitivity and Specificity

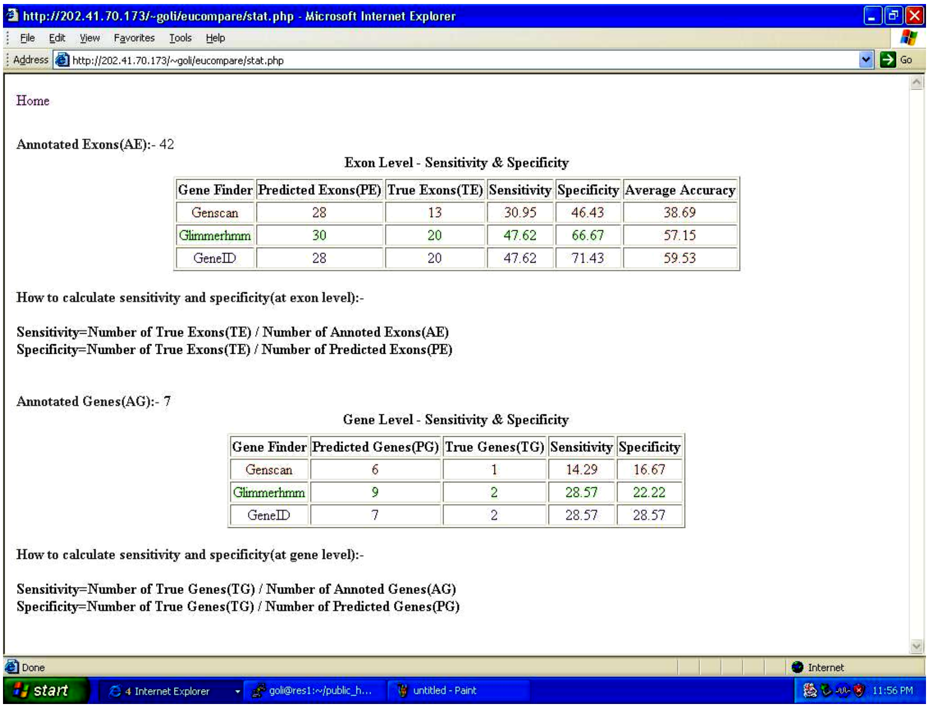
The following steps are used to measure the sensitivity and specificity by GENFOCS.

1. Open the GFF file for the specified organism and chromosome.
2. Split the GFF file line by line and store in array1.
3. Split each above line, with space as the separator and store in array2.
4. Look for the substring *gene* in each GFF line, using array1.
5. If (match found) {*if* (begin and end nucleotides within the allowed limits) {(a) Store gene start, gene end positions in separate arrays.
6. (b) Store the exon details of current gene, like exon begin and exon end, in separate arrays. *}}*
7. For each gene finder, do the following:
8. For each gene finder, check the starting and ending of their exons with the exons got from GFF file. If both begin and end exactly matches, they are taken as True Exons.
9. The total number of exons, within the given range, from the GFF file is taken the number of Annotated Exons.
10. The total number of exons predicted by individual gene-finders is taken as Predicted Exons.
11. If for any genes predicted by individual gene-finders, whose first exon's starting position and last exon's ending position are equal with the starting and ending positions of genes from GFF file, then they are taken as True Genes.
12. The total number of genes predicted by individual gene-finders is taken as Predicted Genes.
13. The total number of genes, within the given range, from the GFF file is taken the number of Annotated Genes.

The sensitivity and specificity are calculated by:

$$\begin{gathered}
\text{Sn\_gene} = (TG/AG) * 100\% \\
\text{Sp\_gene} = (TG/PG) * 100\% \\
\text{Sn\_exon} = (TE/AE) * 100\% \\
\text{Sp\_exon} = (TE/PE) * 100\% \\
\text{Avg\_gene} = ((\text{Sn\_gene} + \text{Sp\_gene})/2) * 100\% \\
\text{Avg\_exon} = ((\text{Sn\_exon} + \text{Sp\_exon})/2) * 100\%
\end{gathered} \tag{2}$$

Where *TG* is True Genes, *AG* is Annotated Genes, *PG* is Predicted Genes, *TE* is True Exons, *AE* is Annotated Exons and *PE* is Predicted Exons. The figure (fig. 3.) shows the sensitivity and specificity measured for gene and exon level on a sample sequence (*Arabidopsis thaliana).*


http://202.41.70.173/~goli/eucompare/stat.php - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address http://202.41.70.173/~goli/eucompare/stat.php
Go
Home
Done
Internet
start
4 Internet Explorer
goli@res1:~/public_h...
untitled - Paint
11:56 PM

Annotated Exons(AE):- 42

Exon Level - Sensitivity & Specificity

Gene Finder	Predicted Exons(PE)	True Exons(TE)	Sensitivity	Specificity	Average Accuracy
Genscan	28	13	30.95	46.43	38.69
Glimmerhmm	30	20	47.62	66.67	57.15
GeneID	28	20	47.62	71.43	59.53

How to calculate sensitivity and specificity(at exon level):-

Sensitivity=Number of True Exons(TE) / Number of Annoted Exons(AE)
Specificity=Number of True Exons(TE) / Number of Predicted Exons(PE)

Annotated Genes(AG):- 7

Gene Level - Sensitivity & Specificity

Gene Finder	Predicted Genes(PG)	True Genes(TG)	Sensitivity	Specificity
Genscan	6	1	14.29	16.67
Glimmerhmm	9	2	28.57	22.22
GeneID	7	2	28.57	28.57

How to calculate sensitivity and specificity(at gene level):-

Sensitivity=Number of True Genes(TG) / Number of Annoted Genes(AG)
Specificity=Number of True Genes(TG) / Number of Predicted Genes(PG)

Fig. 3. Sensitivity and Specificity measured for gene and exon level (*Arabidopsis thaliana)*

3 Discussion

For evaluating the accuracy of eukaryotic gene-finders, the following data is first collected from the NCBI website.

Organism selected: *Arabidopsis thaliana*
Sequence details

- Chromosome: 1 Contig: NC 003070.5 start: 1 stop: 20000
- Chromosome: 2 Contig: NC 003071.3 start: 1 stop: 20000
- Chromosome: 3 Contig: NC 003074.4 start: 1 stop: 20000
- Chromosome: 4 Contig: NC 003075.3 start: 1 stop: 20000
- Chromosome: 5 Contig: NC 003076.4 start: 1 stop: 20000
- Strand: plus

GENFOCS is allowed to run in the comparative mode for each case and the various sensitivity and specificity measures are calculated:
For chromosome: 1 the value of AE=25. The values of PE and TE for the three eukaryotic gene-finders are:

1. Genscan: PE=12, TE=5
2. Glimmerhmm: PE=14, TE=8
3. Geneid: PE=12, TE=8.

Table 1. Shows the exon level accuracy of each eukaryotic gene-finder for chromosome 1 (*Arabidopsis thaliana*)

Gene Finder	Sn_exon	Sp_exon	Avg_exon
Genscan	20.00	41.67	30.84
Glimmerhmm	32.00	57.14	44.57
Geneid	32.00	66.67	49.34

Table 2. Shows the average accuracy of each eukaryotic gene-finder for the five set of sequences selected (*Arabidopsis thaliana*)

Gene Finder	Avg_exon	Avg_gene
Genscan	20.00	41.67
Glimmerhmm	32.00	57.14
Geneid	32.00	66.67

A complete run of the entire genome may beget apparent and better results based on the values of sensitivity and specificity. But the limited resources available for GENFOCS were not sufficient to support an entire genome run. The result shows that all the eukaryotic gene-finders considered are more accurate in exon level sensitivity and specificity compared to than of gene level. In this example, the gene-finders selected can't exactly locate the starting nucleotide of first exon.

4 Conclusion and Future Works

This proposed tool uncovered that; further woks need to be conducted on this analyzing approach. From the studies it is found that the sensitivity and specificity of gene-finders increases from gene level to exon level. It also shows an increase from exon level to nucleotide level. We suppose this work is just the beginning, and future enhancements like including other prediction techniques, dynamic interfaces and implementing various constraints to analyze the predictions etc can make this tool more beneficial to researchers.

Acknowledgments

This work has done at Bioinformatics Centre, University of Pune, Pune, also received supports form Centre for Bioinformatics, Dept of Zoology, University of Kerala. And we acknowledge our special thanks to Mr. G. Gopakumar, Center for Bioinformatics, University of Kerala for the indeed assistance given.

References

1. Burge, C., Karlin, S.: Prediction of complete gene structure in human genomic DNA. J. Mol. Biol., 268–278 (1997)
2. Burset, M., Guig, R.: Evaluation of gene structure prediction programs. Genomics, 353–357 (1996)
3. Delcher, L., et al.: Improved microbial gene identification with glimmer. Nucleic Acids Research, 4636–4641 (1999)
4. Salzberg, S., et al.: Microbial gene identification using interpolated markov models. Nucleic Acids Research, 544–548 (1998)
5. Pertea, M., Majoros, W.H., Salzberg, S.L.: Tigrscan and glimmerhmm: two open source ab initio eukaryotic gene-finders. Bioinformatics, 2878–2879 (2004)
6. Lukashin, A., Borodovsky, M.: GeneMark.hmm: new solutions for gene finding. Nucleic Acids Research 26(4), 1107–1115 (1998)

Gene Machine© – A Hardware/Software Platform for Analyzing Genome Data

Roger Marshall

Computer Science and Technology Department
Plymouth State University, Plymouth NH 03264, USA
rgmarshall@plymouth.edu

Abstract. Gene Machine© is a reconfigurable hardware/software architecture which can be used in studying a variety of problems in the field of computational biology. The key architectural features of Gene Machine© are a) hardware implementation of nth order left-to-right Markov Model where the user ca n specify the number of states of the model and the order of the Markov model; b) cache memory for data input/output; c) shift, and logical instructions which operate at the singleton or set level, and d) floating point and integer arithmetic operations, and variable length operands whose lengths are based on the semantics of the input data. Gene Machine© can be programmed to perform a diverse set of computations such as nucleotide and protein sequence comparisons, pair wise and multiple sequence alignments, and secondary and tertiary structure predictions for DNA, RNA and protein sequences.

Keywords: Genetic sequences alignment, hidden Markov model, computer architecture.

1 Introduction

Recent efforts in designing new hardware and software have focused on abstract machines, molecular computing, etc. [1], [2], [3], [4]. Software packages and algorithms [5], [6], [7], [8], [9] for aligning gene sequences employ indexing, heuristics and fast comparison techniques to compare databases of sequences with a query sequence. The sequences retrieved are dependent on what scoring mechanisms are used. In gene identification algorithms, one typically looks for features such as start and stop codons, promoter sites, etc. Sequence folding studies attempt to find a set of aligned sequences and compute the degree of conservation of each amino acid in the target protein sequence. Gene Machine's architecture has been designed to execute the sequence alignment and scoring algorithms in an efficient manner.

2 Left-to-Right Hidden Markov Models [LMM]

Hidden Markov models (HMM) such as profile HMM and motif HMM [10], [11] have been successfully used in bioinformatics to model dynamic data sequences and random processes using a minimum amount of memory. An HMM can be represented by a finite state machine where transitions between states are specified by some probability function and the output at any given state is also probabilistically specified. Markov models of various orders have been used to find coding areas in nucleotide sequences. Profile HMMs are left-to-right linear models containing match, add and delete states along with their associated probabilities. Motif HMMs are based on many strings of matched states which is appropriate for modeling ungapped blocks of sequence consensus, and a limited number of insert states which is appropriate for modeling spaces between ungapped blocks. In this paper discuss a hidden Markov model architecture in which all state transitions are left-to-right time synchronous.

3 Architectural Details

The principal components of the Gene Machine are a) Query Sequence Vector, b) Consensus Sequence Vector, c) Hidden Sequence Vector, d) State Transition Vector, e) left-to-right Markov Module (LMM), and f) Baum-Welch Module. See Figure 1.

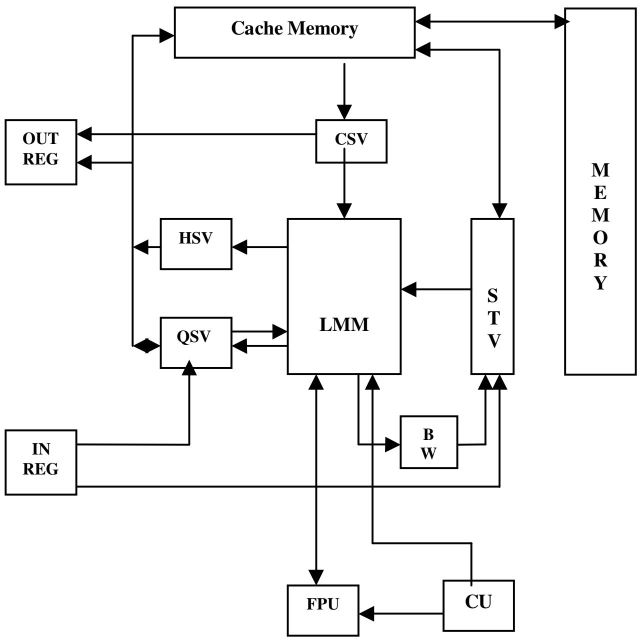


Fig. 1. Machine architecture

3.1 Vector Modules Description

3.1.1 Query Sequence Vector [QSV]

Purpose: Contains the target sequence which is being analyzed/compared with sequences in the database/cache. The dimensionality of QSV is N where N is the number of nodes in the LMM. If the target sequence is of length L, the target sequence vector will have 2L fields comprised of L features and L associated scores. Initially the score fields will be empty; these fields will be filled upon execution of the LMM module. If each feature is assigned f bits and each score field p bits, the length of the target sequence vector will be $L\ (f + p)$ bits. If the number of bits used to store each feature is also p, then the length of QSV will be $2\,L\,p$ bits.

3.1.2 Hidden Sequence Vector [HSV]

Purpose: Contains the hidden state sequence in the LMM that was followed in obtaining an optimal score for aligning the target sequence vector with the consensus sequence vector. If the target sequence is of length L, there can be only (L-1) state transitions in the LMM from start to finish. Therefore, the HSV will contain (L-1) fields, each specifying a state transition or path. Since each field value is an integer between 2 and L, to specify this value only requires $\log_2 (L)$ bits per field. The size of HSV is N^2 bits (N n-bit words).

3.1.3 Consensus Sequence Vector [CSV]

Purpose: contains the consensus sequence which is used as the basis against which the data in the target sequences are compared. The set of output symbols and their associated emission probabilities or scores for all the nodes is stored in CSV. If the consensus sequence is of length M, the consensus sequence vector will have 2M fields comprised of M features and M associated scores. The value of M cannot exceed the maximum number of nodes N in the LMM module configuration. Since there are N nodes and each node requires 2 N p bits, the size of CSV is given by $2\,N^2\,p$ bits.

3.1.4 State Transition Vector [STV]

Purpose: To specify the number of nodes, the probability-based interconnection structure of the nodes and the order of the Markov process used in the N-node hidden Markov model. The dimensionality of STV is N. There are N fields, one for each node in the LMM. Each field, in turn, is comprised of two subfields – one to store interconnection of the node, the other to store the interconnections probabilities. In an LMM with N nodes, each node can only be connected to itself and/or to one or more nodes to its right. The last node may be connected only to itself. Each transition is associated with a probability value. Setting a probability to zero implies the connection does not exist. If a k^{th} - order Markov model is desired this means ensuring that k is less than N. For any node J, where J is between 1 and N inclusive, the size of the state transition set is given by (N – J + 1); likewise, for the probability set associated

Hardware	**Format**				**Number of bits**
QSV	$[f_1\ s_1]$	$[f_2\ s_2]$	[….]	$[f_l\ s_l]$	$2\,N\,p$
HSV	$[\,P_1\,]$	$[\,P_2\,]$	[…]	$[P_{L-1}]$	$L \log_2 L$
CSV	$[f_1\ s_1]$	$[f_2\ s_2]$	[….]	$[f_M\ s_M]$	$2\,N^2\,p$
STV	$[ts_1\ p_1]$	$[ts_2\ p_2]$	[….]	$[ts_N\ p_N]$	$N^2(p+\log_2 N)$

Fig. 2. Hardware Vector Format and Size

with each node. Since any item in the state transition set is an integer between 1 and N, and each set can contain no more than N values, to represent any node's state transition set we use a bit vector of size N requires at most $N \log_2(N)$ bits. The minimum total space requirement for the state transition vector will be $N^2 [p + \log_2(N)]$ bits. See Figure 2.

3.1.5 Left-to-Right Markov Module [LMM]

Purpose: to compare the features of a given target sequence in the target or query sequence vector against the features of some specified sequence in the consensus sequence vector and return the optimal score in the QSV and the hidden state sequence in the HSV. This is a hardware implementation of the forward and backward decoding phases of the left-to-right hidden Markov algorithm. LMM is an n by n node structure where each node has the same m features and associated output probabilities as in CSV. In addition, as the LMM evolves over time, the cells will contain the computed probabilities of the evolved sequence; this will require an additional p bits. The output probabilities of all the features of the first node are set to 1 and the output probabilities of the features of the remaining (n – 1) nodes are set to zero. The module is an N by N structure labeled H containing N^2 cells. The row index represents a time step and the column index represents a state in the LMM. Therefore any reference to an arbitrary cell H_{IJ} means that at time I the machine is in state J. The maximum number of cells that contribute to the value of cell H_{IJ} is given by J and these contributing cells (labeled 'predecessor' cells) are $H_{I-1,K}$ where $1 <= K <= J$. This means the value or score in each cell is determined by the values or scores in its predecessor cells including the cell under consideration, the state transition probabilities and the output emission probabilities of the involved cells, and the particular operations or computations that are used in defining what constitutes the ultimate value or score to be assigned to a cell. In a standard LMM, the value assigned to a cell is typically the largest of several product probabilities where each product probability is specific to a predecessor cell or the negative log of the product probabilities. In Gene Machine, cell value computations can be based on a variety of operators. These operations include arithmetic, relational, logical, shift and string operations; which operators are used is dependent on which features of the CSV or QSV are involved. The value assigned to a cell can be numeric, character, string. Consider feature f (with a score of s) in the CSV which is to be compared/aligned with feature g (whose initial score is zero) in the QSV. The computed 'value' or associated score of g can be defined as follows:

$$\text{Associated score of feature g} \leftarrow [\,f \;\; Op \;\; g\,]\, s$$

where Op is some operator such as 'and', 'xor', 'or', add, multiply, greater than, etc. For example, if f and g are residues, s is a structure predictor such as 'H' for helix, and the operator Op is chosen to be 'and', then if the two residues match, we can assign the score 'H' to feature g. The format and size of LMM are determined by the lengths of QSV, STV and CSV.

3.1.6 Baum-Welch Module [BW]

Purpose: To generate an optimal Markov model based on a number of training sequences. The model's interconnections and associated probabilities are stored in STV. In this paper we do not discuss any additional details of the BW module.[12].

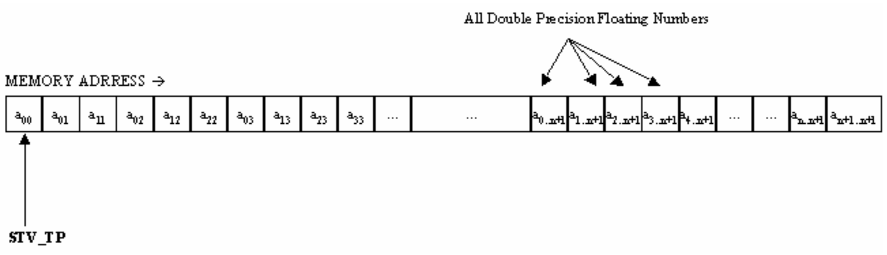

All Double Precision Floating Numbers
MEMORY ADRRESS →
a_{00} a_{01} a_{11} a_{02} a_{12} a_{22} a_{03} a_{13} a_{23} a_{33} … … $a_{0.n+1}$ $a_{1.n+1}$ $a_{2.n+1}$ $a_{3.n+1}$ $a_{4.n+1}$ … … $a_{n.n+1}$ $a_{n+1.n+1}$
STV_TP

Fig. 3. Storage Structure for Transition Probabilities (TPs)

3.2 Storage Structure and Register Organization

The LMM has a set of registers which are used to cache frequently used variables. The variables can be actual values, or addresses pointing to the RAM since all data for initializing the LMM are in the RAM. The value registers are all 8 bytes (to store double precision values). However, each of these 8 byte registers can be addressed as 8 single- byte registers or 4 short (i.e., 2-byte) registers. For example, AX is an 8 byte register. However, we can individually address the 8 single-byte registers as AB1, AB2...AB8, or 4 short registers as AS1, AS2, AS3 and AS4. This will allow the same registers to be used for byte operations as well as short or double value operations. The address registers are all 4-byte registers, allowing a total addressing space of 4 GB. See Figure 3.

4 Instruction Set

A RISC design has been chosen to handle the variable length operands (vectors) whose lengths are based on the semantics of the input data. For example, the shift instruction allows one to change window sizes, introduce gaps in sequences, and switch from nucleotide data to codon data – operations which are essential to local and global sequence alignment. Floating point operations are also needed since floating point values such as probabilities and logarithmic probability ratios, bond angles, hydrophobic scales (e.g., [13]) are involved.

5 Data Types

The various logic operations for the Gene Machine© are defined at the set level and employ non-classical logic definitions. The different types of data are 1) nucleotides and 2) amino acids interpreted as a) codons, b) forming alpha and beta secondary structures, c) hydrophilic and hydrophobic entities, and d) specified by their side chains. Biological and chemical semantic criteria have been used in choosing set names and defining set membership. For example, PB is the protein backbone set whose elements AC, BA, HF, HB stand for acidic, basic, hydrophilic and hydrophobic amino-acids, respectively. AC is itself a set which contains those amino-acids which are primarily acidic in chemical composition.

5.1 Nucleotide Data Input

We define the following basis sets:

NS = {A, T, G, C}; PR = {A, G}; PY = {C, T}; W1 = {A, T}; W2 = {C, G}

NS is the nucleotide set and A, T, G, C are the four nucleotides; PR is the purine set and PY is the pyrimidine set; W1, W2 are the Watson-Crick base pair sets. PR and PY are said to be in the purine/pyrimidine class C1. Likewise, W1 and W2 are in the Watson-Crick base pair class, C2. Note that sets within a given class are mutually exclusive.

5.2 Codon Sequences as Data Input

Basis sets:

AA = {LZ, HR, LS, MR}; MH = {MR, HR}; SZ = {LS, LZ};
MS = {MR, LS}; HZ = {HR, LZ}

AA is the codon set whose elements MR, HR, LZ and LS stand for medium redundancy, high redundancy, low stop and low start, respectively. MS and HZ are in class C1; MH and SZ in class C2. It should be pointed out here that MR, HR, LZ and LS actually represent sets whose elements are drawn from the standard symbols A, B… W, Y representing the 20 amino acids and the stop codon Z.

MR = {A, G, P, T, V}; HR = {L, R, S}
LZ = {C, D, E, F, H, K, N, Q, Y, Z}; LS = {I, M, W}

Each amino acid in set MR can be represented by any of 4 codons specific to the amino acid. Each amino acid in set LZ can be represented by any of 2 codons specific to the amino acid. Note that set LZ also contains the stop codon Z. Each amino acid in set HR can be represented by any of 6 codons specific to the amino acid. The members of set LS are mixed; one codon specifies M and W whereas 3 are needed for I. Note that set LS contains the start codon M (also methionine.)

5.3 Amino Acids Data Input for Interpreting Secondary Structures

Basis sets:

SS = {AF, AB, BF, BB}; ALFA = {AF, AB}; BETA = {BF, BB};
ALFABETA = {AF, BB}; BETALFA = {BF, AB}

SS is the secondary structure set whose elements AF, AB, BF, BB stand for alpha former, alpha breaker, beta former and beta breaker, respectively. ALFA is the alpha set and BETA is the beta set, and ALFABETA, BETALFA are the alpha-beta cross sets. ALFA and BETA are in alpha-beta cross class C2. Likewise, ALFABETA and BETALFA belong to the alpha/beta class C1.

5.4 Amino Acids Data Input Interpreted as Hydrophobic/Hydrophilic Entities

Basis sets:

PB = {AC, BA, HF, HB}; PH = {AC, BA}; HP = {HF, HB};
AHB = {AC, HB}; HFB = {HF, BA}

Table 1. Data Types

Data Type	**Basis Set**	**Class C1 Sets**		**Class C2 Sets**	
Nucleotide	{a, t, c, g}	{a, g}	{c, t}	{a, t}	{c, g}
Amino acids					
Codons	{lz, hr, ls, mr}	{ls, lz}	{mr, hr}	{mr, ls}	{hr, lz}
Secondary Structure	{af, ab, bf, bb}	{af, ab}	{bf, bb}	{af, bb}	{bf, ab}
Hydro- philic/phobic	{ac, ba, hf, hb}	{ac, ba}	{hf, hb}	{ac, hb}	{hf, ba}
Side Chain	{ln, mb, r1, r2}	{ln, mb}	{r1, r2}	{ln, r1}	{mb, r2}

PB is the protein backbone set whose elements AC, BA, HF, HB stand for acidic, basic, hydrophilic and hydrophobic, respectively. PH is the acidic/basic set and HP is the hydrophilic/hydrophobic set; AHB, HFB are the acidic-hydrophobic and base-hydrophilic cross sets. AHB and HFB are in class C1 while PH and HP in class C2.

5.5 Amino Acids Interpreted on the Basis of Side Chains

Basis sets:

SC = {LN, MB, R1, R2}; AL = {LN, MB}; AR = {R1, R2};
LR1 = {LN, R1}; R2M = {R2, MB}

SC is the side chain set whose elements LN, MB, R1, and R2 stand for linear, multi-branched, single ring and double ring, respectively. AL is the aliphatic set and AR is the aromatic set; LR1, R2M are the linear/single ring and multi-branched/double ring cross sets. LR1 and R2M are in class C1 while AL and AR are in class C2. The different data types are summarized in Table 1.

6 Logic Operators

We define ten operators - 4 unary [NOTS, COM, SUB1, SUB2S] and 6 binary [AND, OR, ANDS, XORS, XNORS]. Operators (given by symbols X and Y) whose operands are sets are distinguished from standard logical operators by appending 'S' to the operators' names. Such operators do not have associative and commutative properties. In what follows, we use the nucleotide data type sets as operands in the truth tables for the various unary and binary operators. Partial truth tables are shown in Tables 2a thru 2e. We do not go into the truth tables for the other data types.

6.1 Unary Operators

NOTS X is defined as S – {X} where X is a member of S. *This operation returns a set as the result.* **COM** X returns Y, the Watson Crick base pair of X. Thus X and Y are both in set S1 or they are both in set S2. **SUB1** X returns the transition substitution Y where X, Y are either both purines (i.e. in set S1) or both pyrimidines (i.e. in S2). **SUB2S** X returns the transverse substitution Y where Y is S2 if X is in S1, and Y is S1 if X is in S2. *The result is a set.*

Table 2. (a). Unary Operators Truth Table. **(b).** Binary logic[purine set only or pyrimidine set only] (partial). **(c).** Binary logic[Watson-Crick base-pair sets] (partial). **(d).** Binary logic [different sets within the purine-pyrimidine class] (partial). **(e).** Binary logic [different sets within Watson-Crick base-pair sets] (partial).

(a)

	Classical	Logic	Set-based	Logic
X	**COM X (Watson-Crick)**	**SUB1 X (transitions)**	**SUB2S X (transverse)**	**NOTS X (other)**
A	T	G	{C, T}	{ T, C, G }
T	A	C	{C, T}	{ A, C, G }
C	G	T	{A, G}	{ A, T, G }
G	C	A	{A, G}	{ A, T, C }

(b)

		Classical	Logic	Set-based	Logic
X	**Y**	**X AND Y**	**X OR Y**	**X XORS Y**	**X XNORS Y**
A	A	A	A	{ C, T }	{ A, G }
C	C	C	C	{ A, G }	{ C, T }

(c)

		Classical	Logic	Set-based	Logic
X	**Y**	**X AND Y**	**X OR Y**	**X XORS Y**	**X XNORS Y**
A	A	A	A	{ C, G }	{ A, T }
C	C	C	C	{ A, T }	{ C, G }

(d)

X	**Y**	**X ANDS Y**	**X ORS Y**	**X XORS Y**	**X XNORS Y**
A	C	{A, G, T}	{C, G, T}	{ G, T }	{ A, C, G, T }
	T	{A, C, G}	{C, G, T}	{ C, G}	{ A, C, G, T }
G	C	{A, G, T}	{A, C, T}	{ A, T }	{ A, C, G, T }
	T	{A, C, G}	{A, C, T}	{ A, C }	{ A, C, G, T }

(e)

X	**Y**	**X ANDS Y**	**X ORS Y**	**X XORS Y**	**X XNORS Y**
A	C	{A, T, G}	{C, T, G}	{ T, G }	{ A, C, T, G }
	G	{A, C, T}	{C, T, G}	{ C, T}	{ A, C, T, G }
T	C	{A, T, G}	{A, C, G}	{ A, G }	{ A, C, T, G }
	G	{A, C, T}	{A, C, G}	{ A, C }	{ A, C, T, G }

6.2 Binary Operators

Here there are two cases to consider – both operands may come from the same set within a class or they may come from different sets within the same class.
Case 1: Operands X, Y belong to the same set within a given class.

X **ANDS** Y is defined to be X; X **ORS** Y is defined to be Y. X **XORS** Y is defined to be another set within the same class. The result is a set. X **XNORS** Y is defined to be the set which contains both X and Y. The result is a set.

Case 2: Operands X, Y belong to different sets within a given class. The result is a set.
X **ANDS** Y is defined to be the union of the sets containing X and Y but without the element Y; X **ORS** Y is defined to be the union of the sets containing X and Y but without the element X. X **XORS** Y is defined to be the union of the sets containing X and Y but without the elements X and Y; X **XNORS** Y is defined to be the union of the set containing X and the set containing Y.

7 Execution and Performance Considerations

We assume a RISC machine with 1 cycle per phase in the instruction cycle, 1 cycle for read /write operations in load/store, and delays due to the floating point unit are K for add and M for multiply. In our design, the length of the address pointer is immaterial and does not affect read/write time because pointer arithmetic is assumed. In Table 3 we show the minimum and maximum number of cycles per instruction for various instruction categories; the computation is based on the addressing modes pertinent to an instruction category.

The utility of Gene Machine lies in being able to implement a variety of sequence alignment and scoring algorithms using a single platform. The optimal alignment can be obtained through scores defined on the basis of a single character (codon or nucleotide), secondary structure (alpha, beta, etc.), polarity, hydrophobicity, etc. This flexibility is possible because the architecture is especially tailored to executing the Viterbi search algorithm which is a fast variant of the standard dynamic programming algorithm and independent of any particular scoring mechanism. What algorithms such as Needleman-Wunsch, and Profile HMM share in common are a Viterbi-like approach to extracting the optimal alignment from a set of sequences although different scoring mechanisms are used in each of the algorithms. For example, in the Sellers algorithm, the value of cell M i j is given by M i j = max{M i-1, j-1 + S(a i, b j);

Table 3. Instruction execution time by category

Instruction	Minimum number of cycles	Maximum number of cycles
compare	3	13
jump	4	4
move or input	3	8
add	3+K	3+K
multiply	4+M	4+M
output	3	5
load/store	5	5

M i-k, j - W k ; M i, j-1 - W 1 ;0} where S and W represent similarity scores and gap/substitution penalties. Since Gene Machine is based on hidden Markov models, it is a probabilistic model that can assign likelihoods to all possible combinations of gaps, matches, and mismatches to determine the most likely sequence alignment or set of possible alignments to produce both global and local alignments.

To perform multiple sequence alignment on Gene Machine, we iteratively pick two sequences from a set of sequences and replace them with their alignment (i.e., consensus sequence) obtained by executing the Viterbi algorithm until all sequences are aligned into a single consensus sequence. The Viterbi algorithm is executed after the first sequence is loaded into LMM, the second into QSV. The resulting consensus sequence is then input into LMM and a third sequence is loaded into QSV. The process is repeated until a set of sequences has been successively refined and incorporated into a single consensus sequence. In what follows, we give a synopsis of the machine input description, the Viterbi algorithm as implemented. Let A represent the set of input symbols including the special end marker symbol and let M be the cardinality of the set Denote the sequence to be aligned by $X_1,X_2 \ldots X_m$. We require $Xi \in A$ for all $1 \leq i < m$ and X_m = end marker. V denotes the two dimensional Viterbi array and the states of the LMM are denoted by S_0,S_1.. where S_0 is the start state and S_{n+1} is the end state which always emits the end marker. Let a_{ij} denote the probability of a transition from S_i to S_j and $e_j(k)$ the emission probability of symbol X_k from S_j. Of course, $e_{n+1}(m) = 1$. The two main steps in the Viterbi algorithm are proper initialization and execution of the dynamic programming segment with an appropriate scoring mechanism as shown below.

1. Initialization:
   a. $v_0(0) = 1$; $v_j(0) = 0$ for $0 < j \leq (n + 1)$; $v_0(i) = 0$ for $1 < i \leq m$; $p_j(i) = 0$ for all i and j.
   b. $e_{(n+1)}(i) = 1$ for $i = m$, and $e_{(n+1)}(i) = 0$, for all $i \neq m$.; $a_{00} = 0 = a_{n+1.n+1}$

2. Dynamic Programming:

```
for i = 1 to m
{for j = 1 to (n + 1)
 {       $v_j(i) = e_j(i) + \max_k(v_k(i-1) + a_{kj})$ , /* $0 \leq k \leq j$ */
         $p_j(i) = \text{argmax}_k\{ v_k(i-1) + a_{kj}\}$. }   }
```

The output items are the alignment score given by $v_{n+1}(m)$ and the traceback state path p in which if $p(i) = p(i+1)$ a deletion is implied whereas $p(i+1) > p(i) + 1$ implies an insertion.

8 Concluding Remarks

Gene Machine is the first architectural design of its kind for use in bioinformatics. It incorporates instructions and data types which can used in a hierarchical manner for executing algorithms at the DNA/RNA, protein, gene, protein-protein interaction, cell and genome levels.

References

1. Hood, L.: Systems Biology and Medicine in the 21st century: Dealing with Complexity. In: Microsoft e-Science Workshop, Johns Hopkins University (October 2006)
2. Cardelli, L.: Abstract Machines of Systems Biology. In: Priami, C., Merelli, E., Gonzalez, P., Omicini, A. (eds.) Transactions on Computational Systems Biology III. LNCS (LNBI), vol. 3737, pp. 145–168. Springer, Heidelberg (2005)
3. Regev, A., Shapiro, E.: Cellular Abstractions: Cells as Computation. Nature 419, 343 (2002)
4. Benenson, Y., et al.: An Autonomous Molecular Computer for Logical Control of Gene Expression. Nature 429, 423–429 (2004)
5. Needleman, S., Wunsch, C.: A General Method Applicable to the Search for Similarities in the Amino-acid Sequence of Two Proteins. Journal of Molecular Biology 48, 443–453 (1970)
6. Higgins, D.G., Sharp, P.M.: CLUSTAL: A package for performing multiple sequence alignment on a microcomputer. Gene. 73, 237–244 (1988)
7. Altschul, S.F., et al.: A basic local alignment search tool. Journal of Molecular Biology 215, 403–410 (1990)
8. Fasman, G.D.: Prediction of Protein Structure and the Principles of Protein Conformation. Plenum, New York (1989)
9. Henikoff, S., Henikoff, J.G.: Amino acid substitution matrices from protein blocks. In: Proceedings of the National Academy of Sciences USA, vol. 89, pp. 10915–10919 (1992)
10. Parida, L., et al.: An approximation algorithm for alignment of multiple sequences using motif discovery. Journal of Combinatorial Optimization 3, 247–275 (1999)
11. Henderson, J., Salzberg, S., Fasman, K.H.: Finding Genes in DNA with a Hidden Markov Model. Journal of Computational Biology 4(2), 127–142 (1997)
12. Baum, L.E., et al.: A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains. Annals of Mathematical Statistics 40, 164–171 (1970)
13. Kyte, J., Doolittle, R.F.: A Simple Method for Displaying the Hydropathic Character of a Protein. Journal of Molecular Biology 157, 105–132 (1982)

Complex Representation of DNA Sequences

Carlo Cattani

diFarma, University of Salerno,
Via Ponte Don Melillo, 84084 Fisciano (SA) Italy
ccattani@unisa.it

Abstract. This paper deals with the symbolic representation of a DNA sequence. As indicator it is taken a complex function. A DNA sequence is investigated by using a family of wavelets. The existence of a fractal shape, patterns and symmetries are eventually shown.

Keywords: Wavelets, wavelet coefficients, short Haar wavelet transfor.

1 Introduction

In recent years, the analysis of DNA sequences has been mainly focused on the existence of hidden law, periodicities, autocorrelation [3,4,7,8,10,12,13,14,17,18]. The main task is to find (if any) some kind of mathematical rules in the nucleotide distribution. This would help us to characterize each DNA sequence in order to construct a possible classification. From mathematical point a view the DNA sequence is a symbolic sequence (of nucleotides) with some empty spaces (no coding regions). In order to get some numerical information from this sequence it must be transformed into a digital sequence [4,13,18]. There follows that the symbolic sequence is transformed into a very large time series (from half million of digits, for the primitive organisms such as fungus, eukaryotes, etc.., to several millions, as for mammals, like the nearly 1.5 billion of nucleotides for the humans DNA). These large sequences look like some random sequence, from where it seems to be quite impossible to single out any single correlation (see e.g. [3,8] and references therein).

More recently, the analysis of DNA sequences was done by using wavelets (see e.g. [1,2,7,9,14,16]). This was motivated by the fundamental properties of wavelets, in fact,

1. with the localization property [7], it is possible (at least in principle) to single out local behavior and to characterize local spikes, jumps [2,5,7].
2. due to the de-correlation process of the wavelet transform [15] the DNA sequence is decomposed into sequences of detail coefficients at different levels, each one expressing some kind of auto correlation on the corresponding level.

The DNA sequence is composed by a large string made by 4 chemical elements (nucleotides) called bases (or base pairs): adenine (A), cytosine (C), guanine (G) and thymine (T). They are combined in a such way to form a long filament which has the structure of double spiral which is a very steady chemical structure.

Some problems in DNA analysis are the understanding of the underlying genomic language, to find an organization principle of the genome, to discover some kind of order (symmetries) or hidden structures (patches or regular patterns) and the existence of functions on genes such as localized periodicities, correlation, complexity, etc. The easiest mathematical model is based on the transformation of the symbolic string into a numerical string based on the Voss indicator function [18] which is a discrete binary function. In the following it is proposed a complex modification of the indicator function, in order to single out a fractal law in the cumulative distribution of nucleotides. The existence of patterns and symmetries is shown through the cluster analysis of the wavelet coefficients [5,6].

2 Complex Indicator Function

Let x_n, be the $n-$th symbolic element of the DNA sequence, with $x_1 = A, x_2 = G, x_3 = T, x_4 = C$, the binary Voss indicator function, (or projection operator) is the function $u_{x_k}, (k = 1, 2, 3, 4)$

$$u_{x_k}(x_m) = \delta_{km} \tag{1}$$

δ_{km} being the Kronecker symbol. There follows that for each symbol we have a 2-values map, so that the original sequence of DNA is transformed into 4 strings of binary values $\{0, 1\}$. However, with the Voss indicator the relative weight of the absence is not detected, therefore a slightly modified definition of the indicator function is proposed as follows

$$u(x_m) = \begin{cases} 1 \ \text{ if } x_m = x_1 = A \\ \\ -1 \ \text{ if } x_m = x_2 = G \\ \\ i \ \text{ if } x_m = x_3 = T \\ \\ -i \ \text{ if } x_m = x_4 = C \end{cases}, \qquad m = 1, ..., N \tag{2}$$

thus obtaining the complex sequence $u_m = \{u(x_m)\}_{m=1,..,N}$ which is the mathematical representation of the genome. The DNA random walk is defined as

$$S_n = \sum_{m=1}^{n} u(x_m), n = 1, ..., N \tag{3}$$

which is the cumulative sum on the indicator function. Since the indicator is a complex function, the random walk is a complex function as well. If we map the points $P_n = (\Re[S_n], \Im[S_n]), n = 1, ..., N$, whose coordinates are the real and the imaginary coefficients of each term of the random walk sequence, we obtain a cluster showing some kind of fractal rule (Figs. 1,2). It should be noticed (Fig.

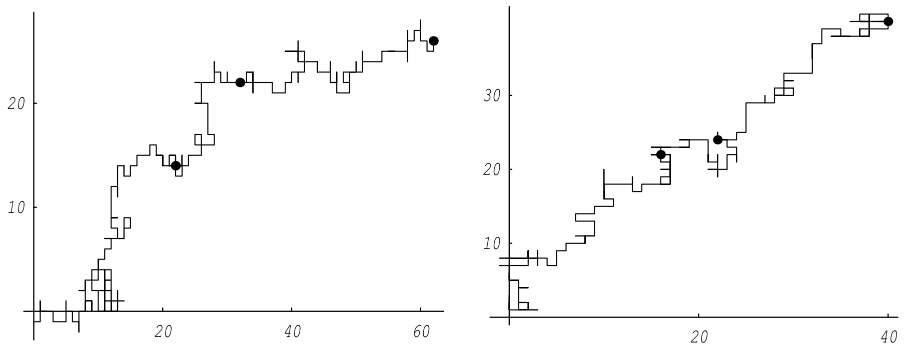


Fig. 1. Random walk ($n \leq 300$) of the Candida's DNA (left) and the dog's (right)

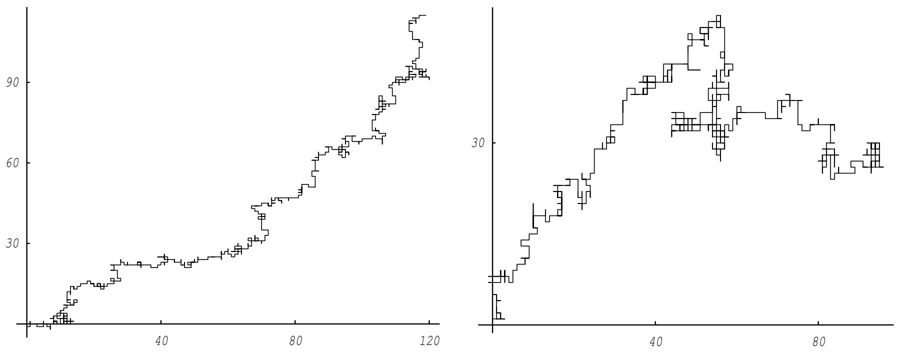


Fig. 2. Random walk ($n \leq 750$) of the candida's DNA (left) and dog's (right) for $n \leq 900$

2) that nearly all points of the random walk lie in the positive sector of the plane so that: $\Re\left[S_n\right] \geq 0, \Im\left[S_n\right] \geq 0,\ (n = n_0, ..., N, n_0 > 1)$.

If we take the absolute value of the random walk, the sequence

$$a_n = |S_n|^2 = \left(\Re\left[S_n\right]\right)^2 + \left(\Im\left[S_n\right]\right)^2, n = 1, ..., N$$

shows that both for the candida DNA and dog's there exists a linear correlation so that both the real and the complex part of the random sequence grow with a linear law as in a long range correlation [3,12,18].

3 Wavelet Analysis of the DNA Random Walk

Let $Y \equiv \{Y_i\}$, $\left(i = 0, ..., 2^M - 1; 2^M = N < \infty, M \in \mathrm{N}\right)$ be a real and square summable sequence sampled at the spots $x_i = i/(N-1)$ and ranging on the regular grid of the dyadic (i.e. $N = 2^M, M \in \mathrm{N}$) points of the interval $[0, 1]$. The discrete Haar wavelet transform is the linear operator $W^N : \mathrm{R}^N \to \mathrm{R}^N$, which associates to a given vector Y the vector of the wavelet coefficients

$$W^N Y = \left\{\alpha, \beta_0^0, \beta_0^1, \dots, \beta_k^{M-1}\right\} \ \left(2^M = N\right) \tag{4}$$

with respect to a given scaling function $\varphi(t)$ and wavelet basis $\psi(t)$.

As wavelet family we choose the Haar wavelets, so that

$$\begin{cases} \varphi_k^n(x) & = 2^{n/2}\varphi\left(2^n x - k\right) \quad , \\ \varphi\left(2^n x - k\right) & = \begin{cases} 1 \ , \ x \in \left[\frac{k}{2^n}, \frac{k+1}{2^n}\right) \\ 0 \ , \ \text{elsewhere} \end{cases} \quad , \end{cases} \tag{5}$$

it is the scaling function with $\varphi(x) = \varphi_0^0(x)$, characteristic function in $[0,1]$, and $\{\psi_k^n(x)\}$ is the wavelet basis

$$\begin{cases} \psi_k^n(x) & = 2^{n/2}\psi\left(2^n x - k\right) \qquad\qquad , \ \left\|\psi_k^n(x)\right\|_{L^2} = 1 \\ \psi\left(2^n x - k\right) & = \begin{cases} -2^{-n/2} \ , \ x \in \left[\frac{k}{2^n}, \frac{k+1/2}{2^n}\right) \\ 2^{-n/2} \quad , \ x \in \left[\frac{k+1/2}{2^n}, \frac{k+1}{2^n}\right) \\ 0 \qquad \text{elsewhere} \ . \end{cases} , \ \left(0 \le n, 0 \le k \le 2^n - 1\right) \end{cases} \tag{6}$$

The wavelet coefficients α, β_k^n , may be used to easily characterized the local variability of data. Let us define the mean value $\langle Y_{i,i+s}\rangle \equiv \sum\limits_{k=i}^{i+s} Y_k/(s+1)$, and, in particular, for $i = 0, s = N-1$ the mean value $\langle Y \rangle = \sum\limits_{k=0}^{N-1} Y_k/N$. It can be easily shown (see e.g. [6]) by a direct computation that,

$$\begin{cases} \alpha \ = \qquad \left\langle Y_{0,2^{M-1}-1}\right\rangle \\ \beta_k^n = 2^{-1+(M-n)/2}\Delta_{2^{M-1-n}}\left\langle Y_{k2^{M-n}, k2^{M-n}+2^{M-n-1}-1}\right\rangle \end{cases}$$

with $n = 0, \dots, M-1, k = 0, \dots, 2^{M-1} - 1$ and

$$\Delta_h Y_i \equiv Y_{i+h} - Y_i, 1 \le h \le 2^M - 1 - i \quad .$$

For example, with $M = 2, N = 4$, we have

$$\begin{cases} \alpha = \dfrac{1}{4}\left(Y_0 + Y_1 + Y_2 + Y_3\right), \ \beta_0^0 = \dfrac{1}{2}\left(Y_2 - Y_0 + Y_3 - Y_1\right) \\ \beta_0^1 = \dfrac{1}{\sqrt{2}}\left(Y_1 - Y_0\right) \quad , \quad \beta_1^1 = \dfrac{1}{\sqrt{2}}\left(Y_3 - Y_2\right) \quad . \end{cases} \tag{7}$$

When the wavelet coefficients are given, the above equations can be solved to obtain the original data. With $M = 2, N = 4$, we have

$$\begin{cases} Y_0 = \alpha - \dfrac{\beta_0^0 + \sqrt{2}\beta_0^1}{2} \ , Y_1 = \alpha - \dfrac{\beta_0^0 - \sqrt{2}\beta_0^1}{2} \\ Y_2 = \alpha + \dfrac{\beta_0^0 - \sqrt{2}\beta_1^1}{2} \ , Y_3 = \alpha + \dfrac{\beta_0^0 + \sqrt{2}\beta_1^1}{2} \end{cases} \quad . \tag{8}$$

4 Short Haar Wavelet Transform

The wavelet transform of a sequence with a huge number of data (like the random walks on DNA) gives a (huge) sequence of detail coefficients which is meaningless when we want to focus on the existence of local (short) or long range correlations. If we are interested on the jumps that can arise from one element of the series and the closer element of the sequence, then we must reduce the number of the elements to be mapped into the wavelet space. This can be achieved by a decomposition of the sequence into short segments of equal length and by a wavelet transform to be applied to each segment.

Given a N-length sequence $Y \equiv \{Y_i\}$ we can decompose it into a set of $\sigma = N/p$ segments with length p

$$\{Y_i\}_{i=0,\dots,N-1} = \{Y_0, Y_1, \dots Y_p\} \oplus$$
$$\oplus \{Y_{p+1}, Y_{p+2}, \dots Y_{2p}\} \oplus \dots\dots \oplus \{Y_{N-1-p}, Y_{N-p+1}, \dots Y_{N-1}\} .$$

Each segment can be transformed into the corresponding segment of the wavelet coefficients. So that, instead to map the whole sequence, the wavelet transform is applied to each segment

$$W^N \{Y_i\}_{i=0,\dots,N-1} = W^p \{Y_0, Y_1, \dots Y_p\} \oplus$$
$$\oplus W^p \{Y_{p+1}, Y_{p+2}, \dots Y_{2p}\} \oplus \dots\dots \oplus W^p \{Y_{N-1-p}, Y_{N-p+1}, \dots Y_{N-1}\} .$$

The resulting transform [5,6] can be considered as a linearization of the wavelet transform in the sense that the wavelet transform of the sequence is taken as a sequence of the wavelet transforms.

For a complex sequence $\{Y_k\}_{k=0,\dots,N-1} = \{x_k + i\, y_k\}_{k=0,\dots,N-1}$ we can consider the correlations (if any) between the wavelet coefficients of the real part $\{x_k\}_{k=0,\dots,N-1}$ against the imaginary coefficients $\{y_k\}_{k=0,\dots,N-1}$. This can be realized by the following cluster algorithm:

$$\begin{array}{l}
\{Y_0, Y_1, \dots Y_p\} \oplus \{Y_{p+1}, Y_{p+2}, \dots Y_{2p}\} \oplus \dots\dots \\
\quad\Downarrow \\
\left.\begin{array}{l}
\{x_0, x_1, \dots x_p\} \oplus \{x_{p+1}, x_{p+2}, \dots x_{2p}\} \oplus \dots\dots \\
\{y_0, y_1, \dots y_p\} \oplus \{y_{p+1}, y_{p+2}, \dots y_{2p}\} \oplus \dots\dots
\end{array}\right\rangle \text{real sequences} \\
\quad\Downarrow \\
\left.\begin{array}{l}
W^p \{x_0, x_1, \dots x_p\} \oplus W^p \{x_{p+1}, x_{p+2}, \dots x_{2p}\} \oplus \dots\dots \\
W^p \{y_0, y_1, \dots y_p\} \oplus W^p \{y_{p+1}, y_{p+2}, \dots y_{2p}\} \oplus \dots\dots
\end{array}\right\rangle \text{wavelet transform} \\
\quad\Downarrow \\
\left.\begin{array}{l}
\{\alpha, \beta^0_0, \beta^1_0, \beta^1_1, \dots\}_1 \oplus \{\alpha, \beta^0_0, \beta^1_0, \beta^1_1, \dots\}_2 \oplus \dots\dots \\
\{\alpha^*, \beta^{*0}_{\ 0}, \beta^{*1}_{\ 0}, \beta^{*1}_{\ 1}, \dots\}_1 \oplus \{\alpha^*, \beta^{*0}_{\ 0}, \beta^{*1}_{\ 0}, \beta^{*1}_{\ 1}, \dots\}_2 \oplus \dots\dots
\end{array}\right\rangle \text{wavelet coefficients} \\
\quad\Downarrow \\
\left.\begin{array}{l}
\{(\alpha, \alpha^*)\}_1 \oplus \{(\alpha, \alpha^*)\}_2 \oplus \{(\alpha, \alpha^*)\}_3 \dots\dots \\
\{(\beta^0_0, \beta^{*0}_{\ 0})\}_1 \oplus \{(\beta^0_0, \beta^{*0}_{\ 0})\}_2 \oplus \{(\beta^0_0, \beta^{*0}_{\ 0})\}_3 \dots\dots \\
\{(\beta^1_0, \beta^{*1}_{\ 0})\}_1 \oplus \{(\beta^1_0, \beta^{*1}_{\ 0})\}_2 \oplus \{(\beta^1_0, \beta^{*1}_{\ 0})\}_3 \dots\dots
\end{array}\right\rangle \text{clusters} \\
\quad\vdots \quad\quad \vdots \quad\quad \vdots
\end{array}$$

Thus we can study the correlation between the real and imaginary coefficients of the random walk. Moreover, by a direct inspection of the random walks (Figs. 2,3) we might be interested not only on the local variations of data but also on the rate of changes. This can be achieved by comparing the sequence of random walk with an artificial sequence which expresses the rate of change thereof. This artificial sequence is the numerical derivative of the random walk and can be defined as follows: Given the $N-$length sequence of real numbers $Y \equiv \{Y_i\}$ we have to compute the interpolating function of the $\{Y_i\}$ either by spline or polynomials or some other differentiable interpolating functions, then we have to compute the derivative of the interpolating function and discretize the derivative with respect to the dyadic points The resulting sequence is taken as the numerical derivative of $\{Y_i\}$.

As for the $\{Y_i\}$, it is useless to map the whole sequence of the derivative, therefore it is expedient to combine the numerical derivative algorithm with the short Haar transform in order to get more information from the cluster analysis. The clusters of wavelet coefficients are obtained as follows. If we call $\{Y'_i\}_{i=0,\dots,N-1}$ the numerical derivative of $\{Y_i\}_{i=0,\dots,N-1}$, the original sequence $\{Y_i\}_{i=0,\dots,N-1}$ is segmented and for each segment is computed the corresponding

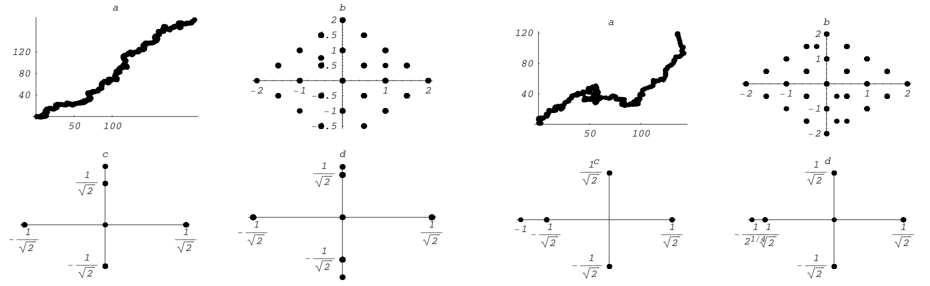


Fig. 3. Cluster analysis of the 4-th short Haar wavelet transform of the random walk ($n \leq 1200$) of the candida (left) and dog's (right) DNA in the planes: a) (α, α^*) ; b) $(\beta^0_0, {\beta^*}^0_0)$; c) $(\beta^1_0, {\beta^*}^1_0)$; d) $(\beta^1_1, {\beta^*}^1_1)$.

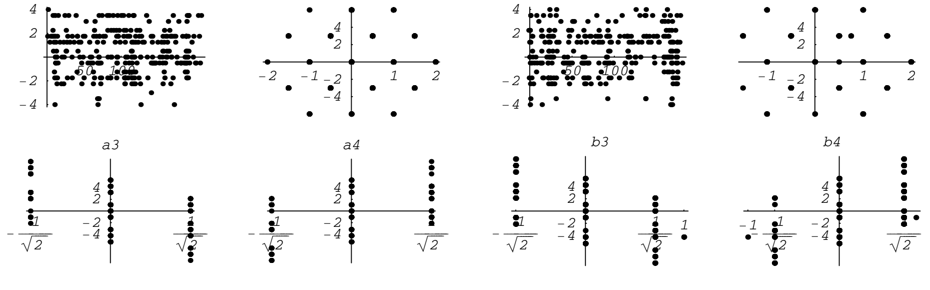


Fig. 4. Cluster analysis of the 4-th wavelet transform of the random walk ($n \leq 1200$) of the candida's DNA for the real part (left) and imaginary (right) in the planes (from top): (α, α') , $\left(\beta^0_0, {\beta'}^0_0\right)$, $\left(\beta^1_0, {\beta'}^1_0\right)$, $\left(\beta^1_1, {\beta'}^1_1\right)$.

numerical derivative. For each segment we have p points where each one belongs to a different 2-dimensional space:

$$\begin{array}{l}
\{(\alpha,\alpha')\}_1 \oplus \{(\alpha,\alpha')\}_2 \oplus \{(\alpha,\alpha')\}_3 \,\ldots.. \oplus \{(\alpha,\alpha')\}_{N/p} \\
\left.\begin{array}{l}
\left\{(\beta^0_0,{\beta'}^0_0)\right\}_1 \oplus \left\{(\beta^0_0,{\beta'}^0_0)\right\}_2 \oplus \left\{(\beta^0_0,{\beta'}^0_0)\right\}_3 \ldots.. \oplus \left\{(\beta^0_0,{\beta'}^0_0)\right\}_{N/p} \\
\left\{(\beta^1_0,{\beta'}^1_0)\right\}_1 \oplus \left\{(\beta^1_0,{\beta'}^1_0)\right\}_2 \oplus \left\{(\beta^1_0,{\beta'}^1_0)\right\}_3 \ldots.. \oplus \left\{(\beta^1_0,{\beta'}^1_0)\right\}_{N/p}
\end{array}\right\rangle \mathit{clusters} \\
\vdots
\end{array}$$

where the prime stands for the coefficients of the derivative sequence $\{Y'_i\}_{i=0,\ldots,N-1}$.

Thus we have as many clusters as many wavelet coefficients in each segment. Since our random walk is a complex series $\{Y_k\}_{k=0,\ldots,N-1} = \{x_k + y_k i\}_{k=0,\ldots,N-1}$ there are at least 4 possible cluster analysis:

1. $(W^p x_k, W^p y_k)_{k=0,\ldots,N-1}$ wavelet coefficients of the real and imaginary part
2. $(W^p x_k, W^p x'_k)_{k=0,\ldots,N-1}$ wavelet coefficients of the real part and its numerical derivative
3. $(W^p y_k, W^p y'_k)_{k=0,\ldots,N-1}$ wavelet coefficients of the imaginary part and its numerical derivative
4. $(W^p x'_k, W^p y'_k)_{k=0,\ldots,N-1}$ wavelet coefficients of the numerical derivative of the real and imaginary part.

In the following we will consider only the first three kind of clusters.

5 Cluster Analysis of the Wavelet Coefficients

For each complex random walk there are 2 sets of wavelet coefficients which correspond to the real and complex coefficient of the complex value. However, even if the real and complex coefficients of the random walk have some nonlinear patterns (Figs. 1,2) the detail coefficients range in some fixed values thus showing the existence of some correlations. It can be seen by a direct computation that the jumps from one value to another belong to some some discrete sets

$$\beta^0_0 \;,\; {\beta^*}^0_0 \in \left\{-2,-\frac{3}{2},-1,-\frac{1}{2},-\frac{1}{4},0,\frac{1}{4},\frac{1}{2},1,\frac{3}{2},2\right\} \;,\; {\beta^*}^1_0\;,{\beta^*}^1_1 \in \left\{-\frac{1}{\sqrt{2}},0,\frac{1}{\sqrt{2}}\right\}$$

$$\beta^1_0 \in \left\{-1,-\frac{1}{\sqrt{2}},0,\frac{1}{\sqrt{2}}\right\} \;,\; \beta^1_1 \in \left\{-\frac{1}{\sqrt[4]{2}},-\frac{1}{\sqrt{2}},0,\frac{1}{\sqrt{2}}\right\}$$

It means that the real and imaginary coefficients of the random walk increases with a given law and the distribution of the nucleotides must follow this rule.

In any case the existence of some patterns is not well shown. From the clusters of wavelet coefficients of the dog's DNA random walk, we can see that while the average values (α,α^*) show a fractal behavior similar to the random walk sequence (Fig. 3) the detail coefficients range instead in some discrete sets.

The existence of symmetries and quantization for the wavelet coefficients of the real and imaginary part of the random walk and its derivatives is also unexpected (Fig. 4).

6 Comparison with a Random Sequence

It should be noticed that the presence of a hidden correlation in the short range for the dog's DNA doesn't depend on the simple (random-like) structure of the complex indicator function. In fact, the first 20 elements of the indicator function for the dog's DNA, according to equation (2) are

$$\begin{array}{c}\{T,\ T,\ C,\ T,\ T,\ C,\ C,\ A,\ A,\ G,\ A,\ G,\ T,\ T,\ C,\ A,\ C,\ A,\ G,\ T, ...\}\\ \Downarrow \\ u\left(x_m\right) \\ \Downarrow \\ \{i,\ i,\ -i,\ i,\ i,\ -i,\ -i,\ 1,\ 1,\ -1,\ 1,\ -1,\ i,\ i,\ -i,\ 1,\ -i,\ 1,\ -1,\ i, ...\}\end{array}$$

So that this sequence looks like a random sequence. Let us compare the clusters of the random walks on DNA with the cluster of a random walk on a random sequence.

For instance if we take the complex pseudo-random sequence

$$r_m = \left(-1\right)^{r_1} i^{r_2}$$

with r_1, r_2, random integer, we get e.g.,

$$\{-1, i, 1, 1, -i, i, -1, -i, i, 1, -i, i, -1, i, 1, -i, -i, -i, -1, ...\} .$$

The corresponding random walk (3) gives rise to a complex sequence whose real and imaginary part are analyzed in the wavelet coefficients planes (see Fig. 5

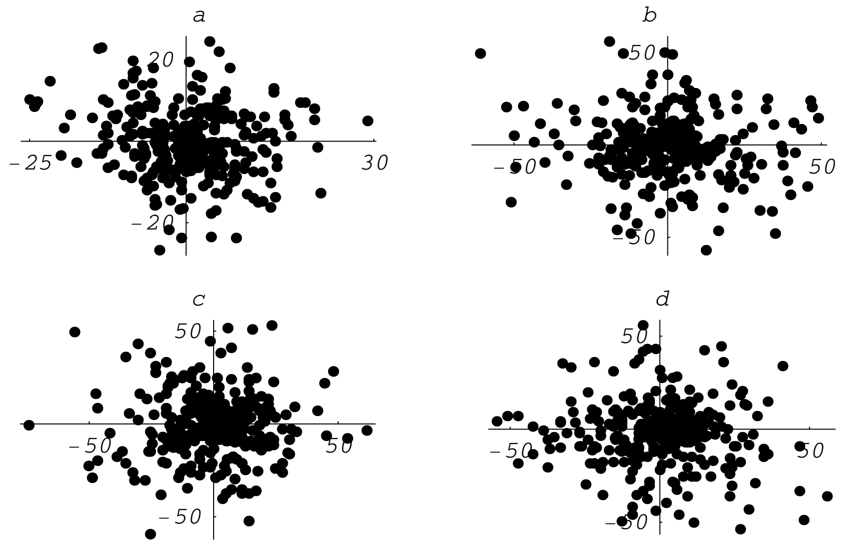


Fig. 5. Cluster analysis of the 4-th short Haar wavelet transform of the random walk ($n \leq 1200$) of a random sequence in the planes: a) $(\alpha, \alpha*)$; b) $\left(\beta^0_0, \beta*^0_0\right)$; c) $\left(\beta^1_0, \beta*^1_0\right)$; d) $\left(\beta^1_1, \beta*^1_1\right)$

to be compared with Fig. 4). We can see that even if the sequence random is similar to the indicator function on a DNA sequence the clusters of the random sequence (Fig. 5) doesn't show the existence of any correlation. While for the random walk on a DNA sequence there are some kind of symmetric distribution of the wavelet coefficients (Fig. 4).

7 Conclusion

In this paper it has been given the definition of a complex indicator function for the DNA sequences. The indicator, applied to the dog's DNA and to the candida'd DNA, has provided a pair of complex strings analyzed with the wavelet transform. By using the wavelet transform together with some algorithms (short Haar transform, numerical derivative and clusters of wavelet coefficients) it has been shown that the random walks have a fractal behavior (with respect to the scaling coefficient α and some (unexpected) symmetries and quantization for the remaining wavelet coefficients $\left(\beta_0^0, \beta_0^1, \beta_1^1\right)$ both for the real coefficients and the imaginary coefficients of the (complex) random walk. A very interesting correlation it has been shown in the comparison of the random walk with its rate of change.

It should be noticed that comparing the clusters of dog and candida there appear some slight changes in the clusters (but not in the quantized values of the wavelet coefficients). This could offer the possibility to organize a different classification of DNA sequences.

References

1. Altaiski, M., Mornev, O., Polozov, R.: Wavelet analysis of DNA sequence. Genetic Analysis 12, 165–168 (1996)
2. Arneado, A., D'Aubenton-Carafa, Y., Audit, B., Bacry, E., Muzy, J.F., Thermes, C.: What can we learn with wavelets about DNA sequences? Physica A 249, 439–448 (1998)
3. Audit, B., Vaillant, C., Arneodo, A., d'Aubenton-Carafa, Y., Thermes, C.: Long range Correlations between DNA Bending Sites: Relation to the Structure and Dynamics of Nucleosomes. JMB, J. Mol. Biol. 316, 903–918 (2002)
4. Berger, J.A., Mitra, S.K., Carli, M., Neri, A.: Visualization and analysis of DNA sequences using DNA walks. Journal of The Franklin Institutes 341, 37–53 (2004)
5. Cattani, C.: Haar Wavelet based Technique for Sharp Jumps Classification. Mathematical Computer Modelling 39, 255–279 (2004)
6. Cattani, C.: Haar wavelets based technique in evolution problems. Proc. Estonian Acad. of Sciences, Phys. Math. 53(1), 45–63 (2004)
7. Cattani, C., Rushchitsky, J.J.: Wavelet and Wave Analysis as applied to Materials with Micro or Nanostructure. Series on Advances in Mathematics for Applied Sciences, p. 74. World Scientific, Singapore (2007)
8. Cristea, P.D.: Large scale features in DNA genomic signals. Signal Processing 83, 871–888 (2003)

9. Dodin, G., Vandergheynst, P., Levoir, P., Cordier, C., Marcourt, L.: Fourier and Wavelet Transform Analysis, a Tool for Visualizing Regular Patterns in DNA Sequences. J. Theor. Biol. 206, 323–326 (2000)
10. Gee, H.: A journey into the genome: what's there. Nature 12 (February 2001), http://www.nature.com/nsu/010215/010215-3.html
11. The Genome Data Base, http://gdbwww.gdb.org/,
Genome Browser, http://genome.ucsc.edu,
European Informatics Institute, http://www.ebl.ac.uk,
Ensembl, http://www.ensembl.org
12. Herzel, H., Trifonov, E.N., Weiss, O., Groe, I.: Interpreting correlations in biosequences. Physica A 249, 449–459 (1998)
13. Li, W.: The study of correlation structures of DNA sequences: a critical review, vol. 21(4), pp. 257–271 (1997)
14. Murray, K.B., Gorse, D., Thornton, J.M.: Wavelet Transform for the characterization and detection of repeating motifs. JMB, J. Mol. Biol. 316, 341–363 (2002)
15. Percival, D.B., Walden, A.T.: Wavelet Methods for Time Series Analysis. Cambridge University Press, Cambridge (2000)
16. Tsonis, A.A., Kumar, P., Elsner, J.B., Tsonis, P.A.: Wavelet Analysis of DNA sequences. Physical Review E 53, 1828–1834 (1996)
17. Vaidyanathan, P.P., Yoon, B.-J.: The role of signal-processing concepts in genomics and proteomics. Journal of The Franklin Institute 341, 111–135 (2004)
18. Voss, R.F.: Evolution of Long-Range Fractal Correlations and 1/f Noise in DNA Base Sequences. Physical Review Letters 68(25), 3805–3808 (1992)

Wavelet Analysis of Impulses in Axon Physiology

Carlo Cattani[1] and Massimo Scalia[2]

[1] DiFarma University of Salerno, Via Ponte Don Melillo I-84084 Fisciano (SA)
[2] Dept. of Mathematics, "G. Castelnuovo", University of Rome, "La Sapienza", P.le A. Moro 2, I-00185 Roma
ccattani@unisa.it, massimo.scalia@uniroma1.it

Abstract. The nonlinear dynamical system which models the axon impulse activity is studied through the analysis of the wavelet coefficients. A system with a pulse source is compared with the corresponding sourceless, through the wavelet coefficients.

Keywords: Haar wavelets, Short Haar Wavelet Transform, Competition model, multidimensional analysis.

1 Introduction

In this paper we consider the nonlinear Fitzhugh-Nagumo system [1,2]

$$\begin{cases} \dfrac{dx}{dt} = \dfrac{1}{\varepsilon}[x(1-x^2)-y], \\ \\ \dfrac{dy}{dt} = x - \beta \ , \end{cases} \tag{1}$$

where ε is a small parameter $0 < \varepsilon \ll 1$ and β is a crucial parameter. This system was proposed in the early 60ties in order to describe the neural activity. Stimulated axons shows their activity by a suddenly change in their electrical potential. These pulse in a short time were called spikes or axons firing. However, the normal activity of neurons is usually describes by a continuous axons firing, even in absence of external stimulations. Therefore one of the main problems is to recognize among all generated spikes those which are caused by some external stimulations. The same system, with an additional wave structure, has been also used to describe solitary wave propagation (spikes or pulses) in a spatial regions in order to modelling neural communication or calcium waves. There follows that small variations in the parameter for signal which have a very short duration gives rise to different physical phenomena. From mathematical point of view system (1) is a nonlinear system which can be derived from the van der Pol by Lienard's change of variables.

This dynamical system is strongly depending [1,2,6,7] on the crucial parameter β in the sense that the evolution would be completely different, starting from a critical time t^*. The dynamics of this system will be studied through the wavelet coefficients of the numerical solution of (1).

Wavelets can capture [5,4] the local changes in a very efficient way. It will be shown that crossing the bifurcation point the wavelet coefficients suddenly change. In particular, if we restrict to the short wavelet transform [3], we have that:

1. For a periodic solution $\beta = 0.57$ all the wavelet coefficients β_0^0 , β_0^1 , β_1^1 both for $x(t)$ and $y(t)$ show a periodic behavior. The amplitude is higher at the lower frequency β_0^0.
2. In correspondence of the bifurcation parameter $\beta = 0.5767$. The amplitude of the wavelet coefficients is the same as in the periodic case only for $t < t^* = 4/5$. The first coefficient β_0^0 is still periodic.
3. When $\beta = 0.6$ all coefficients decay to zero in a short time, thus showing the asymptotic convergence of the solution.

In Sect. 2 some preliminary definitions about Haar wavelets and short Haar wavelet transform [3] are given. The Fitzhugh-Nagumo system is shortly discussed in Sect. 3. The wavelet analysis of Fitzhugh-Nagumo is performed in section 4.

2 Preliminary Remarks on the Short Haar Wavelet Transform

The *Haar scaling function* $\varphi(t)$ is the characteristic function on $[0,1]$. By translation and dilation we get the family of functions defined (in $[0,1]$)

$$\begin{cases} \varphi_k^n(t) \equiv 2^{n/2}\varphi(2^n t - k) \ , \qquad (0 \leq n \ , \ 0 \leq k \leq 2^n - 1) \ , \\ \varphi(2^n t - k) = \begin{cases} 1 \, , t \ \in \Omega_k^n \\ 0 \, , t \ \notin \Omega_k^n \ . \end{cases} \quad , \quad \Omega_k^n \equiv \left[\dfrac{k}{2^n}, \dfrac{k+1}{2^n}\right) \ , \end{cases} \tag{2}$$

The *Haar wavelet* family $\{\psi_k^n(t)\}$ is the orthonormal basis for the $L^2([0,1])$ functions [**?**]:

$$\begin{cases} \psi_k^n(t) \equiv 2^{n/2}\psi(2^n t - k) \ , \qquad ||\psi_k^n(t)||_{L^2} = 1 \ , \\ \psi(2^n t - k) \equiv \begin{cases} -1 \ , \ t \in \left[\dfrac{k}{2^n}, \dfrac{k+1/2}{2^n}\right) \ , \\ 1 \ , \quad t \in \left[\dfrac{k+1/2}{2^n}, \dfrac{k+1}{2^n}\right) \ , \qquad (0 \leq n \ , \ 0 \leq k \leq 2^n - 1) \ , \\ 0 \ , \quad \text{elsewhere} \ . \end{cases} \end{cases} \tag{3}$$

Without loss of generality, we can restrict ourselves to $0 \leq n \ , \ 0 \leq k \leq 2^n - 1 \Longrightarrow \Omega_k^n \subseteq [0,1]$. Let $\boldsymbol{Y} \equiv \{Y_i\}$, $(i = 0, \ldots, 2^M - 1, \ 2^M = N < \infty, \ M \in \mathbb{N})$, be a finite energy time-series; $t_i = i/(2^M - 1)$, is the regular equispaced grid of *dyadic points*.

Let the set $\boldsymbol{Y} = \{Y_i\}$ of N data be segmented into σ segments (in general) of different length. Each segment $\boldsymbol{Y}^s,\ s = 0, \dots, \sigma - 1$ is made of $p_s = 2^{m_s}$, $(\sum_s p_s = N)$, data:

$$\boldsymbol{Y} = \{Y_i\}_{i=0,\dots,N-1} = \bigoplus_{s=0}^{\sigma-1} \{\boldsymbol{Y}^s\}\ , \qquad \boldsymbol{Y}^s \equiv \{Y_{sp_s},\ Y_{sp_s+1}, \dots,\ Y_{sp_s+p_s-1}\}\ ,$$

being, in general, $p_s \neq p_r$. The short discrete Haar wavelet transform of $\boldsymbol{Y}$ is (see [3]) $\mathcal{W}^{p_s,\sigma}\boldsymbol{Y}$,

$$\begin{cases} \mathcal{W}^{p_s,\sigma} \equiv \bigoplus_{s=0}^{\sigma-1} \mathcal{W}_s^p\ ,\ \boldsymbol{Y} = \bigoplus_{s=0}^{\sigma-1} \boldsymbol{Y}^s\ , \\ \mathcal{W}^{p_s,\sigma}\boldsymbol{Y} \qquad = \left(\bigoplus_{s=0}^{\sigma-1} \mathcal{W}^{p_s}\right)\boldsymbol{Y} = \left(\bigoplus_{s=0}^{\sigma-1} \mathcal{W}^{p_s}\boldsymbol{Y}^s\right)\ , \\ \mathcal{W}^{2^{m_s}}\boldsymbol{Y}^s \qquad = \left\{\alpha_0^{0(s)},\ \beta_0^{0(s)},\ \beta_0^{1(s)},\ \beta_1^{1(s)}, \dots,\ \beta_{2^{m_s-1}-1}^{m_s-1(s)}\right\}\ . \end{cases}$$

with $2^{m_s} = p_s$, $\sum_{s=0}^{\sigma-1} p_s = N$. The *discrete Haar wavelet transform* is the operator $\mathcal{W}^N$ which maps the vector $\boldsymbol{Y}$ into the vector of the *wavelet coefficients* $\{\alpha\ ,\ \beta_k^n\}$:

$$\mathcal{W}^N\boldsymbol{Y} = \{\alpha, \beta_0^0, \dots, \beta_{2^{M-1}-1}^{M-1}\}\ , \qquad \boldsymbol{Y} = \{Y_0,\ Y_1,\ \dots,\ Y_{N-1}\}\ . \tag{4}$$

There follows that, the matrix of the wavelet transform is expressed as a direct sum of lower order matrices so that the short transform is a sparse matrix [3]. When the short wavelet transform maps short interval values into a few set of wavelet coefficients, it can be considered as a first order approximation. However, since the wavelet transform maps the original signal into uncorrelated sequences [5], the short wavelet transform describes, for each sequence of detail coefficients, its local behavior. When $p_s = p = N,\ \sigma = 1$, the above coincides with the ordinary wavelet transform. We assume, in the following, $p_s = p = N/\sigma,\ s = 0, \dots, \sigma - 1, (\sigma > 1)$.

3 Fitzhugh-Nagumo System

Let us study the nonlinear model (1) in dependence only on the crucial parameter β. We fix the small parameter $\varepsilon = 0.01$ and we take as initial conditions $x(0) = 0\ , y(0) = 0.39$.

The dynamics of equation (1) has been studied by analyzing the short wavelet transform of the time series obtained by a numerical computation of the solution of (1). By using the Runge-Kutta 4-th order method, with the accuracy 10^{-6}, we obtain in the interval $(0 < t \leq 8)$, four numerical solution in correspondence

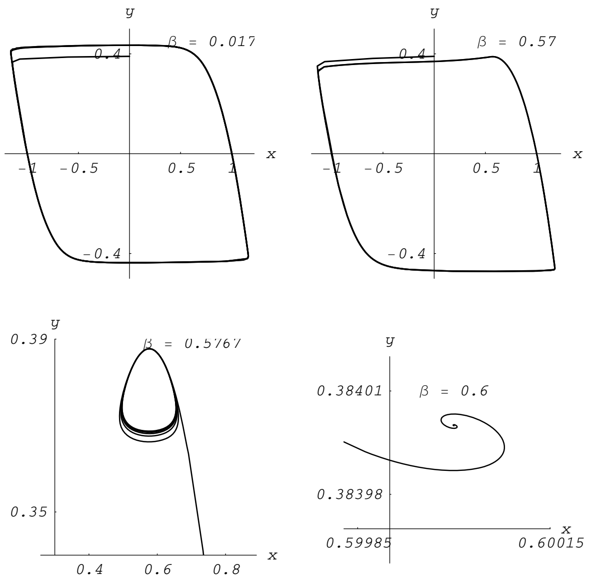


Fig. 1. Numerical solution in the phase space of system (1) with parameters $\varepsilon = 0.01$, and initial conditions $x(0) = 0,\ y(0) = 0.39$, in correspondence of different values of $\beta = 0.017,\ \beta = 0.57,\ \beta = 0.5767,\ \beta = 0.6$

of the values of the parameter $\beta = 0.017\ , \beta = 0.57\ , \beta = 0.5767\ , \beta = 0.6$. These sequences are discretized in $2^9 = 512$ time spots so to obtain 512 values $\boldsymbol{Y} = \{Y_0,\ Y_1, \dots,\ Y_{N-1}\}$, with $N = 512$ and $M = 9$. Moreover, using the short Haar wavelet transform, with $p_s = p = 4$, we compare the wavelet coefficients of three time-series, near the bifurcation value of $\beta = 0.5767$.

In correspondence of 4 different values of β we obtain 4 numerical solution in the phase space as in Fig. 1. It can be seen that

1. when $\beta < 0.5767$, let say $\beta = 0.017\ , \beta = 0.57$ (Fig. 1 top) the orbit is close and the motion is periodic (Fig. 2 top) both for $x(t)$ and $y(t)$. There exist some, periodically distributed, sharp jumps for $x(t)$. These jumps are well shown in the wavelet coefficients pictures (Fig. 3 left).
2. when $\beta = 0.5767$, the dynamical system has a limit cycle (Fig. 1 bottom, left) around the asymptotic limits $x_\infty = 0.6\ , y_\infty = 0.38$ (see also Fig. 2 bottom, left). It can be seen that, after a critical time $t \cong 4$, $x(t)$ has some oscillations around the asymptotic value $x_\infty = 0.6$, while $y(t) \to 0.38$. In this case there remain only some small oscillations of $x(t)$.
3. when $\beta > 0.5767$, let say $\beta = 0.6$, the motion is asymptotically stable (Figs. 1,2 bottom right), in the sense that $x(t) \to 0.6\ ,\ y(t) \to 0.39$ in a finite time $t \cong 4$.

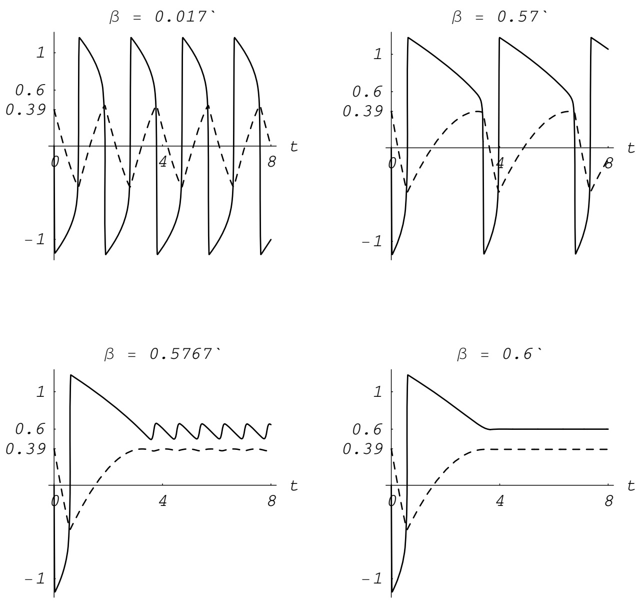


Fig. 2. Numerical solution (plain $x(t)$, dashed $y(t)$) of system (1) with parameters $\varepsilon = 0.01$, and initial conditions $x(0) = 0,\ y(0) = 0.39$, in correspondence of different values of $\beta = 0.017,\ \beta = 0.57,\ \beta = 0.5767,\ \beta = 0.6$

4 Critical Analysis

The qualitative analysis of the previous section can be improved by a further analysis on the scale. It is known [3,5] that the wavelet transform is able to separate the phenomenon into the many scales of approximation. In other words, after transformation, we can observe how is the influence of each scale on the dynamics. In particular, only a small set of detail coefficients (Figs. 3, 4, 5) namely $\beta^0_0,\ \beta^1_0,\ \beta^1_1,$ are able to give a sufficiently good information about the dynamical system, but also to add some information hidden in the previous numerical approach (Figs. 1, 2) in any case better than the numerical evalution.

The detail coefficients show some local maxima and changes which are hidden in the continuous interpolation of the numerical integration. Each detail coefficient performs, at each scale but mostly at the lower scale $n = 0,\ \beta^0_0$, the main feature of the function. Moreover the wavelet coefficients are very sensible to local changes and therefore they can easily describe the intervals where the function is monotonic or when there are some significant jumps. The positive values of the detail coefficients describe the local growth, the negative values the decreasing of the function. Local maxima (minima) of the detail coefficients define some inflexion which enable us to predict if the phenomenon will increase in time or decrease.

In particular, it can be seen that after the critical time $t \cong 4/5$, the absence of jumps in the graphs of β^0_0 (approximately around the time $t \cong 4$) is a sign of

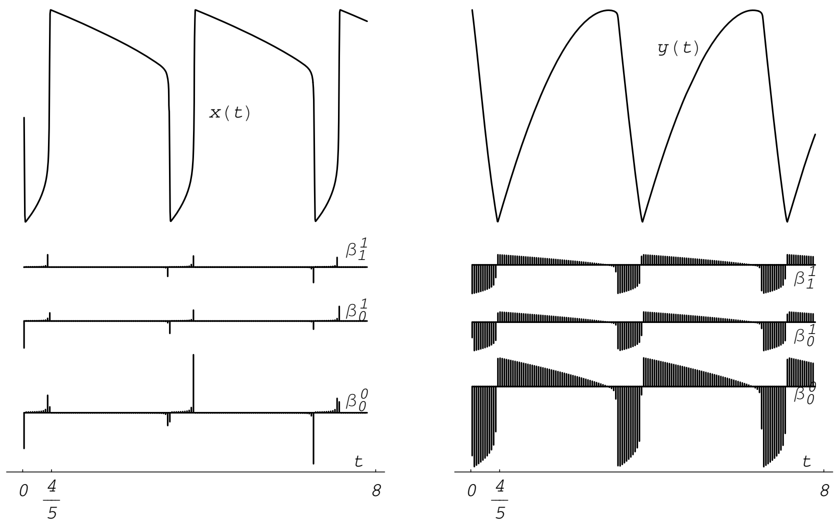


Fig. 3. Numerical solution and wavelet coefficients of 4-parameters of short Haar transform of the numerical solution $x(t)$ (left) and $y(t)$ (right) of system (1) with parameters $\varepsilon = 0.01$, and initial conditions $x(0) = 0,\ y(0) = 0.39$, in correspondence of $\beta = 0.57$

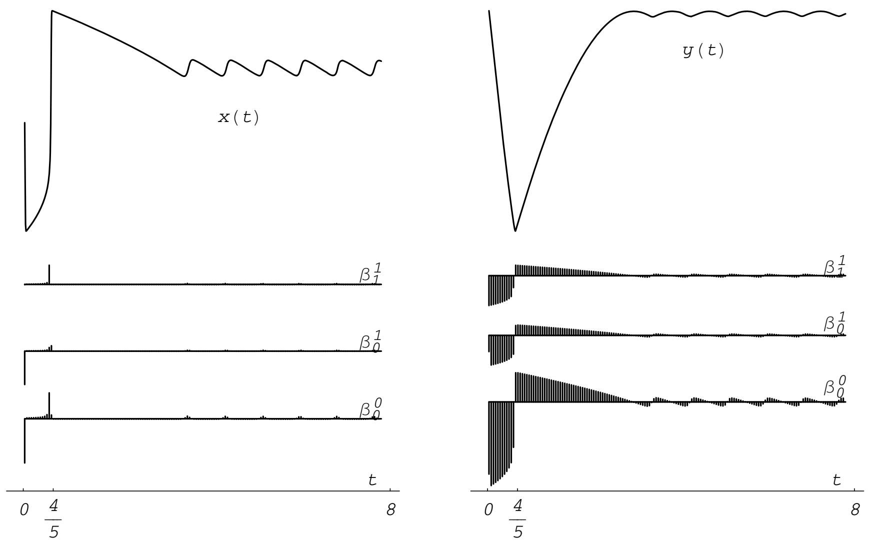


Fig. 4. Numerical solution and wavelet coefficient of 4-parameters of short Haar transform of the numerical solution $x(t)$ (left) and $y(t)$ (right) of system (1) with parameters $\varepsilon = 0.01$, and initial conditions $x(0) = 0,\ y(0) = 0.39$, in correspondence of $\beta = 0.5767$

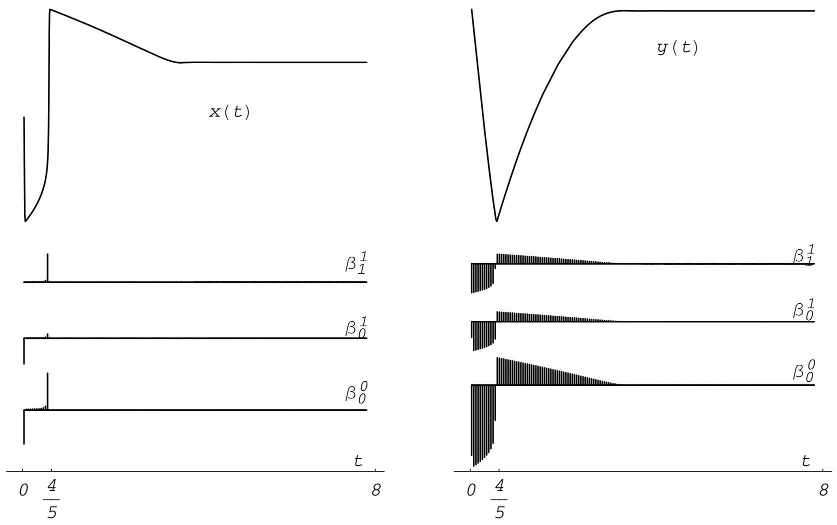


Fig. 5. Numerical solution and wavelet coefficient of 4-parameters of short Haar transform of the numerical solution $x(t)$ (left) and $y(t)$ (right) of system (1) with parameters $\varepsilon = 0.01$, and initial conditions $x(0) = 0,\ y(0) = 0.39$, in correspondence of $\beta = 0.6$

the presence of the bifurcation. In fact, in Figs. 4,5 (left) after $t = 4/5$ there are no changes in β_0^0, β_0^1 ,β_1^1. Just some small periodicity in β_0^0, Fig. 4 (left) shows the existence of small oscillations as expected.

In conclusion, the wavelet method has the following advantages with respect the numerical one, because the qualitative analysis can be performed on a discrete set of points and the wavelet analysis can be well performed by only a few set of wavelet coefficients. In fact the main feature of the system are concentrated on the lower scale coefficients.

References

[1] Fitzhugh, R.: Impulses and physiological states in theoretical models of nerve membrane. Biophys. Journal 1, 445–466 (1961)
[2] Nagumo, J., Arimoto, S., Yoshizawa, S.: An active pulse transmission line simulating nerve axon. Proc. Inst. Radio Eng. 50, 2061–2070 (1962)
[3] Cattani, C.: Haar Wavelet based Technique for Sharp Jumps Classification. Mathematical Computer Modelling 39, 255–279 (2004)
[4] Cattani, C., Rushchitsky, J.J.: Wavelet and Wave Analysis as applied to Materials with Micro or Nanostructure. Series on Advances in Mathematics for Applied Sciences, p. 74. World Scientific, Singapore (2007)
[5] Percival, D.B., Walden, A.T.: Wavelet Methods for Time Series Analysis. Cambridge University Press, Cambridge (2000)

[6] Toma, C.: An Extension of the Notion of Observability at Filtering and Sampling Devices. In: Proceedings of the International Symposium on Signals, Circuits and Systems Iasi SCS 2001, Romania, pp. 233–236 (2001)
[7] Toma, G.: Practical Test Functions Generated by Computer Algorithms. In: Gervasi, O., Gavrilova, M.L., Kumar, V., Laganá, A., Lee, H.P., Mun, Y., Taniar, D., Tan, C.J.K. (eds.) ICCSA 2005. LNCS, vol. 3482, pp. 576–584. Springer, Heidelberg (2005)

Acquisition and Algorithms for Fluorescence Applications

Elena Mocanu and Paul Sterian

University Center for Optical Engineering and Photonics
Bucharest Polytechnic University, 77206 Bucharest, Romania

Abstract. The goal of this study is to present some important updates regarding the photo bleaching field. Fluorescence Recovery after Photobleaching (FRAP) is a versatile technique to determine diffusion coefficients of suitably labeled species in fields like pharmaceutical research, biophysics or macromolecular chemistry. Also we want to emphasize the importance of the numerical simulations especially when we want to extract features of the protein dynamics. The data are processed with Matlab and FEM programs. Of course the simulated situations have to be in a close agreement with the experimental data captured by an Andor Camera, iXON $^+$.

1 Introduction

Fluorescence is an optical phenomenon in cold bodies, in which a molecule absorbs light at a particular wavelength and subsequently emits light of longer wavelength after a brief interval, termed the fluorescence lifetime. The energy difference between the absorbed and emitted photons ends up as molecular vibrations. The different timescales during the emission-absorption cycle play a crucial role for the fluorescence process. The Frank-Condon principle states that electronic transitions take place in times that are very short compared to the time required for the nuclei to move significantly.

2 General Considerations

Photobleaching occurs when a fluorophore permanently loses the ability to fluoresce. Each fluorophore has different photobleaching-characteristics. Its stability can be characterized by the average number of absorption-emission cycles that the molecules of this fluorophore undergo before they are irreversibly photobleached. The number of cycles depends on the local environment and the molecular structure and is, therefore, a fluorophore-specific property. The exact mechanism of photobleaching is yet not known, but it is assumed to be linked to a transition from the excited singlet state to the excited triplet state, a process called intersystem crossing. This transition withdraws the molecule from the absorption-emission cycle and as the triplet state is relatively long-lived with respect to the singlet state it is chemically more reactive. Thus excited molecules have a much longer timeframe to undergo further chemical reactions with components in the environment that are the basis for bleaching reactions.

Usually the systems can be described in terms of a system of partial differential equations (PDE)[9], [5] and to develop the geometry is very useful to use a very popular procedure of finite element methods. By doing this we can define geometries and the initial conditions.

The in-vivo fluorescent images have offered an important new insight into nuclear architecture and function. The photo bleaching process is very important because it can exhibit the biophysical properties of nuclear proteins in intact cells, especially the dynamic part.

The diffusion process stays at the base of the protein movement; we have to bear in mind that the high mobility is a general feature of nuclear proteins[1],[7],[6]. The dynamics of a transcription factor, mainly the glucocorticoid receptor (GR) were investigated with photo bleaching method.

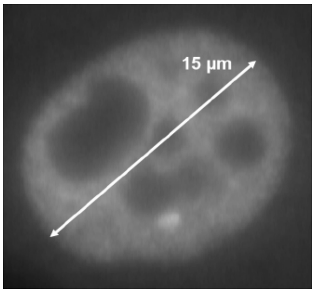


Fig. 1. Glucocorticoid Receptor marked with GFP expressed in a cell nucleus

The excited state has a sufficient long lifetime so that the molecules can achieve a thermally equilibrated lower energy excited state by converting the excess vibrational energy to heat and exchanging it with the environment. The Kosha's rule claims that the fluorescent emission will generally occur from the lowest excited singlet state. Respecting the Stokes shift the energy of the emitted photon will be less than that of the exciting photon and the wavelength is shifted to longer values.

The simplest way to form an image is to use the super positioning algorithm. So we can consider to have an infinite number of point sources, blurred individually. As already known a real optical system produces a small, blurred spot known as Airy disk.

In conclusion the response of the optical system in mathematically terms a convolution process that consists of the object intensity and the point spread functions PSF (x,y).

I(x,y)=

$$PSF(x,y)\otimes O(x,y)=\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty}PSF(x-x',y-y')O(x',y')dx'dy' . \quad (1)$$

$$PSF(x,y)=\delta(x)\delta(y) . \quad (2)$$

The object intensity is directly mapped to the image intensity. The resolution of such a system depends upon NA and the wavelength. Very important for these cases is also the Rayleigh criterion for lateral resolution that is given by the relation:

$$R_{lat} = \frac{0.4\lambda}{NA}. \tag{3}$$

The axial resolution for a confocal configuration that is used in this case:

$$R_{axial} = \frac{1.4n}{NA^2} \tag{4}$$

We have the illumination intensity profile given by the laser beam so we have to deal with a Gaussian profile;

$$I(r,z) = I_0 e^{-2(\frac{r^2}{w^2}+\frac{z^2}{z_0^2})}. \tag{5}$$

Bearing in mind the above considerations we can understand why in FRAP experiments is very common to consider lens of relatively low NA with an almost cylindrical illumination profile.

The fluorescent recovery after photo bleaching is a method that makes the measurement of molecular dynamics possible; it consists of introducing a rapid swift away from the stady-state distribution without disrupting the actual concentration of the molecule under study. This can be achieved by applying a high-powered focused laser beam for a very short time. This phenomenon will determine a photo bleaching process inside the region of interest. After a short period of time the fluorescent molecules will occupy the remaining place and we can study the dynamics. The major process that is very interesting, including for the mathematical simulations is represented by the FRAP recovery curve.

During the photo bleaching process, a significant fraction of the fluorescent protein is made non-fluorescent. This reduces the theoretical maximum intensity that the photo bleached region may recover to from its initial intensity after bleaching. We

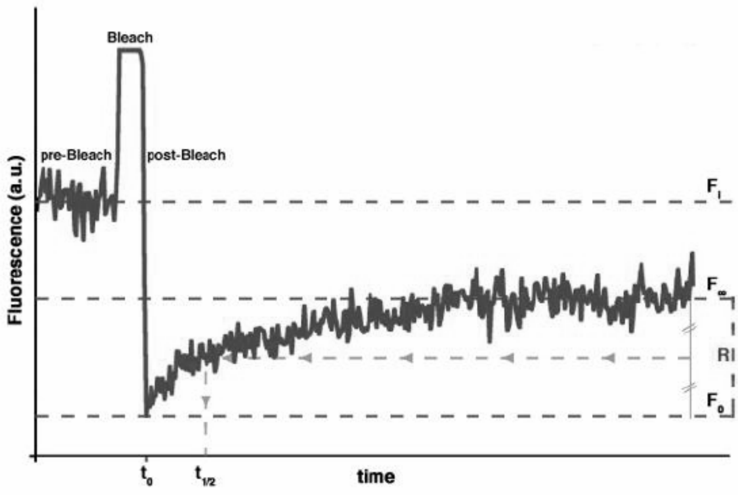


Fig. 2. FRAP recovery curve

have to take into consideration the fact that during the acquisition the cell is also exposed to many iterations of laser illumination and that may be traduced by the fact that a small percent can reach the photo bleaching threshold. Using the experimental data we can approximate that is about 10% because of the photo bleaching and 5% during the acquisition as can be seen in Figure 2. The main and the most useful method of normalization is the Phair and Misteli.

If we consider the F_∞ the assimptote, $\tau_{1/2}$ the time of diffusion that contains information about the protein mobility, T_0 - total intensity of the fluorescence, I_0 the total intensity of the signal before starting the photo bleaching.

$$\mathrm{F(t)}=\frac{I(t)T_0}{I_0 T(t)} \tag{6}$$

And the mobile fraction $R = \dfrac{F_\infty - F_0}{F_1 - F_0}$ (7)

$\tau_{1/2}$ as being the half between the F_0 and F_∞.

3 Mathematical Model

We will use the Axelrod model which is valid for circular bleach spots generated by a stationary Gaussian laser beam [3]. In this case we consider $D = \dfrac{\omega^2}{4\tau_{1/2}}$, ω is half width of the laser. In this way we can obtain information about binding affinity.

For a better mathematical simulation we presume the structure is immobile on the time scale of the FRAP experiment and spatially homogeneously distributed.

The diffusion process is very well described by the Fick's lows [6]:

$$\begin{gathered}\vec{J} = -D\nabla c\\ \frac{\partial c}{\partial t} = D\nabla^2 c\end{gathered} \tag{8}$$

We have also to take into consideration that the expression of D is given by the Stokes-Einstein equation:

$$D = \frac{kT}{6\pi\eta R_n} \tag{9}$$

where R_n represents the hydrodynamic radius of the particle.

One assumption that has to be mention is the fact that T is considered constant during the experiment. The diffusion coefficients are depending strongly on the environment medium e.g. GFP as described below in Figure 3.

Table 1. With coefficients of diffusion

Medium	D[μ m^{2}s^{-1}]
H_2O	87
Cytoplasm	25
Togged glucocorticoid	9.2

As can be seen above it is a direct relation between the diffusion coefficient, mass and the hydrodynamic radius:

$$D \propto R^{-1} \propto M^{-1/3} \tag{10}$$

There is one important thing that we have to consider to understand the phenomena, when a marker is added we don't modify in any way the concentration of the proteins. FRAP only induces a concentration smaller or higher for the marker agent.

The kinetics of the reactions is composes from many classes; we can consider as possible one single binding or multiple bindings. We will consider only the simplest possibility and this is the one binding. This implies of course also one second order reversible reaction. We can describe mathematically the process as:

$$F + S \underset{k_{off}}{\overset{k_{on}}{\Leftrightarrow}} C, \tag{11}$$

where F represents the proteins, S the vacant binding sites and C – the bound complex; k_{on} and k_{off} are two parameters that measures the affinity of the proteins to form new structures. As usually

$$\tau_{1/2} = \tau_d = \frac{1}{k_{on}} \tag{12}$$

and the time of binding is

$$\tau_b = \frac{1}{k_{off}} \tag{13}$$

The presumptions are that for a k_{off} higher we are dealing with a protein that releases quickly after binding while for a k_{on} big we realize that the protein has a very short diffusion time; it forms very easy a bound complex.

To describe a complete system we shall write the equations:

$$\begin{aligned} \frac{\partial f}{\partial t} &= -k_{on} f_s + k_{off} c \\ \frac{\partial s}{\partial t} &= -k_{on} f_s + k_{off} c \\ \frac{\partial c}{\partial t} &= k_{on} f_s - k_{off} c \end{aligned} \tag{14}$$

In same cases is very useful to consider the diffusion of the bound complex and that of the vacant binding as being 0; also although bleaching changes the number of visible free and bound molecules, that meaning P and C it does not change the number of free binding sites so s is constant during the photo bleaching process. In this case our system becomes:

$$\begin{aligned} & k_{on}s = k_{on}S_{eq} \Rightarrow k_{on}^{*} \\ & \frac{\partial f}{\partial t} = D_f \nabla^2 f - k_{on}^{*} f + k_{off} c \\ & \frac{\partial c}{\partial t} = k_{on}^{*} f - k_{off} c \end{aligned} \tag{15}$$

If the system is in equilibrium then there are no changes of the concentrations with respect of time so we can write the system:

$$\begin{aligned} & \frac{\partial f}{\partial t} = \frac{\partial c}{\partial t} = 0 \\ & k_{on}^{*} F_{eq} = k_{off} C_{eq} \\ & C_{eq} = F_{eq} \frac{k_{on}^{*}}{k_{off}} \end{aligned} \tag{16}$$

In the case of more binding states model we may consider a super positioning of more one-binding states.The next step we should consider is the writing of the boundary conditions; the initial values are defined by the photo bleaching process; mainly we have Newmann conditions because during the time scale of the experiment there is no flux of fluorescent biomolecules into or out of the nucleus and the membrane acts as a diffusion barrier.

In order to compare the results from a simulation to the results from an experiment it is necessary to transform the calculated distribution of the fluorophore to a fluorescence intensity image as seen by the microscope.

We have also a very important space independent relation between the concentration of fluorescent molecules and the fluorescent intensity.

If we have $F(t)$ at $t \geq 0$ and a stationary laser beam with a nanoscanning microscope then

$$F(t) = q \iint_{\Omega} I_d(x,t)(f(x,y,t) + c(x,y,t))dxdy \tag{17}$$

over the bleached spot Ω. q is a constant factor that takes all relevant factors concerning illumination and light collection into account; by $f(x,y,t)$ we understand the concentration of the free proteins and by $c(x,y,t)$ the concentration of bound proteins.

As mention before we will consider samples with z<<x and z<<y so we can have 2D geometries. The image formation can be mathematically be described by the convolution of each point and its spread function [9].

If the size of the bleach spot is considered small relative to the size of the fluorescent area then the nucleus can be considered as infinite sized. The fluorescence intensity recovers to the initial pre-bleach value so some measurable fraction of the fluorescence will be lost during the bleaching. The infinite sized nucleus yields the boundary condition that

$$c(r \pm \infty, t) = C_{eq} \tag{18}$$

for all times.

The analytical solution is calculated by the application of the Laplace transformation. This leads to the new set of differential equations, a Bessel set. The mathematical expressions of the FRAP process becomes:

$$\overline{FRAP}(p) = \frac{1}{p} - \frac{F_{eq}}{p}(1 - 2k_1(qw)I_1(qw) \times (1 + \frac{k_{on}^*}{p + k_{off}}) - \frac{C_{eq}}{p + k_{off}} \tag{19}$$

$$q^2 = \left(\frac{p}{D_f}\right)(1 + \frac{k_{on}^*}{p + k_{off}} \tag{20}$$

w- radius of the bleach spot
I_1, k_1 are the modified Bessel functions of the first and second kind.
p- Laplace variable

If we take into consideration the normalization then the fluorescence intensity belongs to the interval [0,1] and we may write the system:

$$\begin{aligned} &F_{eq} + C_{eq} = 1 \\ &F_{eq} = \frac{k_{off}}{k_{on}^* + k_{off}} \\ &C_{eq} = \frac{k_{on}^*}{k_{on}^* + k_{off}} \end{aligned} \tag{21}$$

There are many different scenarios for one binding model:

- Timescal with $$\begin{aligned} \tau_{1/2} &= w^2 / D_f \\ \tau_d &= 1/k_{on} \end{aligned} \tag{22}$$
- Pure diffusion domninant, where binding has no influence

$$\frac{C_{eq}}{F_{eq}} \le 0.01 \Rightarrow \frac{k_{on}^*}{k_{off}} \le 0.01 \tag{23}$$

- Effective diffusion $\tau_d << \tau_{1/2}$

$D_{eff} = \dfrac{D_f}{1+(k_{on}^{*}/k_{off})}$, we can talk in this case about the slower movement of the proteins due to the binding events.

- Reaction dominant meaning that the diffusion is very fast compared to both the binding reactions and the timescale of the FRAP experiment:

$$\tau_b = \frac{1}{k_{off}}$$
$$\frac{F_{eq}}{C_{eq}} = \frac{k_{off}}{k_{on}^{*}} \le 1 \tag{24}$$

So the behavior of the system is mainly determined by the binding rates.

If we presume the particle is supposed to diffuse very rapidly between two traps then we can use a standard Levy law time [2], [4]-from this point the time τ the particle stays in a trap is supposed to have very strong fluctuations which can give rise to anomalous diffusion pattern.

The mathematical expression that describe the Levy law time is:

$$P_0(\tau) = \frac{\alpha}{(1+\tau)^{\alpha+1}} \tag{25}$$

α being correlated with the sub diffusive behavior. For long times $< r^2(t) > \propto t^{\alpha}$ and for $\alpha < 1$ we can talk about a Green function:

$$\vec{g}(k,w) = \frac{1}{w(D_\alpha k^2 w^{-\alpha} + 1)} \tag{26}$$

Where $D_\alpha = D/\Gamma(1-\alpha)$ (27)

Where w and k are the conjugated variables of position r and time t, where $k = |k|$.

We make the change $x = r^2/t^{\alpha}$ and for a higher x we can approximate the inverse transformation via a saddle point method.

The general solution of this type of anomalous diffusion process is then:

$$\rho(r,t) = \int \rho_0(r'-r)g(x(r',t))d^2r' \tag{28}$$

$$g(r,t) \propto \exp(-cstx') \tag{29}$$

ρ represents the probability density to find the particle at the point r at instant t and ρ_0 in its initial state. As the green function is a bell-shaped fast decreasing function, one approximates it by a gaussian shape with the exact dispersion,

$D_\alpha = D\sin(\pi\alpha)/(\pi\alpha)$ (30) which can be calculated. This permits to construct an analytical expression of the fluorescence recovery using standard properties of Gaussian functions. Starting from Axelrod [3] initial density as it is immediately after a Gaussian laser beam profile extinction indeed:

$$\rho_0(r) = \exp(-K\exp(-2\frac{r^2}{R^2})) \tag{31}$$

Where K –photobleaching constant, depending on experimental conditions and using the standard properties of the Gaussian shape in the convolution operation, one can obtain the FRAP signal:

$$I_R(t) = 1 + \sum_1^\infty \frac{(-K)^n}{n!}\frac{1}{2n}(1 - \exp(-\frac{2nR^2}{R^2 + 4nD_\alpha t^\alpha})) \tag{32}$$

The function can be used to fit the experimental data and to compare the experimental with the Monte Carlo simulations.

Nevertheless we intend to use the Virtual Cell program to verify one more time the data as seen in figure 3.

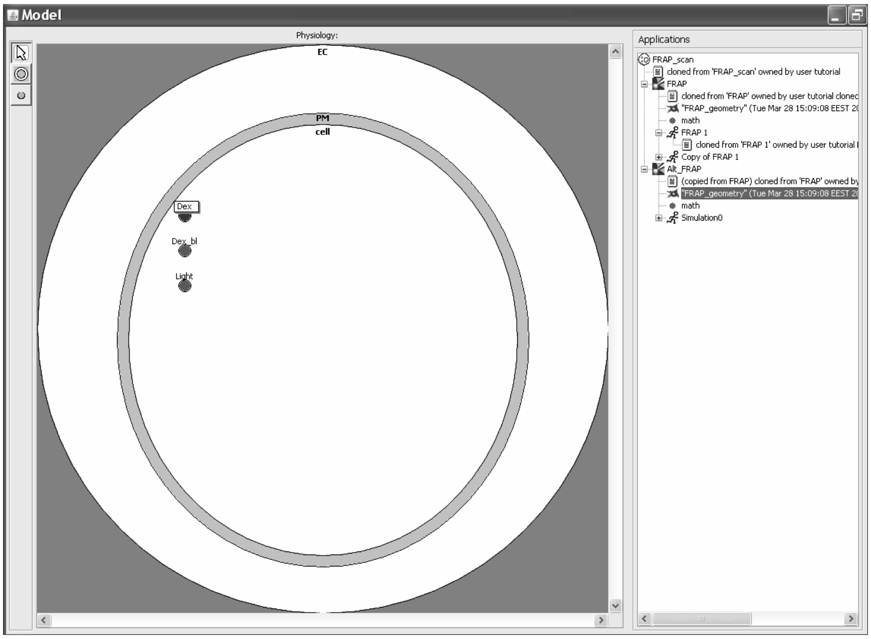

Model
Physiology:
EC
PM
cell
Dex
Dex_bl
Light
Applications
FRAP_scan
cloned from 'FRAP_scan' owned by user tutorial
FRAP
cloned from 'FRAP' owned by user tutorial clonec
"FRAP_geometry" (Tue Mar 28 15:09:08 EEST 2(
math
FRAP 1
cloned from 'FRAP 1' owned by user tutorial
Copy of FRAP 1
Alt_FRAP
(copied from FRAP) cloned from 'FRAP' owned by
"FRAP_geometry" (Tue Mar 28 15:09:08 EEST 2(
math
Simulation0

Fig. 3 Simulation in VCell regarding the FRAP process

4 Conclusions

The development of numerical simulation and the use of virtual free programs can accomplish a very important role regarding the formation of new scientist; also implies few financial results and may open new paths of research.

Matlab and FEM programs have proven an extraordinary ability to simulate so complex phenomena as the interaction of laser with biological molecules; more than the real set-up these simulation programs allows a very large domain of boundary conditions, species and processes. A more physiological version dedicated to the bio-imaging and biophotonics is represented by V Cell a very simple program that simulates the labcell. It is very helpful also that the researchers and the tutorials are made public so you can start an adventure in this wonderful field: the living.

The importance of the kinetics and the dynamics of different biological species is extremely important considering only the medical applications, eg. was observed that the main characteristic that distinguishes anomalous from normal diffusion is the behavior of the mean squared displacement (MSD) as a function of time. For normal behavior the MSD grows linearly with time while for an anomalous process it grows sublinearly.

The results we want to test covers especially the diffusion of proteins in cells where fatty acids are present and how their amount can affect the FRAP process.

Acknowledgement

The authors would like to thank Andor Technology representing in Romania- Apel Laser, The University Center for Optical Engineering and Photonics from “Polytehnica” University, University of Connecticut Health Center.

References

1. Favard, C., Olivi-Tran, N., Meunier, J.-L.: Membrane bound protein diffusion viewed by fluorescence recovery after bleaching experiments: models analysis,arXiv:cond-mat/0210703 v1, October 31 (2002)
2. Lubelski, A., Klafter, J.: Lateral Diffusion of Proteins in Cell Membrane: The Anomalous Case. The Open-Access Journal for the Basic Principles of Diffusion Theory, Experiment and Application
3. Axelrod, D., Koppel, D.E., Schlessinger, J., Elson, E., Webb, W.W.: Mobility measurement by analysis of fluorescence photobleaching recovery kinetics. Biophysical Journal 16 (1976)
4. Metzler, R., Klafter, J.: The random walk’s guide to anomalous diffusion: a fractional dynamics approach, Physics reports vol. 339 (December 2000)
5. Sheets, E.D., Simson, R., Jacobson, K.: New insights into membrane dynamics from the analysis of cell surface interactions by physical methods. Curr. Opin. Cell Bio. 7, 707–714 (1995)
6. Jacobson, K., Sheets, E.D., Simson, R.: Revisiting the fluid mosaic model of Membranes. Science 268, 1441–1442 (1995)
7. Edidin, M.: Fluorescence photobleaching and recovery, FPR, in the analysis of membrane structure and dynamics. In: Damjanovish, S., Edidin, M., Szollosi, J., Tron, L. (eds.) Mobility and Proximity in Biological Membranes, pp. 109–135. CRC Press, Boca Raton (1994)
8. Kennworthy, A.: FRAP Approaches to Studying Lipid Rafts Speaker Paper 14 – Saturday
9. Müller, M., Charypar, D., Gross, M.: Particle-Based Fluid Simulation for Interactive Applications

Control of the Lasers Dynamics by Pumping Modulation for Biophotonics Applications

Paul Sterian, Octavian Marin, and Valerică Ninulescu

Universitary Center of Optical Engineering and Photonics,
Faculty of Applied Sciences, “Politehnica”University of Bucharest,
Splaiul Independenţei 313, Bucharest, 060042, Romania

Abstract. A study of the opportunities offered by the fiber laser output for optical coherent stimulation is performed. The investigation is based on the strong nonlinear behavior of the pump modulated laser. The two-mode laser enhances the variability range of the output including the state of polarization of the laser field. This is expected to have applications in neural stimulation.

Keywords: Optical fiber laser, nonlinear dynamics, optical stimulation.

1 Introduction

Optical fiber lasers were invented in 1963 by Elias Snitzer [1,2] and they were attractive for the large gain [3] and the possibility of enlarging the number of laser wavelengths emitted in the continuous-wave regime. These lasers used single-mode laser diode pumping and delivered tens of milliwats output. In 1990, the first watt-level (4 W) erbium-doped fiber laser output was reported and this became the starting point for a rapid progress in the fabrication of fiber lasers. The years to come are expected to bring a larger growth rate for fiber lasers in comparison to other types of lasers. Single-mode fiber lasers with power up to a few kW and multi-mode fiber lasers of a few tens of kW are now available. Besides the industrial and telecommunication applications, fiber lasers have become important in medicine [4,5,6,7], for microsurgery, optical coherence tomography, or skin resurfacing.

Fiber laser technology offers now several benefits to the user which will determine an increasingly replace of other already used lasers in medical applications. A fiber laser has a high electrical efficiency, there is no requirement for chilling, it is maintenance free during the entire lifetime (no need of flashlamps or diodes replacement), it is compact and the system complexity is reduced. A laser beam furnished by a fiber laser can reach a diameter of a few micrometers which is advantageous in microsurgery for the quality of the cuts, a faster incision, and a faster excision.

The paper deals with the the erbium-doped fiber laser that operates at 1.55 μm. For medical applications this wavelength is important due to the existence of a water absorption peak near his wavelength, at 1.44 μm, while the most important two peaks are at 2.94 μm and 1.94 μm, respectively [8]. For the low level laser terapy, the single-mode and collimated laser beam at this wavelength can create a precise skin resurfacing in a nonablative way [5] elliminating the disadvantages of the CO_2 laser use. In neural stimulation [9,10], the fiber laser is a candidate for its infrared coherent radiation.

For improved laser beam characteristics or versatility in their choice we propose pumping modulation of the laser, thus taking advantage of the nonlinear laser dynamics. The modulation is easily performed through direct modulation of the injected current in the semiconductor laser that pump the fiber laser. A continuous or discontinuous change of the laser output parameters like pulse repetition rate, pulse height and shape, sequences of various pulses can be easily obtained through a change in the modulation depth or frequency of modulation.

Despite the multimode operation, a clustering effect makes it possible to treat a fiber laser dynamics in terms of one or two laser modes [11,12,13,14]. The laser models are based on the rate equation approximation and emphasize the nonlinear dynamics even near the laser threshold.

2 The Single-Mode Laser Model

The erbium-doped laser emitting around 1.55 μm ($^4I_{13/2} \rightarrow {}^4I_{15/2}$ transition) is a three-level system [3] (Fig. 1). Er^{3+}-ions at a concentration N_0 are pumped from level 1 ($^4I_{15/2}$) to level 3 (for example $^4I_{11/2}$, 980 nm above the ground state); we denote Λ the probability of an ion in state 1 to be pumped in unit time to state 2. The level 3 is fastly depopulated through a non-radiative transition on the upper laser level 2 ($^4I_{13/2}$) which is metastable with the lifetime $\tau_2 = 10\,\mathrm{ms}$ [11]. The letter σ in Fig. 1 denotes the absorption cross section in the laser transition. In the rate equation approximation, the laser dynamics by two coupled differential equations, one for the population inversion and the other for the laser intensity:

$$\frac{\mathrm{d}n}{\mathrm{d}t} = 2\Lambda - \frac{1}{\tau_2}(1+n) - 2\sigma n I, \tag{1}$$

$$\frac{\mathrm{d}I}{\mathrm{d}t} = -\frac{1}{\tau} I + \sigma N_0 n I. \tag{2}$$

The dependent variable n is the difference of the occupation probability of level 1 and 2, respectively, and I is the photon density inside the laser cavity. The photon lifetime in the laser cavity is τ. Note that parameter σ in the above equations is the absorption cross-section times light speed.

Laser action takes place for $\Lambda > \Lambda_{\mathrm{th}}$, where the threshold pumping parameter is

$$\Lambda_{\mathrm{th}} = \frac{1}{2\tau_2}\left(1 + \frac{1}{\sigma N_0 \tau}\right), \tag{3}$$

and the laser intensity in the stationary state increases linearly with the pumping strength above threshold:

$$\bar{I} = N_0 \tau (\Lambda - \Lambda_{\mathrm{th}}). \tag{4}$$

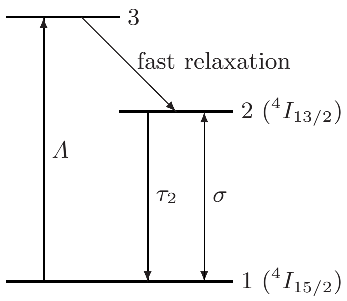


Fig. 1. Er^{3+}-ion energy levels implied in the laser effect

For modulation purposes, the linear responce of the laser to the input pumping is analyzed. The transfer function for the frequency f defined as [15]

$$H(f) = \frac{\delta I}{\delta \Lambda} \tag{5}$$

shows a sharp peak (Fig. 2) for the resonance frequency

$$f_{\mathrm{r}} = \frac{1}{2\pi}\sqrt{\frac{2\sigma}{\tau}\bar{I} - \frac{1}{2}\left(\frac{1}{\tau_2} + 2\sigma\bar{I}\right)^2} \approx \frac{1}{2\pi}\sqrt{\frac{2\sigma}{\tau}\bar{I}} = \frac{1}{2\pi}\sqrt{2\sigma N_0(\Lambda - \Lambda_{\mathrm{th}})}, \tag{6}$$

which is some tens of kilohertz.

The efficient pump modulation is performed for frequencies around the value f_{r}. We consider the parameters used in Fig. 2, a relatively low pumping strength factor $\Lambda/\Lambda_{\mathrm{th}} = 1.5$ and investigate the laser output for a modulation frequency $f = 0.85 f_{\mathrm{r}}$. Some temporal dynamics are shown in Fig. 3. We notice that the linear behavior of the laser considered in Fig. 2 is valid only for small modulation depths [Fig. 2(b)]; the nonlinearity is first responsible for creation of short pulses [Fig. 2(c)] and then for bifurcations with period doubling [Fig. 2(d,e)] and finally chaotic pulses are reached [Fig. 2(f)]. All such dynamics can be obtained by sweeping an electrical physical quantity.

3 The Two-Mode Laser Model

The starting point for the extension of the single-mode model of the fiber laser [Eqs. (1,2)] to a two-mode model is the addition of the other state of polarization for the laser field. Under the same conditions, laser equations write now

$$\frac{\mathrm{d}n_1}{\mathrm{d}t} = 2\Lambda - \frac{1}{\tau_2}(1+n_1) - 2\sigma n_1(I_1 + \beta I_2), \tag{7}$$

$$\frac{\mathrm{d}n_2}{\mathrm{d}t} = 2\gamma\Lambda - \frac{1}{\tau_2}(1+n_2) - 2\sigma n_2(\beta I_1 + I_2), \tag{8}$$

$$\frac{\mathrm{d}I_1}{\mathrm{d}t} = -\frac{1}{\tau}I_1 + \sigma N_0(n_1 + \beta n_2)I_1, \tag{9}$$

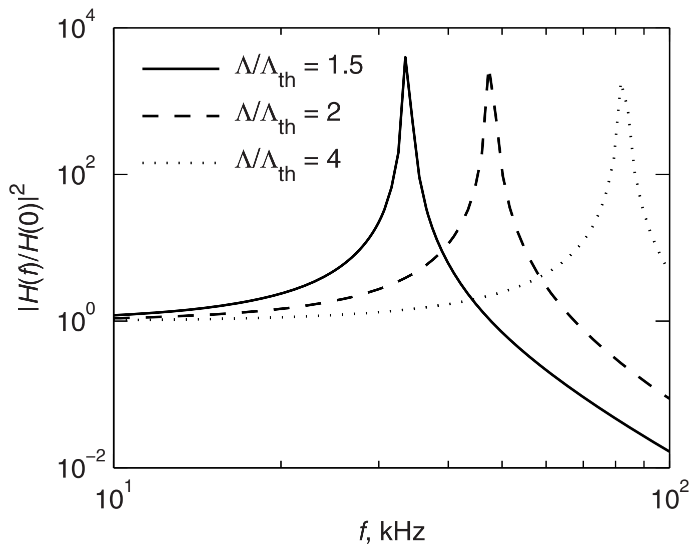


Fig. 2. Transfer function of the fiber laser described by Eqs. (1) and (2) and its dependence on pumping strength. $N_0 = 5 \times 10^{24}\,\mathrm{m}^{-3}$, $\sigma = 1.6 \times 10^{-16}\,\mathrm{m}^3\mathrm{s}^{-1}$ and $\tau = 10\,\mathrm{ns}$ [11].

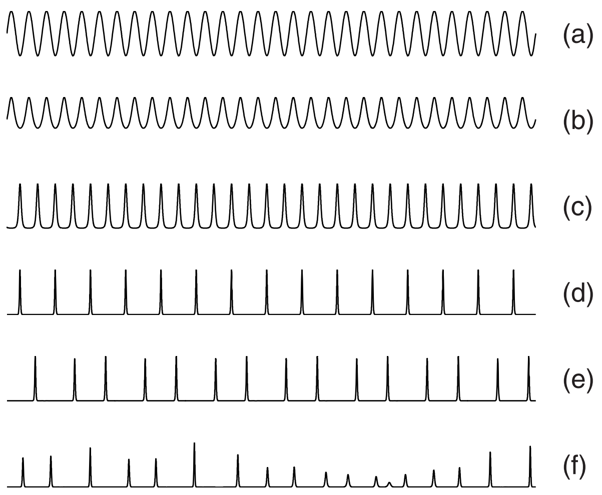


Fig. 3. Temporal dynamics of the single mode fiber laser sinusoidally pumped (a) at a frequency near the resonance frequency and the modulation depths: (b) 0.05; (c) 0.10; (d) 0.30; (e) 0.38; (f) 0.41

$$\frac{dI_2}{dt} = -\frac{1}{\tau} I_2 + \sigma N_0(\beta n_1 + n_2) I_2, \tag{10}$$

where subscripts 1 and 2 refer to the two modes. The parameter γ takes into account the anisotropy in pumping the two modes and β is the cross-saturation parameter. Taking mode 1 as the dominant one, $\gamma < 1$; we consider below $\beta = 0.5$ and $\gamma = 0.85$ [12]. Beside the stationary solution of the dominant mode and that with both modes on, the coupling of the two modes can result in self-pulsations as a result of a Hopf bifurcation [16] as pumping parameter surpasses a critical value.

For pumping modulation it is important to know that the two-mode laser system contains two eigenfrequencies. One has been encountered in the case of the single-mode laser; the other one appears due to the coherent interaction of the two laser modes. They can be observed in the laser response to a step-function pumping (Fig. 4): the dominant mode relaxes at the frequency characteristic to the single-mode laser (high frequency), to the small amplitude laser field is associated the low frequency, and the total laser field relaxes at the high frequency only.

Pumping modulation of a laser with two fields of orthogonal polarization states offers multiple output variants (Fig. 5, Fig. 6). The shape of the pulses, their sequence, or height can be manipulated using the additional tool of a polarizer that retains one state of polarization if necessary.

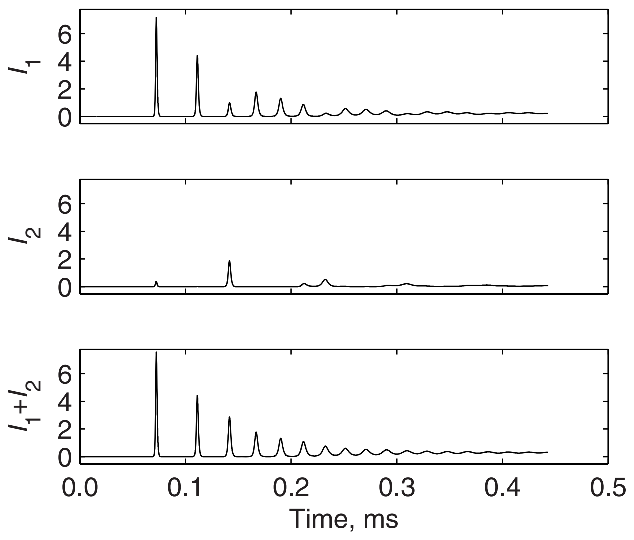


Fig. 4. Relaxation oscillations of the modes 1, 2, and total laser intensity. The laser is pumped at a level twice compared to threshold level for the single mode laser. Photon lifetime is $\tau = 200$ ns and the other parameters are given in text.

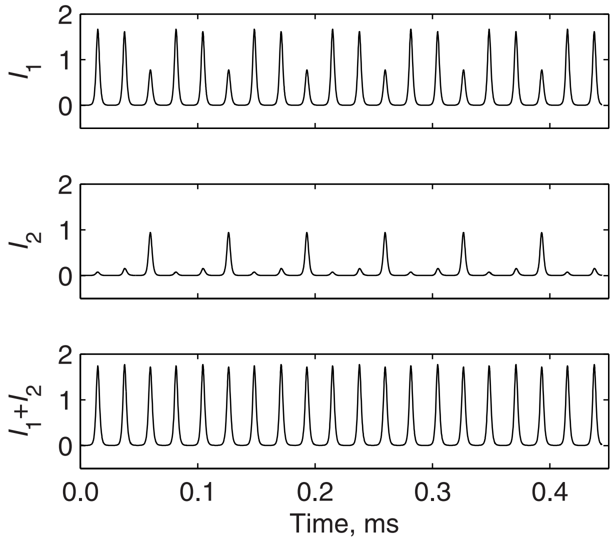


Fig. 5. Pumping modulation of the two-mode laser. The frequency of amplitude modulation is 45 kHz and the modulation depth is 0.25. The other parameters are identical to those used in Fig. 4. The unity for laser intensity is arbitrary.

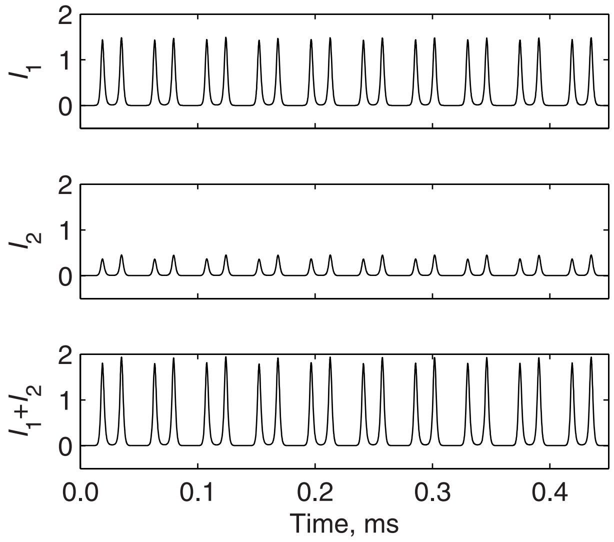


Fig. 6. Pumping modulation of the two-mode laser. The frequency of amplitude modulation is 45 kHz and the modulation depth is 0.62. The other parameters are identical to those used in Fig. 4. The unity for laser intensity is arbitrary.

4 Conclusion

As a member of the class B lasers, the single-mode fiber laser delivers a constant output under constant pumping. The nonlinearity of the dynamical system can be exploited under modulation at a frequency near the system resonance frecuency (Fig. 2) and this can be the appropriate method in reaching a large variety of output pulses easily controlled. If the laser has two modes, for a constant pumping the output is constant or self-pulsations takes place at sufficiently large pumping levels. Moreover, two modes of orthogonal polarization states makes it possible the selection of the output polarization state needed in optical stimulation.

References

1. Snitzer, E.: Neodymium Glass Laser. In: Proc. of the Third International Conference on Solid Lasers, Paris, pp. 999–1019 (1963)
2. Koester, C.J., Snitzer, E.: Amplification in a fibre laser. Appl. Opt. 3, 1182 (1964)
3. Agrawal, G.P.: Nonlinear Fiber Optics, 2nd edn. Academic Press, San Diego (1995)
4. Kleine, K.F., Whitney, B., Watkins, K.G.: Use of Fiber Lasers for Micro Cutting Applications in the Medical Device Industry. In: 21st International Congress on Applications of Lasers and Electro-Optics, Scottsdale (2002) ISBN 0-912035-72-2
5. Fractional Photothermolysis Redefines Facial Skin Regeneration Science, `http://www.reliant-tech.com/downloads/FractionalPhotothermolysis.pdf`
6. Sterian, P., Mocanu, E.: Acquisition and Applications of 3D Images. In: Proc. of SPIE, vol. 6785, 678525 (2007)
7. Lazăr, B., Sterian, A., Puşcă, S., Păun, V., Toma, C., Morărescu, C.: Simulating Delayed Pulses in Organic Materials. In: Gavrilova, M.L., Gervasi, O., Kumar, V., Tan, C.J.K., Taniar, D., Laganá, A., Mun, Y., Choo, H. (eds.) ICCSA 2006. LNCS, vol. 3980, pp. 779–784. Springer, Heidelberg (2006)
8. Dumitraş., D.: Biophotonics. All Educational, Bucharest (1999)
9. Izzo, A.D., Walsh, J.T., Jansen, E.D., Bendett, M., Webb, J., Ralph, H., Richter, C.-P.: Optical Parameter Variability in Laser Nerve Stimulation: A Study of Pulse Duration, Repetition Rate and Wavelength. IEEE Trans. Biomed. Eng. 54, 1108–1114 (2007)
10. Izzo, A.D., Suh, E., Pathria, J., Whitlon, D.S., Walsh, J.T., Richter, C.-P.: Selectivity of Neural Stimulation in the Auditory System: A Comparison of Optic and Electric Stimuli. J. Biomed. Opt. 12, 021008 (2007)
11. Sanchez, F., LeBoudec, P., François, P.L., Stephan, G.: Effects of Ion Pairs on the Dynamics of Erbium-Doped Fiber Lasers. Phys. Rev. A 48, 2220–2229 (1993)
12. Sanchez, F., Stephan, G.: General Analysis of Instabilities in Erbium-Doped Fiber Lasers. Phys. Rev. E 53, 2110–2121 (1996)
13. Sterian, A.R., Ninulescu, V.: The Dynamics of Erbium-Doped Fiber Laser. In: Proceedings of SPIE, vol. 5830, pp. 551–555 (2005)
14. Sterian, A.R.: Computer Modeling of the Coherent Optical Amplifier and Laser Systems. In: Gervasi, O., Gavrilova, M.L. (eds.) ICCSA 2007, Part I. LNCS, vol. 4705, pp. 436–449. Springer, Heidelberg (2007)
15. Kressel, H., Butler, J.K.: Semiconductor Lasers and Heterojunction LEDs. Academic Press, London (1977)
16. Crawford, J.D.: Introduction to Bifurcation Theory. Rev. Mod. Phys. 63, 991–1037 (1991)

Quantum Well Lasers for Medical Industry

Valerică Ninulescu, Vlăduţ-Bogdan Nicolae, and Andreea Sterian

Universitary Center of Optical Engineering and Photonics,
Faculty of Applied Sciences, "Politehnica" University of Bucharest,
Splaiul Independenţei 313, Bucharest, 060042, Romania

Abstract. A distributed feedback (DFB) multiple quantum well (MQW) InGaAsP-InGaAs laser for use in biological and medicine industry is investigated. The laser is amplitude modulated and its output features (height, shape and rate of pulses) are investigated in terms of the frequency of modulation and modulation depth. It is proven a large variety of outputs as a result of the nonlinear behavior of the laser.

Keywords: Multiple quantum well laser, optical stimulation, nonlinear dynamics.

1 Introduction

The new generation of lasers, quantum well (QL) lasers [1,2], have a number of practical advantages starting from the control of output energy, wavelength and better tuning. Also, the compact size allows a better manipulation. Additionally, the energy necessary for excitation is the electric current; compared to other types of lasers (CO_2, for instance), QW lasers have a more efficient conversion of the pumped current into the laser beam. The laser physical equations allow a more precise design with smaller construction errors. Nowadays it is proven the fabrication of QW lasers with a wavelength from far infrared to ultraviolet region of the spectrum. This is essential for the conventional laser replacement with a QL laser emitting the same wavelength. This type of laser can work in the continuous wave regime or a pulsatory one, that is why QL lasers can be used in a widespread range of medical interventions [3,4,5,6,7,8,9].

Semiconductor lasers are used mainly in laser mobile systems with applications in eye operations. Moreover it can be used in dermatological applications for treatment of different skin affections or hair obliteration.

Nowadays, modulation of semiconductor lasers is widely used in fiber optics links [10]; a RF or microwave signal is modulated onto the laser beam either directly or externally. We propose the modulation use in medical applications in order to achieve shaped laser pulses in the microwave domain. These pulses can be obtained directly modulating the laser. In the linear regime of modulation, the shape of the modulating signal is preserved in the laser output. As the laser is mathematically described by nonlinear equations, it is quite easy to reach a nonlinear modulation regime. We prove that a continuous change of one parameter can give outcomes of a large variety, thus having a simple method in achieving very different and reproductible laser outputs required in medical studies [11,12].

2 Physical Model

2.1 Rate Equations

We discuss a single-mode MQW laser with no noise [13,14]. The laser is characterized through the photon density S in the optical volume, the density of carriers N in the active volume V, and the density of carriers N_{B} in the barrier volume. The rate equations can be written in the form of the three coupled first-order differential equations

$$\frac{\mathrm{d}N_{\mathrm{B}}}{\mathrm{d}t} = \Gamma_{\mathrm{q}}\frac{I}{eV} - \frac{N_{\mathrm{B}}}{\tau_{\mathrm{c}}} + \Gamma_{\mathrm{q}}\frac{N}{\tau_{\mathrm{e}}}, \tag{1}$$

$$\frac{\mathrm{d}N}{\mathrm{d}t} = \frac{1}{\Gamma_{\mathrm{q}}}\frac{N_{\mathrm{B}}}{\tau_{\mathrm{c}}} - \left[\frac{1}{\tau_{\mathrm{n}}(N)} + \frac{1}{\tau_{\mathrm{e}}}\right] N - v_{\mathrm{g}}G(N,S)S, \tag{2}$$

$$\frac{\mathrm{d}S}{\mathrm{d}t} = \left[\Gamma v_{\mathrm{g}}G(N,S) - \frac{1}{\tau_{\mathrm{p}}}\right] S + \Gamma\beta B N^2. \tag{3}$$

The rate of change for the barrier carriers is determined by the injected current I, carrier capture by the active layers, and carrier escape from the active layers. In Eq. (1) e is the magnitude of electron charge, Γ_{q} is the fraction of MQW laser well occupied by the QL's, τ_{c} is the capture time of carriers, and τ_{e} is the escape time of carriers from the QW's. The rate of change for the carriers in the QW's is caused by carrier capture from the barriers, losses in the QW's, carrier escape, and stimulated recombination in the optical resonator. The carrier lifetime $\tau_{\mathrm{n}}(N)$ is

$$\frac{1}{\tau_{\mathrm{n}}(N)} = A + BN + CN^2, \tag{4}$$

where A is the carrier non-radiative recombination rate, B is the bimolecular recombination rate, and C is the Auger recombination constant. In Eq. (2), v_{g} is the group velocity of light, and $G(N,S)$ is the optical gain modeled by

$$G(N,S) = \frac{G_0}{1+\varepsilon S}\ln\frac{N}{N_0}. \tag{5}$$

In the above, G_0 is the gain constant, ε is the gain compression factor, and N_0 is the transparent carrier density. Finnaly, photon density change is caused by stimulated emission, losses, and spontaneous emission into the laser mode. Γ is the mode confinement factor, τ_{p} is the photon lifetime, and β is the probability of spontaneous emission into the laser mode.

The total laser power is

$$P = \eta_{\mathrm{d}}\frac{1}{\tau_{\mathrm{p}}}\frac{V}{\Gamma}\frac{hc}{\lambda}, \tag{6}$$

where λ is the wavelength of the laser and η_{d} is the differential quantum efficiency [10].

Typical parameter values of a DFB InGaAsP-InGaAs MQW laser are considered [14]: $\lambda = 1.53\ \mu\mathrm{m}$, $V = 5.1 \times 10^{-17}\,\mathrm{m}^3$, $\Gamma = 0.22$, $\Gamma_{\mathrm{q}} = 0.66$, $\tau_{\mathrm{p}} = 1.3\,\mathrm{ps}$, $\beta = 10^{-6}$, $v_{\mathrm{g}} = 7.5 \times 10^7\,\mathrm{ms}^{-1}$, $G_0 = 141107\,\mathrm{m}^{-1}$, $N_0 = 2.41 \times 10^{24}\,\mathrm{m}^{-3}$, $\varepsilon = 3.24 \times 10^{-23}\,\mathrm{m}^3$, $\tau_{\mathrm{c}} = 20\,\mathrm{ps}$, $A = 10^8\,\mathrm{s}^{-1}$, $B = 10^{-16}\,\mathrm{m}^3\mathrm{s}^{-1}$, $C = 3 \times 10^{-41}\,\mathrm{m}^6\mathrm{s}^{-1}$, $\tau_e = 191\,\mathrm{ps}$, $\eta = 0.2$.

2.2 The Stationary State

Solving Eqs. (1)–(3) for the stationary state ($\mathrm{d}N_{\mathrm{B}}/\mathrm{d}t = \mathrm{d}N/\mathrm{d}t = \mathrm{d}S/\mathrm{d}t = 0$), we find one stationary solution; its dependence on the driving current is seen in Fig. 1. The sudden rise in S marks the position of the laser threshold; it is given by

$$N_{\mathrm{th}} = N_0 \exp(1/\Gamma v_{\mathrm{g}} \tau_{\mathrm{p}} G_0) \quad \text{and} \quad I_{\mathrm{th}} = eV(A + BN_{\mathrm{th}} + CN_{\mathrm{th}}^2)N_{\mathrm{th}}. \tag{7}$$

Above threshold the photon density and laser power are

$$S = \frac{\Gamma \tau_{\mathrm{p}}}{eV}(I - I_{\mathrm{th}}) \quad \text{and} \quad P = \eta_{\mathrm{d}} \frac{hc}{e\lambda}(I - I_{\mathrm{th}}). \tag{8}$$

The stationary state is attained through relaxation oscillations (Fig. 2). To find out an approximate formula of these oscillations, a small deviation from the stationary state is considered. In the linear approximation one gets

$$f_{\mathrm{r}} = \frac{1}{2\pi} \sqrt{\frac{\Gamma v_{\mathrm{g}} G_0}{eVN_0 \exp(1/\Gamma v_{\mathrm{g}} \tau_{\mathrm{p}} G_0)}(I - I_{\mathrm{th}})}. \tag{9}$$

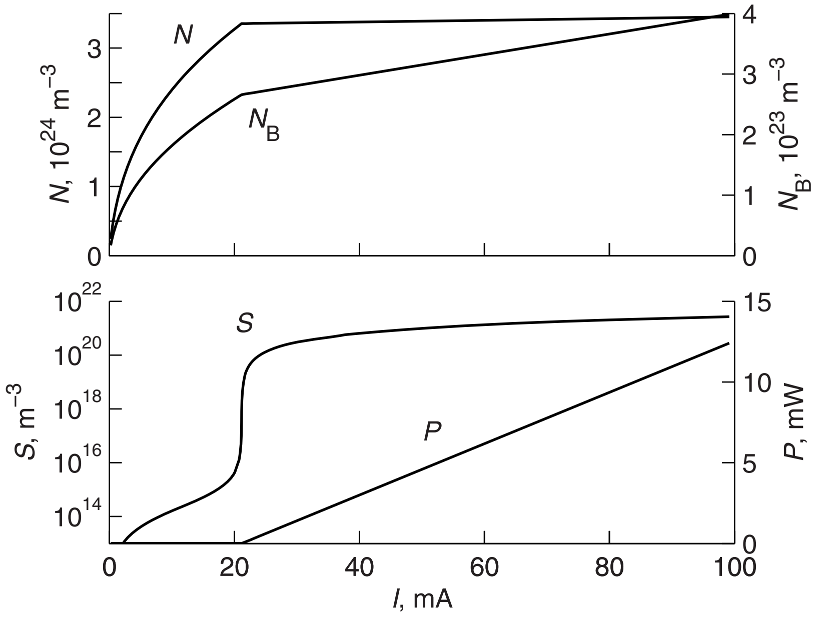


Fig. 1. Laser physical variables, N_{B}, N, S and P, in the stationary state, versus the injected curent

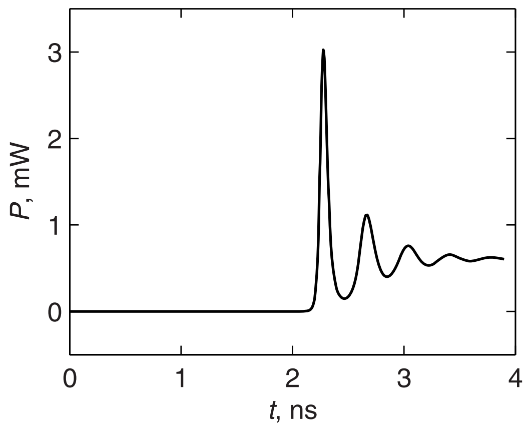


Fig. 2. At $t = 0$ the laser switch is turned on. After some relaxation oscillations, the stationary state is reached. The injected current is $I = 25\,\mathrm{mA}$.

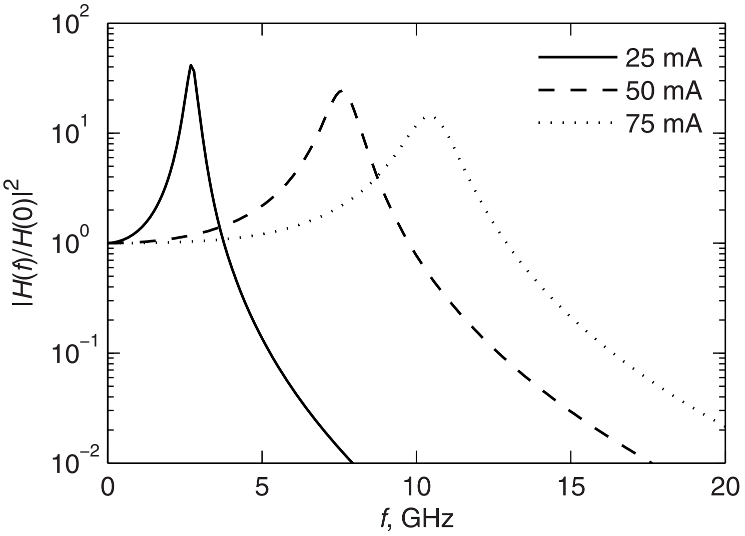


Fig. 3. Transfer function of the MQW laser in the linear approach. Parameters are given in text.

2.3 Transfer Function

Many applications of semiconductor lasers use modulated semiconductor lasers. In the direct modulation case, the injected current in the laser is

$$I = I_{\mathrm{DC}} + I_{\mathrm{RF}} \sin(2\pi f t), \tag{10}$$

where I_{DC} is the dc bias current, I_{RF} – the amplitude of modulation and f – the frequency of modulation. The transfer function is defined as [2]

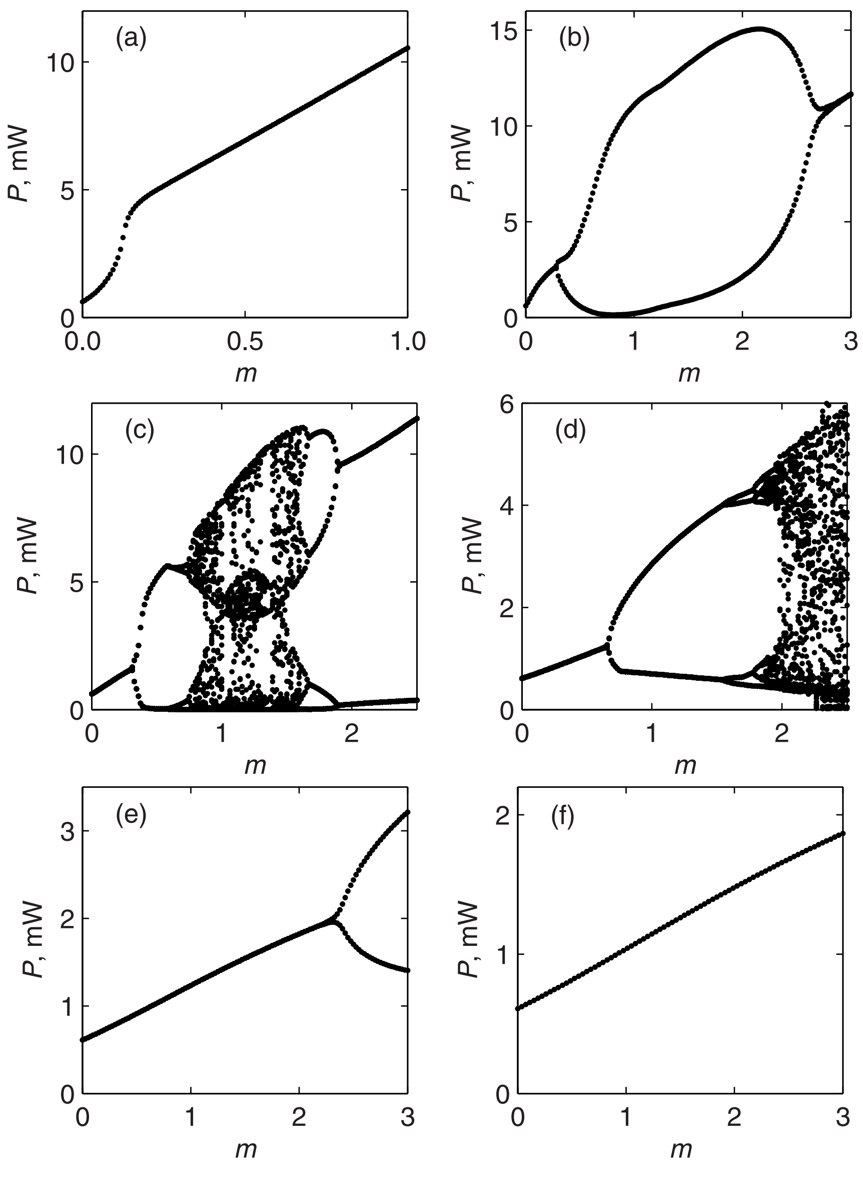


Fig. 4. Bifurcation diagram of the MQW laser for $I_{\mathrm{DC}} = 25\,\mathrm{mA}$ and the frequency of modulation (a) 2 GHz, (b) 3 GHz (c) 4 GHz, (d) 6 GHz (e) 7 GHz, and (f) 8 GHz

$$H(f) = \frac{\delta S}{I_{\mathrm{RF}}}. \tag{11}$$

In the linear approximation the calculus is easily performed; Fig. 3 shows the transfer function for three dc input currents. The function is maximum for f equal to the relaxation frequency [Eq. (9)]. In the next section we write $I_{\mathrm{RF}} =$

mI_{DC}, with m the modulation index and the analysis is not limited to the linear case ($0 < m \ll 1$).

3 The Modulated MQW Laser

Equations (1)–(3) are integrated using a fourth order Runge-Kutta method in MATLAB. As we are interested in the asymptotic dynamics, transient dynamics is everywhere discarded.

Figure 4 presents some bifurcation diagrams of the output power versus the modulation index m for frequencies around the relaxation frequency (I_{DC} is fixed). In the vicinity of f_{r} there are bifurcations with period doubling and reverse bifurcations [15]. The number of bifurcations decreases as the modulation frequency goes away from f_{r} and finally no bifurcations appear. Besides, a larger modulation is required to initiate the bifurcations.

Figure 5 shows a bifurcation diagram for $I_{\mathrm{DC}} = 30\,\mathrm{mA}$ and $f = 7\,\mathrm{GHz}$. Some typical temporal dynamics and evolutions in the plane $N - P$ are given in Fig. 6. At small indices of modulation the linear regime is satisfied ($m = 0.1$ in Fig. 6). Before the first period doubling bifurcation point, the output takes the shape of short pulses (not showed) with the period $T = 1/f$. In the period doubling range of m, we show a period $2T$ dynamics ($m = 1$) and a $4T$ one ($m = 1.8$). After the accumulation point of the period doubling bifurcations, a chaotic deterministic dynamics settle down (eg. $m = 2.3$). At the exit from this region we find a period $3T$ dynamics ((eg. $m = 2.7$). The discontinuous jump in Fig. 5 at $m \approx 2.6$ is a clear indication that multistability occurs there. This can be proven for example by tracing the bifurcation diagram for decreasing modulation index.

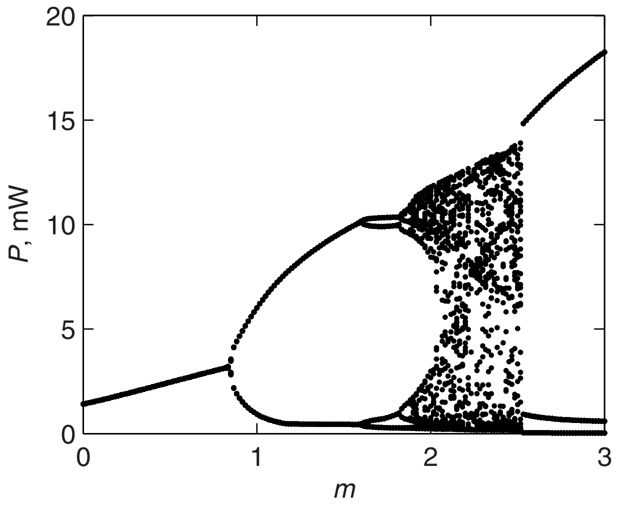


Fig. 5. Bifurcation diagram of the MQW laser for $I_{\mathrm{DC}} = 30\,\mathrm{mA}$ and the frequency of modulation $f = 7\,\mathrm{GHz}$

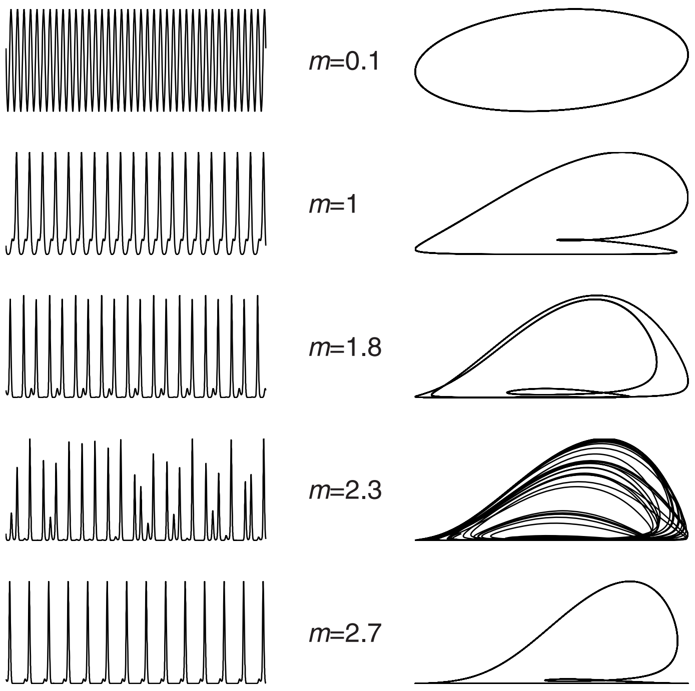


Fig. 6. Temporal evolution in 40 periods of modulations (*left*) and the corresponding phase portraits in the plane $N - P$ (*right*) for some dynamics from Fig. 5

4 Conclusion

The nonlinear dynamics of a MQW laser with direct modulation is characterized in some details. Even if such nonlinear phenomena can be obtained with other types of lasers, semiconductor lasers are unique in the time scale (nanosecond) for such phenomena. This is essential for biostimulation applications characterized by such short relaxation times.

The model presented [Eqs. (1)–(3)] make use of the rate-equation approximation. For more complete treatments a master equation approach is required [16].

References

1. Zory Jr., P.S.: Quantum Well Lasers. Academic Press, San Diego (1993)
2. Kressel, H., Butler, J.K.: Semiconductor Lasers and Heterostructure LED's. Academic Press, New York (1977)
3. Karu, T.I.: Photobiology of Low-Power Laser Therapy. Harwood Academic, London (1989)
4. Dumitraş, D.: Biophotonics. All Educational, Bucharest (1999)
5. Pokora, L.J.: Semiconductor Lasers in Selected Medical Applications. In: Proc. SPIE, vol. 2203, pp. 31–54 (1995)

6. Charamisinau, I., Happawana, G., Evans, G., Rosen, A., Hsi, R.A., Bour, D.: Semiconductor Laser Insert with Uniform Illumination for Use in Photodynamic Therapy. Appl. Opt. 44, 5055–5068 (2005)
7. Sterian, P., Mocanu, E.: Acquisition and Applications of 3D Images. In: Proc. of SPIE, vol. 6785, p. 678525 (2007)
8. Lazăr, B., Sterian, P.: Photonic Crystals Resonant Cavities. Quality Factors. J. of Optoelectronics and Advanced Materials 10, 44–54 (2008)
9. Ghelmez, M., Toma, C.I., Sterian, P.E.: Study of Some Laser Signals Emergent from Nonlinear Optical Media. Computer Phys. Commun. 147, 633–636 (2002)
10. Pipreck, J., Bowers, J.: Analog Modulation of Semiconductor Lasers. In: Chang, W. (ed.) RF Photonic Technology in Optical Fiber Links. Cambridge University Press, Cambridge (2002)
11. Izzo, A.D., Walsh, J.T., Jansen, E.D., Bendett, M., Webb, J., Ralph, H., Richter, C.-P.: Optical Parameter Variability in Laser Nerve Stimulation: A Study of Pulse Duration, Repetition Rate and Wavelength. IEEE Trans. Biomed. Eng. 54, 1108–1114 (2007)
12. Izzo, A.D., Suh, E., Pathria, J., Whitlon, D.S., Walsh, J.T., Richter, C.-P.: Selectivity of Neural Stimulation in the Auditory System: A Comparison of Optic and Electric Stimuli. J. Biomed. Opt. 12, 021008 (2007)
13. Nagarajan, R., Ishikawa, M., Fukushima, T., Geels, R.S., Bowers, J.E.: High Speed Quantum-Well Lasers and Carrier Transport Effects. IEEE J. Quantum Electron. 28, 1990–2008 (1992)
14. Bennett, S., Snowden, C.M., Iezekiel, S.: Nonlinear Dynamics in Directly Modulated Multiple-Quantum-Well Laser Diodes. IEEE J. Quant. Electron. 33, 2076–2083 (1997)
15. Crawford, J.D.: Introduction to Bifurcation Theory. Rev. Mod. Phys. 63, 991–1037 (1991)
16. Stefanescu, E., Sterian, P.: Exact Quantum Master Equations for Markoffian Systems. Optical Engineering 35, 1573–1575 (1996)

Computational Model for the Study of the Fractal Parameters Characteristic for the Surface of Titanium Dental Implants

Stefan Pusca[1] and Theodora Toma[2]

[1] Politehnica University, Department of Physics, Bucharest, Romania
[2] Smart Edu Software Publishing LTD, Bucharest, Romania

Abstract. In order to check if the Fractal theory could be a useful tool for some quantitative descriptions of the fracture parameters, the present work studied diferent theoretical models. The possibility of using different theoretical models (e.g. the Bazant's Size Efect Law (SEL) [1], the Modifed Size Efect Law [2, 3] and the Carpinteri's MultiFractal Scaling Law (MFSL) [4] wich have been already confirmed for the fracture parameters of concrete specimen, and the compatibility of some of the above studied theoretical models relative to the experimental data, using certain recent procedures to study the global and local compatibility have been analysed. The fracture parameters can be considered as main quantities for computational procedures for modeling the fracture of a certain ensemble (a suddenly emerging phenomena). In the next phase, the thermoelastic generation of ultrasonic perturbations in titanium implant material coated with hidroxiapatite was analyzed (using computer simulation) so as to find similarities with material properties as fractal dimensions. The algorithm, the numerical analysis has taken into account three main physical phenomena: the absorption of electromagnetic energy in substance with heat generation; thermal difusion with electromagnetic energy based heat source and elastodynamic wave generation by thermoelastic expansion.

1 Introduction

The applications in Physics of the mathematical theory of the ideal fractals [5] need a good understanding of this concept, from the physical point of view.

A frst attempt in this direction was done recently by M. Rybaczuk and W. Zielinski [6], the presence of measurement errors being presented in [7]. In frame of their well-known paper [11], B.B. Mandelbrot and his collaborators claimed that the value of (the fractal dimension) D decreases smoothly with an increase of the impact energy and D is shown to be a measure of toughness in metals; moreover, Mandelbrot stated [12] that the frequent use of numbers by physicists is a mistake and it would be better if they could focus mainly on the study of plots. Taking into account that for high accuracy of the experimental data, the theoretical relations are not more compatible with these experimental data, even for high values of the correlation coefcient, and using the existence for concrete of several experimental data [3], [14] referring to the size dependence of the fracture parameters, as well as of some (multi)fractal descriptions of these efects (see also [7], [8] and [9]), this work will start by presenting a numerical analysis of these existing experimental data, as well as of the compatibility of the multifractal and similitude expressions of the fracture parameters relative to the analyzed experimental results.

2 Experimental Data for the Size Dependence of Fracture Parameters of Some Concrete and Rock Specimen and for Some Titanium Implant Coated with Hidroxiapatite

The study of the compatibility of some theoretical relations relative to the analyzed experimental data needs the previous elimination of the rough errors. This was performed using the Chouvenet's criterion [15], and eliminating the individual values whose absolute values of the reduced errors were larger than the Chouvenet's threshold. The use of this procedure to the study of the numerical values of some fracture parameters [3], [14] pointed out that some individual values of the critical strain corresponding to some specimens of size equal to 20cm of dry and wet concrete [3], respectively are roughly erroneous. After the elimination of these individual values, the mean values were recalculated for the compatible individual values. The obtained results corresponding to the tensile strength Ts(MPa), critical strain (deformation) w(fm) and to the fracture energy Gf(N=m) for specimens of diferent natures (materials) and sizes were synthesized in the following tables; they present the average values corresponding to some existing experimental data concerning the main fracture parameters of some concrete and rock specimens, respectively.

Table 1. Material: Concrete

Specimen size (cm)	Av. value $Ts(MPa)$	Av. value $w(\mu m)$	Av. value $Gf(N/m)$
2.5	4.7925	-	146.25
5.0	4.56	-	257.57
10.0	4.3543	-	236.25
20.0	3.80	-	158.0
40.0	3.7225	-	214.0

Because the studies [13], [14] point out a considerable curvature of the plots for certain logarithmic representations, some semiempirical expressions of the tensile strength and of the fracture energy have to be obtained (which difer to that of Bazant [5], [6] and to those of Carpinteri [7], [8]).

3 Obtained Results

The obtained results concerning the compatibility of the (multi)fractal (Carpinteri's) expressions and of the classical elasticity theory (Bazant's) expressions of the

Table 2. Material: DRY Concrete

Specimen size (cm)	Av. value $Ts(MPa)$	Av. value $w(\mu m)$	Av. value $Gf(N/m)$
5.0	2.536	1.64	97.045
10.0	2.9725	5.20	125.70
20.0	2.750	9.833	124.243
40.0	2.298	14.78	125.22
80.0	2.0725	22.025	142.30
160.0	1.8575	44.40	141.10

Table 3. Material: WET Concrete

Specimen size (cm)	Av. value $Ts(MPa)$	Av. value $w(\mu m)$	Av. value $Gf(N/m)$
5.0	2.174	2.180	91.48
10.0	2.230	4.10	99.66
20.0	2.476	8.025	88.92
40.0	2.365	15.725	100.367

Table 4. Material: TITANIUM IMPLANT COATED WITH HIDROXIAPATITE

Specimen size (cm)	Av. value $Ts(MPa)$	Av. value $w(\mu m)$	Av. value $Gf(N/m)$
5.0	2.450	28.45	76.75
10.0	1.2167	44.167	111.333
20.0	1.010	69.20	93.80
40.0	0.960	156.80	135.067
80.0	1.300	263.70	143.350
160.0	1.200	434.70	81.80

size-efects presented by the tensile strength Ts, the fracture energy Gf and by the critical strain w relative to the existing experimental data are synthesized in frame of next table. The analysis points out: a) the local incompatibility of the studied theoretical models with the experimental data (TM/ED) corresponding to the tensile strength of the Titanium implant material hidroxapatite coated for 3 or 4 sizes (from the 6 studied ones) of the studied specimens, the Carpinteri's expression being though somewhat more accurate, b) the compatibility TM/ED for Dry Concrete specimens, the Bazant's description being somewhat more accurate, c) the compatibility TM/ED for Wet Concrete, but with strange (negative) values of the characteristic lengths, d) the compatibility TM/ED for Gf, with somewhat better accuracy and physical meaning of the values of the characteristic lengths obtained by means of Carpinteri's model for the Titanium implant material hidroxapatite coated and Dry Concrete specimens,

Table 5.

Sp(Ref.)	Rs dm (Th-m)	Rfd	Rev I	MPa	dm
(RFS) [3]	0.5...16 [7]-[9]	1.6747...1.9725	11.831	1.02726	9.31076
RFS [3]	0.5...16 [4]-[6]	1.99896...1.99997	31.941	1.15839	77082.17
(DryC) [3]	0.5...16 [7]-[9]	1.76495...1.98651	17.427	2.0753	4.43573
Dry C. [3]	0.5...16 [4]-[6]	1.71015...1.97934	14.0293	2.7873	116.005
(WetC) [3]	0.5...4 [7]-[9]	2.01381...2.13695	9.9172	2.43535	- 1.075
Wet C. [3]	0.5...4 [4]-[6]	2.01054...2.0989	9.3845	2.21002	242.225
Conc. [14]	0.25...4 [7]-[9]	1.78465...1.97743	9.5395	3.74006	1.8913
Conc. [14]	0.25...4 [4]-[6]	1.77875...1.9764	9.3627	4.77456	50.3977

e) the compatibility TM/ED for Gf, with somewhat equal accuracy of both studied models for the Wet Concrete, f) the compatibility TM/ED with somewhat better accuracy (excepting the van Vliet's results for Dry Concrete) of Bazant's type description of w(b).

In this table Sp(Ref.) represents Specimen (with Reference written into brackets), Rs dm represents Range of sizes in dm (with Theoretical model written into brackets), Rfd represents Range of fractal dimensions, Rev I represents Ratio of extreme values of Increment, Ts (in MPa) represents tensile strength and Lch (in dm) represents the characteristic length the characteristic lengths obtained by means of Carpinteri's model for the Titanium implant material hidroxapatite coated and Dry Concrete specimens.

The accomplished study allows the obtainment of the following conclusions:

a) despite of its fruitful qualitative contribution (which allowed the derivation of Carpinteri's relations) of the Fractal Theory to the description of the size efects on fracture parameters, itself this theory cannot ensure always accurate results concerning the fracture parameters corresponding to specimens of diferent dimensions, b) it seems that the fractal character of fracture surfaces and the (classical) elasticity theory implications on the size-efects of fracture parameters represent cooperative processes, the most accurate descriptions implying (generally) contributions of both these factors.

4 Ultrasonic Waves, Generated by Heat Sources

Short ultrasonic pulses can be induced in solids using a range of techniques, most of them being direct contact procedures. In other words, the excitation device has to be placed directly onto the surface of a certain solid body. However, using laser energy based thermoelastic expansion of the substance [22], [23], the disadvantage of direct coupling is eliminated. In essence, a strong and short time volumic dilatation appears if a narrow and short duration beam of light, having high enough power density in its transversal section, hits a solid surface belonging to a TDI HC object. The laser energy, absorbed within the penetration depth is transformed in heat, which in its turn leads to an increase in the temperature of the irradiated portion of substance. The quick dilatation that follows is in fact the source of a high frequency elastic perturbation which propagates inside and onto the separation surface of the investigated object [24], [25].

The thermogeneration of ultrasound relies on an equation that makes the connection between the source time dependent thermal gradient and the generated elastodynamic feld is needed. Secondly an equation that associate the thermal gradient with the laser generated heat distribution, Q(r; t), inside the TDI HC object, is required. This connection is made by the thermal difusion equation. The only unknown that remained to be expressed in a mathematical form is Q(r; t) which depends on the particular property of the radiated material. The heat distribution can be considered as having an exponential decay rate with the depth in substance, according to

$$Q(z,t) = F(z)G(t) = Q_0 exp\left[-\eta^2\left(z-\frac{h}{2}\right)^2\right]\frac{t}{t_0}\exp\left(-\frac{t}{t_0}\right) \tag{1}$$

or can be represented as a test function - for example the bulk-like function

$$\varphi(\tau) = \begin{cases} \exp\left(\frac{1}{\tau^2-1}\right) & \text{if} \quad \tau \in (-1,1) \\ 0 \text{ otherwise} \end{cases} \tag{2}$$

or a practical test function. Unlike test functions, practical test functions can be represented as solutions of diferential equations as

$$\varphi^{(2)} = \frac{6\tau^4-2}{\left(\tau^2-1\right)^4}\varphi \tag{3}$$

(generating a function similar to the bump-like function). Yet such practical test-functions possess only a fnite number of continuous derivatives on the whole real axis.

Numerical simulations for an usual medium which receives a laser pulse with parameters: t0 = 10ns;Q0 = 1x1015W=m3 show a thermoelastically generated ultrasonic pulse traveling inside the structure with a speed less than 1cm=s.

This velocity corresponding to a certain axis can be put in correspondence with characteristics of material (characteristic length of fractal structure, for example, as was shown in previous paragraphs). However, such generalization procedures must fulfll certain requirements which are studied at this time moment by the working team:

a) the possibility of establishing time relaxation constants for explaining propagation patterns of sound and ultrasound waves (or other type of vibrations) inside the structure

b) the accuracy of measuring methods for propagation patterns inside materials (for selecting the proper theoretical model using measuring methods of higher accuracy)

c) the possible implementation of specifc computational architecture based on parallel computing, with many computing units (corresponding to a granular component of the material) connected in greater computing units (corresponding to medium size component of the material) and with the same type of connections implemented at a large scale, according to fractal laws.

5 Some Elements on the Physical Similitude and Fractals Theory

For a rigorous connection between propagation patterns and the structure of material, we must search for the relaxation time constants (for electromagnetic or acoustic phenomena) as related to the fractal structure of the material. In order to do this, some elements of physical similitude and fractal theory must be taken into account.

Unlike the mathematical systems (geometrical fgures, symmetry groups, polynomes, etc), whose elements are determined by a given (specifc) number nU of uniqueness parameters, the number Ui of uniqueness parameters corresponding to a physical state (or process) depends on the required accuracy, increasing with the accuracy level. If the physical dimension of a parameter specifc to the studied state (or process) is:

$$[P] = \prod_{i=1}^{n_U} [U_i]^{\alpha^i} \tag{4}$$

then 2 states (or processes) S', S" are named similar if the values of the parameters [Ui]; i = 1::n and P corresponding to these states fulfll the relation:

$$\frac{P'}{P"} = \prod_{i=1}^{n^U} \left(\frac{U'_i}{U"_i} \right)^{\alpha_i} \tag{5}$$

Some of the uniqueness parameters could be similitude criteria, i.e. non dimensional parameters: [s] = 1 , with equal values: s0 = s" in all similar states or processes. In the macroscopic Physics, the similitude indices fi are integers or semi-integers, very seldom intervening other rational values. In order to explain the rather strange similitude indices intervening in the description of turbulent ows parameters, Kolmogorov proposed a hierarchical structure of vortices, the energy being injected frstly in the largest vortices and transferred in a cascade from the larger to the smaller vortices, up to the smallest ones, where the energy is dissipated. This hypothesis was strengthened by the contributions of Mandelbrot. Taking into account that (for samples of diferent sizes) it is difcult to have exactly equal values of all uniqueness parameters (excepting the sample size), in order to fulfll the classical relation:

$$C = L^D \tag{6}$$

(with a non-rational similitude index D, called the fractal dimension of the fractal theory), it results that the physical applications of the fractal theory correspond to the prevalence (dominance) of the size (length) uniqueness parameter. If for diferent size domains, the values of D belong to a set (discrete or continuous) of real numbers, the corresponding physical structure is called multi (poly)fractal. In the last years, there were published several identifcations of multi (poly)fractal structures, as those corresponding to: the fracture surfaces of metals, (ii) fracture surfaces of concrete specimen, (iii) several parameters of disordered and porous media, aggregates, polymers and membranes , (iv) electrode surfaces (of fractal dimension. We have to underline that even some classical equations of the relaxation phenomena could reect some fractal structures. E.g., the well-known relations of Cole-Cole for dielectrics, and Mikami for magnetic materials,

$$\epsilon' = \frac{\epsilon_0 + \epsilon_\infty (i\omega\tau)^\beta}{1 + (i\omega\tau)^\beta}, \quad \chi' = \frac{\chi_0}{1 + (if/f_l)^\beta} \tag{7}$$

can be obtained starting from constitutive equations as:

$$D + \tau^\beta \frac{d^\beta D}{d\tau^\beta} = \epsilon_0 E + \epsilon_\infty \tau^\beta \frac{d^\beta E}{d\tau^\beta} \tag{8}$$

using fractional derivatives. In order to explain the appearance of both the frequency power laws and fractals, we will mention here also some theorems of the physical similitude: a) the number nis of irreducible similitude numbers (criteria) corresponding to a state (or process) of a physical system is equal to the diference between the number nU of independent uniqueness parameters and the number naF of the active fundamental physical quantities, and b) every physical law or relation can be expressed by means of some similitude numbers (criteria) and only by means of similitude criteria (Federman's theorem).

The time constants corresponding to relaxation phenomena can be observed as time constants for transient phenomena when external pulses are applied or generated at the material surface; an accurate measurement can't be achieved without using a great amount of energy received on a very short time interval, and ultrashort high-energy laser pulses are the best choice.

6 (Multi)Fractal Scaling and Frequency Power Laws

The fractal distribution of the main parameters (area, perimeter) of the slit islands (of steel, surrounded by nickel) of the fracture surfaces was pointed out for some steel specimen plated with electroless nickel. Assuming a similar fractal distribution of the grains of some ferrimagnetic materials, the efectively occupied (by grains) part of a volume V = l3 being V f = l3□d (where d is the corresponding fractal decrement), it results that the magnetic energy stored in a such material is:

$$W = \frac{B^2}{2\mu_{app}} V = \frac{B^2}{2\mu} V^* \tag{9}$$

That is why we can compare the main parameters of the parabolic correlations:

$$\mu_{app} = \mu \frac{V}{V*} = \mu l^d \Rightarrow \log \mu_{app} = \log \mu + d \log l \tag{10}$$

corresponding to multifractal scalings of some microstructural parameters of certain alloys (steel) and of some ferrimagnetic materials, respectively. A study of the experimental results shows a multi-fractal grains size dependence of the apparent permeability of some Mn-Zn ferri-magnetic materials, somewhat similar to that reported for some steels.

Both the electromagnetic waves dispersion and the elastic waves propagation can be described by means of the 6 uniqueness parameters: specimen size D, microelements (grains, inclusions, cracks, pores, etc) size d, characteristic (wave phase, oscillations) velocity v and frequency f, the density f and the dynamic viscosity f of the medium. Because the above indicated phenomena have a dynamical character (with 3 active fundamental quantities), it results that the number of irreducible similitude criteria for the above processes is: nis = 6 □ 3 = 3.

These irreducible similitude criteria can be chosen as: D=d; fD=v; fdv=f. From the Federman's theorem of the similitude theory, it results that any other physical parameter (similitude criterion) can be expressed as a function of (only) irreducible criteria:

$$\log P = c_0 + c_1 \log l + c_2 \log P^2 \tag{11}$$

According to Barenblatt's theorem, if f is a self-similar (and diferentiable) function, so that f(x2=x1) = f(x2)=f(x1), then f(x) = xn1 , where n1 = f0

$$P = f(D/d, fD/v, \rho dv/\eta \tag{12}$$

This explains the concomitant appearance of: (i) frequency power laws (of flicker noise, particularly): P = fn3 , (ii) size efects relative to the microstructure elements sizes, (iii) size efects relative to the specimen dimensions. The concomitant presence of some frequency power laws and of the fractal scaling seems to indicate the appearance of some self-organized criticality states; it allows us to determine the fractal parameters of the material using also a set of measurements based on frequency power laws. This requires the use of continuous waves inside the material (instead of pulses) and a good intensity and frequency stabilization for the laser source.

7 Conclusions

This paper has presented an analysis of compatibility with experimental data of fractal descriptions of the fracture parameters for materials as concrete, wet concrete and titanium implant material hidroxapatite coated. The diferences between predicted values according to certain theoretical models and experimental data were analyzed, being pointed out the fact that this aspect requires better theoretical and experimental methods. Then it was shown that major advances would be achieved if the fractal methods for spatial structure of materials would be replaced by four-dimensional fractal models able to predict also the propagation patterns of sound and ultrasound waves (or other type of vibrations) inside the structure.

By comparing the experimental patterns (an example being the heat generated ultrasound pulses in a TDI HC medium, difering to the rocks presented in frst paragraph - electrical and optical insulating materials, and difering also to conductors where the small penetration depth, characteristic to conductors, acts as a singularity similar to a short impulse, leading to a very broad spectrum and to a great number of coefcients afecting the accuracy of numerical simulations) we can check the validity of theoretical models. For a better accuracy, specifc computational architecture based on parallel computing (with many computing units - corresponding to a granular component of the material- connected in greater computing units corresponding to medium size component of the material and with the same type of connections implemented at large scale, according to fractal laws) should be implemented.

References

1. Bazant, Z.P.: Size efect in blunt fracture: Concrete, rock, metal. Journal of Engineering Mechanics, ASCE Journal of Engineering Mechanics, ASCE 110, 518–535 (1984)
2. Kim, J.K., Eo, S.H.: Size efect in concrete specimens with dissimilar initial cracks. Magazine of Concrete Research 42, 233–238 (1990)
3. Bazant, Z.P., Kazemi, M.T., Hasegawa, T.: Size efects in Brazilian split-cylinder tests: measurements and fracture analysis. J. Mazers, ACI Materials Journal 88, 325–332 (1991)
4. Carpinteri, A.: Fractal nature of material microstructure and size efects on apparent mechanical properties. Mechanics of Materials 18, 89–101 (1994)
5. Mandelbrot, B.B.: The Fractal Geometry of Nature. W. H. Freeman, San Francisco (1982)
6. Rybaczuk, M., Zielinski, W.: The concept of fractal dimension. Chaos, Solitons and Fractals 12, 2537–2552 (part I), 2537-2552 (part II) (2001)
7. van Vliet, M.R.A.: Ph. D. Dissertation, Tehnical University of Delft (January 31, 2000)
8. Carpinteri, A., Ferro, F.: Scaling behaviour and dual renormalization of experimental tensile softening responses. Materials and Structures 31, 303–309 (1998)
9. Carpinteri, A., Chiaia, B.: Size Efects on Concrete Fracture Energy: dimensional transition from Order to disorder. Materials and Structures 29, 259–264 (1996)
10. Iordache, D.: On the Compatibility of some Theoretical Models relative to the Experimental Data. In: Proceedings of the 2nd Colloquium Mathematics in Engineering and Numerical Physics, Bucharest, vol. 2, pp. 169–176 (2002)
11. Mandelbrot, B.B., Passoja, D.E., Paullay, A.J.: Fractal character of fracture surfaces of metals. Nature 6, 721–722 (1984)
12. Mandelbrot, B.B.: Opinions. Fractals (Complex Geometry, Patterns, and Scaling in Nature and Society) 1(1), 117–123 (1993)
13. Delsanto, P.P., Iordache, D., Pusca, St.: Study of the Correlations between different efective fractal dimensions used for fracture parameters descriptions. In: First South-East European Symposium on Interdisciplinary approaches in fractal analysis, Bucharest, May 7-10 (2003)
14. Carpinteri, A., Ferro, F.: Scaling behaviour and dual renormalization of experimental tensile softening responses. Materials and Structures 31, 303–309 (1998)
15. Jnossy, L.: Theory and Practice of the Evaluation of Measurements. Oxford University Press, Oxford (1965)

16. Gukhman, A.A.: Introduction to the Theory of Similarity. Academic Press, New York (1965)
17. Eadie, W.T., Drijard, D., James, F.E., Roos, M., Sadoulet, B.: Statistical Methods in Experimental Physics. North-Holland Publ. Company, Amsterdam (1982)
18. Eadie, W.T., Drijard, D., James, F.E., Roos, M., Sadoulet, B.: Handbook of Applicable Mathematics. In: Ledermann, W. (ed.) Statistics, vol. VI. John Wiley & Sons, New York (1984)
19. John, P.W.M.: Statistical Methods in Engineering and Quality Assurance. John Wiley & Sons, New York (1990)
20. Levenberg, K.: A method for the solution of certain nonlinear problems in least squares. Quart. Appl. Math. 2, 164–168 (1944)
21. Marquardt, D.W.: An algorithm for the least-square estimation of non-linear parameters. J. of Soc. Industr. Appl. Math. 11, 431–441 (1963)
22. Gaelle, R., Pandora, P., Roland, O., Sophie, C., Christian, C.: Simultaneous Laser Generation and Laser Ultrasonic Detection of Mechanical Breakdown of a Coating Substrate Interface, Ultrasonics (2001)
23. Storkely, U.: GHz Ultrasound Wave Packets in Water Generated by an Er Laser. J. Phys. D: Appl. Phys. (1998)
24. Kritsakorn, L., Wonsiri, P., Laurence, J.J.: Guided Lamb Wave Propagation in Composite Plate. Journal of Engineering Mechanics, 1337–1341 (2002)
25. Frass, A., Lehmann, G., Lomonosov, A., Hess, P.: Linear and Nonlinear Elastic Surface Waves; From Seismic Waves to Materials Science. Analytical Sciences 17, 9–12 (2001)

VEMS-TM – A Management Infrastructure

Trandafir Moisa and Cristian Morarescu

CornerSoft Technologies S.R.L, Bucharest, Romania

Abstract. Virtual Enterprise Management System (VEMS-TM) is an Internet workspace that allows users to share and manage community information associated with projects and other enterprise activities. This web based workspace runs on top of a work and document flow management engine, able to automatically/manually enact project tasks and to manage and monitor them and their related information. In this approach VEMS is used as an elearning infrastructure for lab work.

1 Introduction

In a world where time to market means everything, efficient enterprise-wide project management for software system engineering is critical. VEMS-TM is focused on providing Internet/Intranet communication and collaboration solutions to the virtual enterprise. VEMS-TM delivers integrated, scalable software project and document management solutions for the entire enterprise. VEMS-TM Virtual Community workspaces, allow organizations to quickly assemble a project team from one end of the globe to the other and manage the communications and collaborative activities that drive the specifications, design, development and delivery of their products and/or services. Through the use of these communities, we are committed to offering the most comprehensive project and document management software and services to meet the requirements of every level of an organization, from the front-line project managers and engineers to the administrative and executive teams. Our solution is instantly deployable and teams can set up software projects and begin collaborating in minutes independent of time or location and with little or no training required.

2 VEMS-TM Architecture Overview

VEMS provides the means that make program and project success a reality, helping organizations around the world bridge the gap between cultural, language and technological boundaries enabling these organization to innovate, communicate, collaborate, negotiate, and interact. VEMS is helping to create horizontal and vertical project communities in a virtual workspaceproviding the a place to get work done.

Community Management: One of the greatest challenges of an organization is to organize and get a global prospective of projects in progress. VEMS enables the

Community Administrator to build “Virtual Hierarchical Structured Community Workplaces”. VEMS provides the functions to group organize and manage projects, customers and suppliers more effectively. VEMS has a robust User Management and Access Rights tool that let the responsible persons to set up payment rates per every individual user according to skills and experience on a project bases. VEMS gives the organization better flexibility and control on who gets in, who sees what and what tools they have the right to use.

Project Management: VEMS provides: Project Planning, Reports, Issue Management, Change Management, Test and Error Management, Deployment and Maintenance, Notifications and much more. VEMS also incorporates advanced Project Tracking tools like Critical Path Analysis, Project Estimating, Time Reporting, Gantt Analysis, EVA, Milestones and tasks behind schedule or over budget notifications. VEMS keeps top priority items in your To Do list. VEMS provides integration with other commonly used project planning tools by allowing users to import and export files from MPX, TXT, RTF and other standard formats.

Document and Workflow Management: Every organization has its own business practices and preferred methods of project control. VEMS enables the creation of custom workflows for handling a multitude of processes and document types such as: Proposals, Statements of Work, Functional Specifications, High Level Design, images, faxes, and others. Users can define workflows unique to the organization or the project and set up multiple dispatchers for handling and distributing the information to various destinations. Along with a robust Windows Explorer style Document Management tool with access rights and version control, VEMS ensures the right process for the right task at the right time all the time!

Communication Management: Communication and Collaboration are critical in every project VEMS provides robust and comprehensive communication tools like: Message Center, Discussion Forums, Integrated Project Navigator, Instant Project Messenger and User defined notifications. So no matter where you are in the world, or what method of communication you prefer, VEMS offers the tools to ensure fast and accurate communication with your project teams.

Quality Management: Standards are constantly changing, but one thing that is constantly needed is a quality assurance management system. VEMS integrates the standards and documents of the global recognized quality standards of IS0 9000 (Products and Services), ISO 14000 (Environment), ISO 18000 (OHSMS Employee) and CMM (Software Development). VEMS provides over 120 document templates to use and customize for implementing a quality system. VEMS extensive CMM reports cover: Progress, Effort, Cost, Quality, Stability, Computer Resource Utilization and Training. So whether you are looking to become ISO or CMM certified, or to maintain an existing certification the VEMS Quality Assurance Management system provides the tools, documents and means to be successful in your quality objectives. VEMS is a web based, system for use within any project driven, document intensive and quality assured organization. VEMS provides tools that reach far beyond the basics of Project Management creating an advanced Community environment of communication, collaboration, planning and control.

3 Software System Lifecycle Structure Covered by VEMS-TM

VEMSTM provides IT companies or IT departments with the functionality to define and improve their process and quality system based on ISO 9001/9000-3 standards and the CMM model. Our quality system embedded in the current version of VEM provides the environment to manage organizations' activities in compliance with CMM level 4 and some reports for CMM level 5.

We are developing the next version of VEMSTM to include all premises of a total management system or specified in ISO 9000, ISO 14000 and ISO 18000 standard series, as well as full compliance with CMM level 5 for software industry.

It is our objective to promote VEMSTM in the global market. Developed as a project management tool and quality system implementation, VEMSTM has been identified as a powerful enterprise management system for corporations where community management, work and document flow management, project management, quality and knowledge management are key needs.

The final goal of VEMSTM is to provide the appropriate tools to define and implement a set of unified system documents, the document work flow functionalities which cover the three quality systems: products and services (ISO 9000), EMS (ISO 14000) and OHSMS (ISO 18000).

In addition to this, VEMSTM provides IT organizations/departments the environment for: - Organization Process Focus - Organization Process Definition - Training Program - Integrated Software Management - Software Product Engineering - Intergroup Coordination and Peer Reviews VEMSTM covers the entire system life cycle procedures starting with project feasibility, contracting, project initiation, project planning, system design, system development, system testing and ending with system deployment (Fig. 5). These are grouped into 3 main phases: Research and Development (CPAF- Cost Plus Award Fee), Production (FFP- Firm, Fixed Price) and Maintenance (LOF-Level Of Effort).

Each phase contains stages of the system which are Proof of Concept and Elaboration for the Research and Development phase, Construction and Deployment for the Production phase. Each stage might be completed through a couple of iterations that ends up with: - Conceptual Prototype for the Proof of Concept stage, - Architectural Prototype and Architecture Baseline for the Elaboration stage, - System Releases for the Construction stage, - System Deliveries for the Deployment stage.

Through the whole system lifecycle, the process is based on activities like: Planning, Analysis, Architecture, Design, Implementation, Integration and Test/Assessment. When a customer requests CST to build an application system, the customer gives some notion of what the system should do. Thus, the purpose of the proposed system is to meet the customer's requirements. A functional requirement document is a feature of the system describing all or some aspect of the system and how it is capable of performing. Although the focus is to determine the nature of the customer's problem, there are two purposes in this stage. On one hand, the requirement analysis yields a functional specification document. Written in terms that the customer can understand, the functional requirement documents everything the customer expects the system to do. This may not be a technical document to be used by system designers. Usually, the technical counterpart of the functional specification document, used by the system designers, is also formed in this stage. In many situations, the requirements from a customer are ambiguous in terms of a technical view. So, the functional requirements need to be specified carefully by our teams together with our clients and VEMSTM enables a very good infrastructure to accomplish all these by means of the communication and document repository modules.

It is the transformation of the problem provided by customers into a functional system. The most important task in system design is to set up the system architecture. There are two parts, high level design and detailed design system. The high level design answers what the system will do for customers. The low-level design explains the system to hardware and software experts who will then implement the system. Writing the code is an extension of the design process. Writing code is straightforward, because all the difficult decisions should already have been made during design. During the production phase, we are very careful with the testing process at system, integration and unit level.

For many software systems, coding does not mean the end of the developer's job. If errors are discovered after the system has been accepted, a maintenance team fixes them. In addition, the customer's requirements may change as time passes, and corresponding changes to the system must be made. Thus, maintenance can involve different personnel: analysts who determine what requirements need to be added or changed, designers who determine where in the system design the change should be made, programmers that implement the changes, testers who make sure that the changed system still runs properly, and trainers who explain to users how the change affects the use of the system.

VEMSTM is an online collaborative business management system and Internet workspace that allows you to communicate, share, manage and distribute information associated with projects and their related tasks or with enterprise activities. The core part of VEMSTM consists of an automated workflow engine that is configurable by the user via a graphical toolkit. Based on this graphical description of the workflow, the engine automatically creates tracks and monitors tasks and their related actions and documents. The engine can work totally independent, the only user intervention being for performing the actions required by task description or for monitoring the other user activities. Besides this automatic behavior of the engine, the user can manually control it by adding new tasks, change assignments, forcing task completion etc.

The test and error management feature provides means to manage errors and issues occurred during the testing activities, allowing for development of the test cases and test suites during design time, posting of errors during testing time and error solving during debugging time. The testing documentation includes system/acceptance test cases, integration test cases, and unit test cases. These documents are produced at design time and they are grouped in test suites during the testing iterations. Once a test suite is generated, the project manager needs to schedule a task and the test suites are handed over to the testers. Once a test case in a test suite has failed, the whole test suite has failed and a test error document is automatically produced. Once an error document is generated, the project manager assigns a task for solving the error. Resources allocated to a test task or to a debug task might be automatically notified by email. Every project defines the organizational framework to develop a system. The system is the durable we obtain when a project is completed. The project defines plans and monitors the actions to develop a system in all steps involved in the system life cycle. Test and Error Management is dealing with the resulting project artifacts we called systems in the virtual enterprise management system. Test and Error Management provides the environment and the tools to test and accept the system developed in the frame of a project during the system life cycle.

The project reports management feature provides a mean for delivering project reports like progress, effort, cost, stability, quality, computer resource utilization and training. All these features cover chapters of the ISO 9000 and CMM standards requirements dealing with software system development lifecycle processes.

4 Software Project Management from a Training Perspective

VEMSTM is well suited for any organization that has the production process organized as projects, like software development projects, building construction, hospitals, government projects, etc. More over if we assimilate a project with a course we can use VEMSTM and the virtual workspace it provides as an eLearning tool especially for courses for ”project management”. Furthermore we can assimilate project tasks with lessons, the project manager with a teacher and team members with students. Home works/projects can be assigned to lessons or to the entire course together with the corresponding terms/milestones. The teacher/student might be notified via different channels like email, SMS, instant messenger, etc when a certain milestone was not reached by a student. All courses can be stored in any multimedia format in the document repository. Publish documents in any format (Word, PDF, HTML, Video...). The document repository provides the possibility to track document changes, so that every course version can be retrieved as the times goes by. During the course lifecycle the teacher may change the access rights to the different course chapters according to the course schedule. Comments can be posted on every course chapter both by student and teacher so that the knowledge is well received by the student. The teacher has the ability to start collaborative workflows or even projects on specific themes. The system allows a teacher to show examples based on real life customs.

Activities like courses, users and groups administration, agenda, documents, announcements, forums, links, student papers, exercises, statistics, add a page to site, link to an external site, modify course information, activate/deactivate course components, queries can be easily achieved with VEMSTM. If we talk to project management specific issues VEMS provides Project Planning, incorporates advanced Project Tracking tools like Critical Path Analysis, Project Estimating, Time Reporting, Gantt Analysis, EVA, Milestones and tasks behind schedule or over budget notifications. VEMS allows integration with other commonly used project planning tools by allowing users to import and export files from MPX, TXT, RTF or .CSV and other standard formats. By contrast with two dimension time management systems which provide the user with the 'scheduled' time as one dimension and the 'actual' time as the other dimension, VEMS provides the third time management dimension which is the 'estimated' time for a project or project task. The estimation of project completion date is done by the system, based on the percentage of completion of all tasks and the time reports entered by the users allocated to tasks, which differentiate VEMS from the two dimension time management systems. Based on the estimated finish time high level management reports are delivered based on the projection of the projects behavior in the future. The first two dimensions are covered by the basic level management systems. VEMS covers especially the third dimension which is one of the key features of the high level management systems in order to provide prediction to the executive management. This way the executives can prevent issues instead and detecting and solving them. The change management feature provides means to manage project changes to give the user better control of project related 'changes'. It allows the user to initiate and record all 'change' requests, to assign responsibilities and to track the change through its lifecycle. It also allows seeing the impact of the changes in time, cost, and quality for any given project. Another feture would be telelerning via iButton technology. The system is web oriented so that it is easy to be interfaced with laboratory equipment.

References

1. European Commission Europe Aid Co-operation office, General Affairs Evaluation, Manual - Project Cycle Management (2002)
2. European Commission Europe Aid Co-operation office, General Affairs Evaluation. Hand Book - Project Cycle Management (2002)
3. PMBoK
4. McConnell, S.: Code Complete Microsoft Press ISBN 1-55615-484-4
5. McConnell, S.: Rapid Development Microsoft Press ISBN 1-55615-900-5
6. http://vems.cst-us.com
7. http://www.cordis.lu/ist/projects.htm
8. Regulation of the European Parliament and of the Council concerning the rules for the participation of undertakings, research centers and universities in the implementation of the European Community sixth framework programme (2002-2006), http://www.cordis.lu/fp6/find-doc.htm

PTF Model for Evaluating the Relationships between the VAC and Antimicrobial Resistance in Human Communities

Irina Codita[1] and Stefan Pusca[2]

[1] Cantacuzino National Institute of Research-Development for Microbiology and Immunology, Bucharest, Romania
[2] Politehnica University, Department of Physics, Bucharest, Romania

Abstract. This paper shows the possibility to create the basis of a computational (mathematical) model for the relationship between the volume of antimicrobial consumption (VAC) and the frequency of antimicrobial resistance in the human communities, based on an analogy with oscillations and wavelets. A heuristic algorithm for generating asymmetrical practical test functions (ie PTF) using MATLAB procedures was elaborated. Based on the fact that differential equations can generate only functions similar to test functions (defined as practical test functions), the invariance general properties suitable for generating symmetrical pulses as related to the middle of the working interval are presented.

Then some possibilities for obtaining asymmetrical pulses as related to this middle of the working interval using the derivative of such symmetrical pulse are studied, for certain differential equations corresponding to second order systems (with unity-step input and for an input represented by a Gaussian pulse). Finally it is shown that we can reach an oscillating system by joining such working intervals and restoring the initial null conditions for a second order system, in an adequate manner.

1 Introduction

After the declaration of the 1996 World Health Assembly appreciating the antimicrobial resistance as a “global threat”, the Invitational EU Conference on The Microbial Threat held in Copenhagen, Denmark, 9-10 September 1998, delivered the renowned “Copenhagen recommendations” for antimicrobial resistance containment.

Most of the sense of these recommendations is to strengthen surveillance of resistance in order to curb it by quickly and appropriately modeling the antimicrobials use.

Though there is generally recognized that the cumulated effect of increasing antimicrobial consumption is raising the antimicrobial resistant organisms’ proportion, establishing a precise quantitative relationship between the frequency of resistance and antimicrobials consumption proved difficult, both because of the lack of long lasting antimicrobial resistance/antimicrobials consumption data bases and of theoretical models. Some general pattern-s were yet described, e.g.: a. typically a long period of very low-level resistance preceding a phase of rapid increase in frequency and a slow approach to an equilibrium level of < 100%; b. the sigmoid shape of observed longitudinal changes, which theory predicts under a constant selective pressure [18].

The consensus accepted unit to measure the selective pressure exerted by the antimicrobial use is the number of DDDs (Defined Daily Doses)/1,000 inhabitants or individuals.

However, resistance phenomenon is not a linear and/or similar one in every organism, even in the presence of antimicrobials pressure.

There are microorganisms known as adapting to antimicrobials by point mutations (e.g. *Mycobacterium tuberculosis*) and there are others which are surviving by transferring each others fragments of genetic material coding for resistance factors (e.g. *Staphylococcus aureus*, gram negative enteric bacteria, enterococci etc.).

Microorganisms' population genetics studies were addressed to help understanding the fine dynamics of general epidemiological trends.

On the other hand, there are changes at the human population level, leading to transmission/colonization with resistant or susceptible microorganisms, pending on the antimicrobial treatments. For example, a patient colonized with both susceptible and resistant microbial genotypes (strains) treated with an antimicrobial which was not efficient, will further transmit to other individuals the resistant strain selected by the antimicrobial treatment etc. We may consequently conclude on having both direct and indirect effects of antimicrobial use.

The types of resistance mechanisms have implications for the choice of antimicrobial therapy and the evaluation of strategies to minimize resistance and "adopting the individual and population level perspective informs therapeutic decision-making, clinical study design and public policy" [19]. Choices have to be done evidence based, for individual cases, but antimicrobial use policies are needed too at local, national and regional level.

What becomes clear even after sketchily describing the antimicrobial resistance selection/spreading complexity is the fact that we are not dealing with ergotic systems.

To study these kinds of phenomena you need adequate mathematical models.

The analysis of signals on limited time intervals requires often the use of adequate mathematical models able to generate alternating function. Using undumped differential equations of second order, able to generate signals with a certain angular frequency for obtaining sine functions is a well-known option. But for obtaining pulses limited on certain time intervals some specific models must be set up. An alternative is represented by the use of test-functions, but ideal test functions can not be generated by a differential equation of evolution [1]. On the other side, a propagation phenomenon for an ideal test function can not be taken into consideration as in [2], because we are looking for causal pulses, generated in a rigorous manner by an equation of evolution. This implies the use of practical test-functions (functions which possess a limited number of derivatives equal to zero at the limits of the working interval and which can be solutions of differential equations). Consequently we must study invariance properties of such equations, so as the output to be represented by an asymmetrical function as related to the middle of the working period. If we consider as working interval the time interval (-1; 1), then the middle of the interval would be the origin, and the condition for the output f(t) being asymmetrical corresponds to the condition.

$$f(-t) = -f(t) \tag{1}$$

Aiming to obtain such a function on the time interval (-1; 1), we have to begin by studying equations able to generate a symmetrical function g on this time interval [3] so as to find a method for translating some of their properties to asymmetrical functions; finally we must use Runge-Kutta equations ([4]) for studying the properties of the mathematical models obtained. We are looking for controlled oscillations on a limited time interval (unlike unstable oscillations for second order difference systems [5]).

A first attempt would be the use of the signal which is integrated for sampling electronic or optoelectronic signals in a robust manner - using oscillating second order systems working on a period [6],. The filtering possibilities of such systems (as low pass filters) were presented in [7]. This would lead to a sine or cosine function, with possibilities of joining together such working intervals for obtaining a controlled oscillation extended in time. Yet we are looking for general differential equations able to generate asymmetrical pulses of different shapes (not only sine or cosine functions). We may extend our analyze at wavelets corresponding to PDE [8] or to equations able to generate wavelets represented by solitary waves [9].

Both previously mentioned aspects could be joined together if we are looking for functions similar to test-functions having a shape similar to wavelets. As one may notice by studying [3], practical test-functions of second order possess a derivative with null initial and final values. By analyzing its mathematical expression on the whole working interval (-1; 1), we may notice that this derivative is an asymmetrical function as related to the middle of the working interval (considered as origin), while the symmetry of g function implies that its slope is asymmetrical as related to the origin (the same modulus and opposite sign). So we have to analyze the differential equations able to generate symmetrical functions g and to study the shape of their derivatives for different input functions.

2 Asymmetrical Pulses Obtained as Derivatives of Symmetrical Functions

As it is known, a test-function on [a, b] is a C1 function on R which is nonzero on (a; b) and zero elsewhere. For example, the bump-like function

$$\varphi_a(\tau) = \begin{cases} \exp\left(\frac{1}{\tau^2-1}\right) & \text{if} \quad \tau \in (-1,1) \\ 0 \text{ otherwise} \end{cases} \tag{2}$$

is a test-function on [-1; 1].

On the other side, test-functions as the bump-like function:

$$\varphi_b(\tau) = \begin{cases} \exp\left(\frac{0.1}{\tau^2-1}\right) & \text{if} \quad \tau \in (-1,1) \\ 0 \text{ otherwise} \end{cases} \tag{3}$$

are almost equal to a constant value for 2/3 of the working period (similar to a step-function).

Such functions can not be generated by differential equations of evolution; however, we can use differential equations able to generate a practical test-function on (-1; 1) (a Cn nonzero function on (-1; 1) which satisfies the boundary conditions f(k)(a) = f(k)(b) = 0 for k = 0; 1; :::; n and can be a solution of an initial value problem on this interval).

The first and second derivatives of 'a are

$$\varphi_a^{(1)}(\tau) = \frac{-2\tau}{(\tau^2-1)^2}\exp\left(\frac{1}{\tau^2-1}\right) \tag{4}$$

$$\varphi_a^{(2)}(\tau) = \frac{6\tau^4-2}{(\tau^2-1)^4}\exp\left(\frac{1}{\tau^2-1}\right) \tag{5}$$

Considering the corresponding differential equation

$$f^{(1)} = \frac{-2\tau}{(\tau^2-1)^2}f \tag{6}$$

with initial values considered at f0 = -0:99 as

$$f_0 = \exp\left(\frac{1}{0.99^2-1}\right) \tag{7}$$

it results a function f symmetrical as related to the middle of the working interval. In the same way the correspondence is:

$$\varphi_a^{(2)} = \frac{6\tau^4-2}{(\tau^2-1)^4}\varphi_a \tag{8}$$

By considering the corresponding differential equation:

$$f^{(2)} = \frac{6\tau^4-2}{(\tau^2-1)^4}f \tag{9}$$

with initial values considered at f0 = -0:99 as

$$f_0 = \exp\left(\frac{1}{0.99^2-1}\right) \tag{10}$$

$$f_0^{(1)} = \left[2\frac{0.99}{(0.99^2-1)^2}\right]\exp\left(\frac{1}{0.99^2-1}\right) \tag{11}$$

it results also a function f symmetrical as related to the middle of the working interval. In a similar way, for the function 'b (f) we obtain the correspondence:

$$\varphi_b^{(2)} = \frac{0.6\tau^4 - 0.36\tau^2 - 0.2}{(\tau^2 - 1)^4}\varphi_b \tag{12}$$

By considering the corresponding differential equation

$$f^{(2)} = \frac{0.6\tau^4 - 0.36\tau^2 - 0.2}{(\tau^2 - 1)^4}f \tag{13}$$

with initial values considered at f0 = -0:99 as

$$f_0 = \exp\left(\frac{0.1}{0.99^2 - 1}\right) \tag{14}$$

$$f_0^{(1)} = \left[0.2\frac{0.99}{(0.99^2 - 1)^2}\right]\exp\left(\frac{0.1}{0.99^2 - 1}\right) \tag{15}$$

we obtain a function f symmetrical as related to the middle of the working interval I.

The shape of these outputs offers also the possibility of joining together such time intervals and the corresponding asymmetrical pulses so as to obtain a controlled oscillation. While at the beginning and at the end of each working interval the state-variables of the differential system are approximately equal to zero it would be quite easy to adjust the final values of these variables for a working interval to the initial values of these variables for the next working interval; thus the cycle can continue in a controlled manner.

We must point the fact that an asymmetrical pulse represents in fact a test function for the derivative of an input signal; by multiplying an input signal with an asymmetrical pulse and by integrating the resulting function on the working interval, we obtain a result proportional to the slope of the input signal (as it can be easily checked). Thus a possible application would be a faster estimation of acceleration by multiplying the input signal corresponding to velocity (a robust method which is faster than the method presented in [10], based on an estimation performed over two working periods). Another application can be represented by phase-detection; by multiplying the alternating input signal with an asymmetrical function and integrating the resulting function we obtain a result proportional to the amplitude of an input sine function (a more robust method than the one presented in [11], where the input function is processed by a nonlinear second order system).

Finally we have to study the output generated by a second order differential equation able to generate an asymmetrical output for an input represented by a very short pulse, so as to check its stability at such kind of disturbances. For this, we consider the differential equation:

$$f^{(2)} = \frac{0.6\tau^4 - 0.36\tau^2 - 0.2}{(\tau^2 - 1)^4}f + 0.1\exp\left[-\frac{(\tau + 0.9)^2}{0.01^2}\right] \tag{16}$$

with initial null conditions (the external pulse being represented by a short Gaussian pulse received at the moment of time tp = -0:9). One may notice that a major influence appears at the end of the working interval, after a time interval of about 1.8 units. Thus the final output pulse can be considered as an acausal pulse for an external observer studying the input and the output of the system on the time interval (0; 1), for example.

Unlike aspects connected with acausal traveling waves as possible solutions of the wave equation presented in [12] (where are no sources inside a certain string beginning to move from initial null conditions), the acausal pulse generated by the previous equation appears with almost null conditions existing for the state variables of a unique system (a single point). The term almost null conditions implies the use of a multiscale analysis of phenomena [13]. The study have to be completed by searching invariance properties for a good determination of inner structure of material [14], for a corresponding computer program able to perform an accurate estimation of the corresponding parameters [15], taking also into consideration phenomena appearing for thin-walled materials subject to external pulses [16].

For rejecting the inuence of such short Gaussian pulses, we must use some low-pass filters for delaying the moment of time when such short pulses are received by the processing system with about 0.2 units, so as no effect upon the output to be noticed any more (the whole working interval is about 2 units).

The computational signification of the influence of the short Gaussian pulse considered as an acausal pulse can be interpreted in terms of the superinfection mechanisms, which may result in coexistence of both types of strains (resistant and susceptible strains of the same microorganism) isolated from one person with resistance at an equilibrium frequency lower than 100%.

3 Conclusions

This paper presents some methods for generating asymmetrical practical test functions using MATLAB procedures (based on Runge-Kutta equations). Based on the fact that differential equations are able to generate only functions similar to test functions (defined as practical test functions), the invariance general properties suitable for generating symmetrical pulses as related to the middle of the working interval are presented. Then some possibilities for obtaining asymmetrical pulses as related to the middle of this interval using the derivative of such symmetrical pulse are studied, for certain differential equations corresponding to second order systems. Finally it is shown that we can obtain an oscillating system by joining such working intervals and restoring the initial null conditions for a second order system, in an adequate manner. Results of using such asymmetrical functions for analyzing the relationship between the volume of antimicrobial consumption in human communities and the frequency of resistance (similar to [17] will be published in the future.

References

1. Toma, C.: An extension of the notion of observability at filtering and sampling devices. In: Proceedings of the International Symposium on Signals, Circuits and Systems Iasi SCS 2001, Romania, pp. 233–236 (2001)

2. Toma, C.: The possibility of appearing acausal pulses as solutions of the wave equation. The Hyperion Scientific Journal 41, 25–28 (2004)
3. Toma, G.: Practical test-functions generated by computer algorithms. In: Gervasi, O., Gavrilova, M.L., Kumar, V., Laganá, A., Lee, H.P., Mun, Y., Taniar, D., Tan, C.J.K. (eds.) ICCSA 2005. LNCS, vol. 3482, pp. 576–584. Springer, Heidelberg (2005)
4. Kulikov, G.: An advanced version of the local-global step-size control for Runge-Kutta methods applied to index 1 differential algebraic systems. In: Bubak, M., van Albada, G.D., Sloot, P.M.A., Dongarra, J. (eds.) ICCS 2004. LNCS, vol. 3037, pp. 565–569. Springer, Heidelberg (2004)
5. Zhang, Z., Ping, B., Dong, W.: Oscillation of unstable type second order nonlinear difference equation. Korean J. Computer and Appl. Math. 91, 87–99 (2002)
6. Toma, C.: The necessity for using oscillating systems for sampling optoelectronic signals. Bulgarian Journal of Physics 27, Suppl. 2, 187–190 (2000)
7. Toma, C.: Filtering possibilities based on oscillating systems for optoelectronic signals. In: SPIE Proceedings, vol. 4430, pp. 842–845 (2001)
8. Cattani, C.: Harmonic Wavelets towards Solution of Nonlinear PDE. Computers and Mathematics with Applications 50, 1191–1210 (2005)
9. Rushchitsky, J.J., Cattani, C., Terletskaya, E.V.: Wavelet Analysis of the evolution of a solitary wave in a composite material. International Applied Mechanics 40(3), 311–318 (2004)
10. Sterian, A., Toma, C.: Filtering possibilities for processing optoelectronic current for acceleration measurements. In: SPIE Proceedings, vol. 4827, pp. 403–408 (2002)
11. Sterian, A., Toma, C.: Phase detection for vibration measurements based on test functions. In: SPIE Proceedings, vol. 5503, pp. 164–168 (2004)
12. Toma, C.: Equations with partial derivatives and differential equations used for simulating acausal pulses. In: International Conference Physics and Control Physcon 2003, August 20-22, pp. 1178–1183. Sankt-Petersburg, Russia (2003)
13. Cattani, C.: Multiscale Analysis of Wave Propagation in Composite Materials. Mathematical Modelling and Analysis 84, 267–282 (2003)
14. Paun, V.P.: A Model for the Accurate Determination of Crystaline Network Parameters. Revue Roumaine de Chimie 54(4), 335–336 (2003)
15. Paun, V.P.: Computer programme for the determination of the crystalline network parameters. Revue Roumaine de Chimie 49(1), 85–92 (2004)
16. Paun, V.P.: A creep collapse model for thin-walled Zircolay-4 tubes. Revue Roumaine de Chimie 48(11), 903–906 (2003)
17. Cattani, C.: Harmonic Wavelet Solutions of the Schroedinger Equation. International Journal of Fluid Mechanics Research 5, 1–10 (2003)
18. Austin, D.J., Kristinsson, K.G., Anderson, R.M.: The relationship between the volume of antimicrobial consumption in human communities and the frequency of resistance. Proc. Natl. Acd. Sci. USA, 1152–1156 (1996)
19. Lipsitch, M., Samore, M.H.: Antimicrobial Use and Antimicrobial Resistance: A Population Perspective. Emerg. Inf. Diseases 8(4) (2002)

Author Index

Abtholuddin, M. Faris 392
Albrecht, A. 167
Albrecht, Andreas A. 128
Alhajj, Reda 102
Alshalalfa, Mohammed 102
Althaus, Ernst 489
Anto, M.A. Lijo 509
Antoniou, Pavlos 446

Bahn, Sabine 362
Baldock, Richard 347
Bernhart, Stephan H. 114
Bhattacharyya, Balaram 221
Bhattacharyya, Ramkishore 221
Boeva, Veselka 193
Bontempi, Gianluca 273
Bošnački, Dragan 303
Breit, Timo 243
Brilli, Matteo 207

Canzar, Stefan 489
Carapelli, Antonio 404
Cattani, Carlo 528, 538
Caulfield, Mark J. 377
Codita, Irina 587
Coelho, Luís Pedro 235

Dalziel, Ben 16
Dobson, Richard J.B. 377
Dong, Qiwen 74
Dou, Yong 43
Duval, Béatrice 499

Elloumi, Mourad 427, 479

Fani, Renato 207
Federico, Maria 456
Fionda, Valeria 331
Fisher, Susan 16
Flouri, Tomáš 436
Frati, Francesco 404
Furletova, Eugenia 466

Gambin, Anna 466
Gan, Xiangchao 128, 167
Ghosh, Indira 509
Gopakumar, G. 509
Gormley, Matthew 16
Gupta, Arvind 180

Hadj Khodabakhshi, Alireza 180
Halachev, Mihail 58, 87
Hao, Jin-Kao 499
Hernandez Hernandez, Jose Crispin 499
Hernandez-Rosales, Maribel 114
Hofacker, Ivo L. 114

Iliopoulos, Costas S. 436, 446

Jayasekera, Inuka 446

Kalsi, Petri 417
Kambol, Roziah 392
Kapsokalivas, Leonidas 128, 167
Kohl, Jochen 258
Kontos, Kevin 273
Kucherov, Gregory 466
Kurbatova, Natalja 154

Lasota, Sławomir 466
Lee, Timothy 140
Leweke, F. Markus 362
Lian, Chen Na 87
Lin, Lei 74
Lió, Pietro 207, 362, 404
Liu, Bin 74

Mainz, Dominic 258
Mainz, Indra 258
Maňuch, Ján 180
Marin, Octavian 556
Marshall, Roger 517
Martin, Christian 1
Mein, Charles A. 377
Milanesi, Luciano 404
Mocanu, Elena 546
Moisa, Trandafir 581
Mokaddem, Ahmed 479
Morarescu, Cristian 581
Mückstein, Ulrike 114

Munroe, Patricia B. 377
Murphy, Robert F. 235

Nair, Achuthsankar S. 509
Nardi, Francesco 404
Nattkemper, Tim W. 1
Nicolae, Vlăduţ-Bogdan 563
Nijholt, Anton 243
Ninulescu, Valerică 556, 563
Noé, Laurent 466

Palopoli, Luigi 331
Panni, Simona 331
Paulsen, Ingo 258
Peltola, Hannu 417
Phuong, N.M. 32
Pisanti, Nadia 456
Pitkänen, Esa 288
Pizzuti, Clara 318
Pusca, Stefan 571, 587

Rafiey, Arash 180
Rahman, M. Sohel 436, 446
Rantanen, Ari 288
Rauwerda, Han 243
Rombo, Simona E. 318, 331
Rousu, Juho 288
Roytberg, Mikhail 466

Saqi, Mansoor A.S. 377
Scalia, Massimo 538
Schwarz, Emanuel 362
Shiri, Nematollaah 58, 87
Singh, Rahul 16, 140
Stadler, Peter F. 114
Steinhöfel, Kathleen 128, 167
Sterian, Andreea 563
Sterian, Paul 546, 556
Szczurek, Ewa 466

Tafer, Hakim 114
Tahri, Hatem 427
Tarhio, Jorma 417
Toma, Theodora 571
Tsiporkova, Elena 193
Tuan, H.D. 32

Ukkonen, Esko 288

Vagner, Ladislav 436
van der Vet, Paul 243
van Hemert, Jano 347
van der Wath, Elizabeth 404
van der Wath, Richard C. 404
Viksna, Juris 154
Vogel, Jörg 114
von Haeseler, Arndt 258
Voráček, Michal 436

Wagemans, Stefan 303
Wang, Xiaolong 74
Wang, Xuan 74
Wassink, Ingo 243
Weller, Katrin 258

Xia, Fei 43
Xu, Jiaqing 43

Yang, Hui 16

Zantema, Hans 303