Madhu Chetty
Alioune Ngom
Shandar Ahmad (Ed

LNBI 5265

Pattern R
in Bioinfo

Third IAPR International Co
Melbourne, Australia, Octo
Proceedings

Lecture Notes in Bioinformatics 5265

Edited by S. Istrail, P. Pevzner, and M. Waterman

Subseries of Lecture Notes in Computer Science

Madhu Chetty Alioune Ngom
Shandar Ahmad (Eds.)

Pattern Recognition in Bioinformatics

Third IAPR International Conference, PRIB 2008
Melbourne, Australia, October 15-17, 2008
Proceedings

Springer

Volume Editors

Madhu Chetty
Faculty of Information Technology
Monash University, Gippsland Campus
VIC-3842, Australia
E-mail: madhu.chetty@infotech.monash.edu.au

Alioune Ngom
School of Computer Science
University of Windsor
Windsor, Ontario, Canada
E-mail: angom@cs.uwindsor.ca

Shandar Ahmad
National Institute of Biomedical Innovation
Saito Asagi, Ibaraki-shi, Osaka, Japan
E-mail: shandar@nibio.go.jp

Library of Congress Control Number: Applied for

CR Subject Classification (1998): I.5, J.3, I.2, H.3, F.2.2

LNCS Sublibrary: SL 8 – Bioinformatics

ISSN 0302-9743
ISBN-10 3-540-88434-3 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-88434-7 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12530489 06/3180 5 4 3 2 1 0

Preface

In the post-genomic era, a holistic understanding of biological systems and processes, in all their complexity, is critical in comprehending nature's choreography of life. As a result, bioinformatics involving its two main disciplines, namely, the life sciences and the computational sciences, is fast becoming a very promising multidisciplinary research field. With the ever-increasing application of large-scale high-throughput technologies, such as gene or protein microarrays and mass spectrometry methods, the enormous body of information is growing rapidly. Bioinformaticians are posed with a large number of difficult problems to solve, arising not only due to the complexities in acquiring the molecular information but also due to the size and nature of the generated data sets and/or the limitations of the algorithms required for analyzing these data. Although the field of bioinformatics is still in its embryonic stage, the recent advancements in computational and information-theoretic techniques are enabling us to conduct various in silico testing and screening of many lab-based experiments before these are actually performed in vitro or in vivo. These in silico investigations are providing new insights for interpretation and establishing a new direction for a deeper understanding. Among the various advanced computational methods currently being applied to such studies, the *pattern recognition* techniques are mostly found to be at the core of the whole discovery process for apprehending the underlying biological knowledge. Thus, we can safely surmise that the ongoing bioinformatics *revolution* may, in future, inevitably play a major role in many aspects of medical practice and/or the discipline of life sciences.

The aim of the Pattern Recognition in Bioinformatics (PRIB) conference is to provide an opportunity for academia, researchers, scientists and industry professionals to present their latest research in pattern recognition and computational intelligence-based techniques applied to problems in bioinformatics and computational biology. It also provides them with an excellent forum to interact with each other and share experiences. The conference is organized jointly by Monash University, Australia, and the IAPR (International Association for Pattern Recognition) Bioinformatics Technical Committee (TC-20).

This volume presents the proceedings of the Third IAPR International Conference on Pattern Recognition in Bioinformatics (PRIB 2008), held in Melbourne, October 15–17, 2008. It includes 39 technical contributions that were selected by the International Program Committee from 121 submissions. Each of these rigorously reviewed papers was presented orally at PRIB 2008. The proceedings consists of six parts:

Part 1 Protein: Structure, Function, and Interaction
Part 2 Learning, Classification, and Clustering
Part 3 Bio-Molecular Networks and Pathways Analysis
Part 4 Microarray and Gene Expression Analysis

Part 5 Data Mining and Knowledge Discovery
Part 6 Applications of High-Performance Computing

Part 1 of the proceedings contains eight chapters on *Protein: Structure, Function, and Interaction*. Gromiha et al. propose a method based on a decision tree for discriminating the stabilizing and destabilizing mutants and predicting protein stability changes upon single-point mutations. The chapter also includes methods developed for discriminating thermophilic proteins from mesophilic ones. In the next chapter by Kumar et al., a new approach to locating the occurrences of user-defined motifs in a specified order in large proteins and in nucleotide sequence databases is proposed. Bauer et al. explore the nature of post-translation modifications (SUMOylation) using non-local sequence and structural properties, including secondary structure, solvent accessibility and evolutionary profiles. Hoque et al. combine a genetic algorithm with depth-first search for the solution of the protein structure prediction problem. Lonquety et al. present a stability-based analysis of the protein-folding nucleus to increase the recall and precision of two well-known protein mutant stability prediction methods. Kato et al. report a dynamic programming algorithm for an up-down class of antiparallel protein β-sheet, which can also be extended for more general classes of β-sheets. In Li et al., the concept of multi-scale glide zoom window feature extraction is used for predicting protein homo-oligomers. Koizumi et al. propose a method of searching for and comparing concave structures in protein-binding sites.

Part 2 of the proceedings contains seven chapters on *Learning, Classification, and Clustering*. Yang et al. propose an hybrid system for analyzing high-dimensional mass spectrometry data. Medvés et al. propose a modified Markov clustering algorithm for an efficient clustering of large protein sequence databases, based on a previously evaluated sequence similarity criteria. Stiglic et al. present a classification ensemble of decision trees called Rotation Forest and evaluate its classification performance on small subsets of ranked genes for 14 genomic and proteomic classification problems. Al Seesi et al. describe a new inference algorithm, based on tree adjoining grammars, for RNA pseudoknot structure identification. Mundra et al. propose to use support vector points for computation of t-scores for gene ranking. Anand et al. consider two sets of features based on DNA sequences and their physicochemical properties and applied a one-versus-all support vector machine with class-wise optimized features to identify transcription factor family-specific features in DNA sequences. Ji et al. present a novel protein classifier, the gapped Markov Chain with Support Vector Machine, that models the structure of a protein sequence by measuring the transition probabilities between pairs of amino acids.

Part 3 of the proceedings contains six chapters on *Bio-Molecular Networks and Pathways Analysis*. Zhao et al. propose a novel discriminative method for predicting domain–domain interactions in protein pairs by making use of interacting and non-interacting protein pairs, which improves the prediction reliability. Jancura et al. develop an algorithm for dividing protein–protein interaction networks that combines the graph theoretical property of articulation with a

biological property of orthology. Ram et al. demonstrate the application of a Markov blanket learning algorithm to gene regulatory networks, enhanced further by application of a proposed constraint logic minimization technique. Chaturvedi et al. model time delayed interactions in gene regulatory networks using a skip-chain model which finds missing edges between non-consecutive time points based on protein-protein interaction networks. Zhou et al. propose a new pattern recognition technique to help represent metabolic networks as weighted vectors. Ram et al. present an approach for synthetically generating gene regulatory networks using causal relationships.

Part 4 of the proceedings contains seven chapters on *Microarray and Gene Expression Analysis*. Huerta et al. introduce a new wrapper approach to the difficult task of microarray data gene selection, where a genetic algorithm is combined with Fisher's linear discriminant analysis. Wang et al. present a new heuristic approach for finding near-minimal non-unique probe sets for oligonucleotide microarray experiments. Using a well-known yeast cell cycle data set, Pittelkow et al. compare a method being used for finding genes following a periodic time series pattern with a method for finding genes having a different phase pattern during the cell cycle. Bedo explores the design problem of selecting a small subset of clones from a large pool for creation of a microarray plate. Nagarajan et al. use two distinct approaches, namely, the classical order zero-crossing count and the Lempel-Ziv complexity, in identifying non-random patterns from temporal gene expression profiles. Ooi et al. propose to determine the theoretical basis for the concept of differential prioritization through mathematical analyses of the characteristics of predictor sets found using different values of the degree of differential prioritization from realistic but toy datasets. Luo et al. propose a weighted top scoring pair method for gene selection and classification.

Part 5 of the proceedings contains seven chapters on *Data Mining and Knowledge Discovery*. Kasturi et al. present an algorithm to identify statistically significant and conserved discriminative motifs that distinguish between gene expression clusters. McGarry et al. describe methods to develop a reliable, automated method of detecting abnormal metabolite profiles from urinary organic acids, which can be used as GC-MS biomarkers. Girão et al. have applied multi-relational data mining methods with hidden Markov models and a Viterbi algorithm to mine tetratricopeptide repeat, pentatricopeptide and half-a-tetratricopeptide repeat in genomes of pathogenic protozoa Leishmania. Sehgal et al. present an enhanced heuristic non-parametric collateral missing value imputation algorithm which uses collateral missing value estimation as its core estimator and a heuristic non-parametric strategy to compute the optimal number of estimator genes to exploit optimally both local and global correlations. Han et al. develop a non-negative principal component analysis algorithm and propose a non-negative principal component analysis-based support vector machine algorithm with sparse coding in the cancer molecular pattern analysis of proteomics data. Macintyre et al. have developed a novel clustering algorithm which incorporates functional gene information from gene ontology into the clustering

process, resulting in more biologically meaningfull clusters. Meydan et al. try evolutionary classification methods for selecting the important classifier genes in hexachlorobenzene toxicity using microarray data.

Part 6 of the proceedings contains four chapters on *Applications of High-Performance Computing*. Stamatakis et al. address parallelism issues via a thorough performance study by example of a widely used bioinformatics application for large-scale phylogenetic inference under the maximum likelihood criterion. Schröder et al. present an enhanced version of an existing DNA motif search algorithm tailored to fit on a massively parallel machine. Chen et al. present a novel approach to accelerate motif discovery based on commodity graphics hardware. Zhang et al. demonstrate how the PlayStation 3, powered by the Cell Broadband Engine, can be used as an efficient computational platform to accelerate the popular BLASTP algorithm.

Many have contributed directly or indirectly toward the organization and success of PRIB 2008 conference. We would like to thank all individuals and institutions, especially the authors for submitting the papers and the sponsors for generously providing financial support for the conference. We are very grateful to IAPR for the sponsorship and the IAPR Technical Committee (TC-20) on Pattern Recognition for Bioinformatics for their support and advice. Our gratitude goes to the Faculty of Information Technology, Monash University, Australia, and also to the Gippsland campus, Monash University, Australia, for supporting the conference in many ways.

We would like to express our gratitude to all PRIB 2008 International Program Committee members for their objective and thorough reviews of the submitted papers. We fully appreciate the PRIB 2008 Organizing Committee for their time, efforts, and excellent work. We would also like to thank Jagath Rajapakse, Program Co-chair, and Raj Acharya, General Co-chair, for their continuous support and guidance. We sincerely thank Shyh Wei Teng, Local Organization Chair, for his relentless work in managing various operational issues and finance matters related to the conference organization. We thank Dieter Bulach for organizing the conference sponsorship and the Publication Co-chair, Sy Loi Ho, for his hard work in getting the proceedings ready on time. We are also grateful to Tina Bradshaw, PRIB 2008 secretary, for coordinating all the logistics of the workshop and to Margot Schuhmacher for meticulously maintaining the PRIB 2008 conference website.

Last but not least, we wish to convey our sincere thanks to Springer for providing excellent professional support in preparing this volume.

October 2008

Madhu Chetty
Alioune Ngom
Shandar Ahmad

PRIB 2008 Organization

General Chair

Madhu Chetty	Monash University, Australia

General Co-chair

Raj Acharya	Pennsylvania State University, USA

Program Co-chairs

Alioune Ngom	University of Windsor, Canada
Jagath C. Rajapakse	Nanyang Technological University, Singapore

Technical Co-chairs

Elena Marchiori	Vrije Universiteit, The Netherlands
Bertil Schmidt	NICTA, Australia

Short Paper Co-chairs

Shandar Ahmad	National Institute of Biomedical Innovation, Japan
Ponraj Prabhakaran	Duke University, USA
Michael Grominha	Computational Biology Research Centre, Japan

Special Session Chair

Alioune Ngom	University of Windsor, Canada

Local Organization Chair

Shyh Wei Teng	Monash University, Australia

Publicity Co-chairs

Shandar Ahmad	National Institute of Biomedical Innovation, Japan
Phoebe Chen	Deakin University, Australia
Mariofanna Milanova	University of Arkansas at Little Rock, USA
Md. Tamjidul Hoque	Griffith University, Autsralia

Publication Co-chairs

Sy Loi Ho	Nanyang Technological University, Singapore
Girija Chetty	University of Canberra, Australia

Sponsorship Chair

Dieter Bulach CSIRO, Australia

Finance Manager

Narelle Wilkins Monash University, Australia

Secretariat

Tina Bradshaw Monash University, Australia

Webmaster

Margot Schuhmacher Monash University, Australia

Technical Manager

Glen Pringle Monash University, Australia

Program Committee

Raj Acharya	Pennsylvania State University, USA
Shandar Ahmad	National Institute of Biomedical Innovatation, Japan
Tatsuya Akutsu	Kyoto University, Japan
James Bailey	University of Melbourne, Australia
Tim Bailey	University of Queensland, Australia
Sanghamitra Bandyopadhyay	Indian Statistical Institute, India
Wolfgang Banzhaf	Memorial University of Newfoundland, Canada
Justin Bedo	The Australian National University, Australia
Nitin Bhardwaj	University of Illinois at Chicago, USA
Mikael Boden	University of Queensland, Australia
Sebastian Boecker	Friedrich-Schiller-Universitaet Jena, Germany
Guillaume Bourque	Genome Institute of Singapore, Singapore
Roelof Brouwer	University of Stellenbosch, South Africa
Vladimir Brusic	Dana-Farber Cancer Institute, USA
Shakuntala Bulusu	DNA Research Centre, India
Conrad Burden	Australian National University, Australia
Yu-Dong Cai	Shanghai Institutes for Biological Science, China
Frederic Cazals	INRIA Sophia Antipoh's, France
Francis Chin	The University of Hong Kong, China
Peter Clote	Boston College, USA
Michael Cohen	Griffith University, Australia
Yun Cui	Nanyang Technological University, Singapore
Antoine Danchin	Institut Pasteur, France

Suash Deb	National Institute of Science and Technology, India
Omid Dehzangi	Nanyang Technological University, Singapore
Yong-Sheng Ding	Donghua University, China
M. Michael Gromiha	Computational Biology Research Center, Japan
Nadia El-Mabrouk	University of Montreal, Canada
Alexandru Floares	Oncological Institute Cluj-Napoca, Romania
Gary B. Fogel	Natural Selection Inc., USA
Mehdi Ghayoumi	Azad University, Iran
Prashant Goley	University Nanded Maharashtra, India
Iqbal Gondal	Monash University, Australia
Robin Gras	University of Windsor, Canada
Saman K. Halgamuge	University of Melbourne, Australia
Xiaoxu Han	Eastern Michigan University, USA
Jin-Kao Hao	University of Angers, France
Jaap Heringa	Vrije Universiteit, Belgium
Sy Loi Ho	Nanyang Technological University, Singapore
Md Tamjidul Hoque	Griffith University, Australia
Seiya Imoto	University of Tokyo, Japan
Shunsuke Inenaga	Kyushu University, Japan
R. Krishna Murthy Karuturi	Genome Institute of Singapore, Singapore
Nawaz Khan	Middlesex University, UK
Amit Kumar	DNA Research Centre, India
Lukasz Kurgan	University of Alberta, Canada
Adam Kowalczyk	NICTA, Australia
Vlad Kuznetsov	Genome Institute of Singapore, Singapore
Zoe Lacroix	Arizona State University, USA
Hon Wai Leong	National University of Singapore, Singapore
Jinyan Li	Nanyang Technological University
Vladimir Likic	University of Melbourne, Australia
Feng Lin	Nanyang Technological University, Singapore
Tsun-Chen Lin	Dahan Institute of Technology, Taiwan
Frederique Lisacek	Swiss Institute of Bioinformatics, Switzerland
Feng Liu	Vrije Universiteit, Belgium
Lifang Liu	Xi'an University, China
Weiguo Liu	National Technological University, Singapore
Geoff Macintyre	Victorian Research Lab, Australia
Niranjan Mahesan	University of Sheffield, UK
Veli Makinen	University of Helsinki, Finland
Elena Marchiori	Vrije Universiteit, The Netherlands
Hiroshi Matsuno	Yamaguchi University, Japan
Geoff McLachlan	University of Queensland, Australia
Aleksandar Milosavljevic	Baylor College of Medicine, USA
Martin Middendorf	University of Leipzig, Germany
Mariofanna Milanova	University of Arkansas at Little Rock, USA

Satoru Miyano	University of Tokyo, Japan
Krishna Mohan	NIBIO, Japan
Sukanta Mondal	NIBIO, Japan
Bernard Moret	Swiss Federal Institute of Technology, Switzerland
Narayana Nagesh	Centre for Cellular and Molecular Biology, India
See-Kiong Ng	Institute for Infocomm Research, Singapore
Alioune Ngom	University of Windsor, Ontario, Canda
Bing Niu	Shanghai University, China
Diana Oliveira	Universidade Estadual do Ceara, Brazil
Nikhil R. Pal	Indian Statistical Institute, India
Kuldip Paliwal	Griffith University, Australia
Mihail Popescu	University of Missouri-Columbia, USA
Jagath Rajapakse	Nanyang Technological University, Singapore
Ramesh Ram	Monash University, Australia
Alice Richardson	University of Canberra, Australia
Luis Rueda	Universidad de Concepción, Chile
Marie-France Sagot	INRIA, France
Meena Kishore Sakharkar	Nanyang Technological University, Singapore
Alexander Schliep	Max Planck Institute for Molecular Genetics, Germany
Bertil Schmidt	NICTA, Australia
Christian Schoenbach	Nanyang Technological University, Singapore
Jan Schröder	Christian Albrechts University of Kiel, Germany
Heiko Schroeder	RMIT University, Australia
Muhammad Shoaib Sehgal	University of Queensland, Australia
Mona Singh	Princeton, USA
Ingolf Sommer	Max Planck Institute for Informatics, Germany
Alexandros Stamatakis	Ludwig Maximilians University of Munich, Germany
Gregor Stiglic	University of Maribor, Slovenia
Wing Kin Sung	National University of Singapore, Singapore
Roberto Tagliaferri	Università di Salerno, Italy
Y-H. Taguchi	Chuo University, Japan
Shyh Wei Teng	Monash University, Australia
Qi-Chuan Tian	Taiyuan University of Science and Technology, China
Lokesh Tripathi	NIBIO, Japan
Lusheng Wang	City University of Hong Kong, China
Alan Wee-Chung Liew	Griffith University, Australia
Yanqing Zhang	Georgia State University, USA
Xingming Zhao	Shanghai University, China
Justin Zobel	NICTA, Australia

Table of Contents

Part I: Protein: Structure, Function, and Interaction

Sequence Based Prediction of Protein Mutant Stability and Discrimination of Thermophilic Proteins 1
M. Michael Gromiha, Liang-Tsung Huang, and Lien-Fu Lai

A Method to Find Sequentially Separated Motifsin Biological Sequences (SSMBS) 13
Chetan Kumar, Nishith Kumar, Sarani Rangarajan, Narayanaswamy Balakrishnan, and Kanagaraj Sekar

Predicting SUMOylation Sites 28
Denis C. Bauer, Fabian A. Buske, and Mikael Bodén

DFS Based Partial Pathways in GA for Protein Structure Prediction ... 41
Md Tamjidul Hoque, Madhu Chetty, Andrew Lewis, and Abdul Sattar

Evaluation of the Stability of Folding Nucleus upon Mutation 54
Mathieu Lonquety, Zoé Lacroix, and Jacques Chomilier

Prediction of Protein Beta-Sheets: Dynamic Programming Versus Grammatical Approach 66
Yuki Kato, Tatsuya Akutsu, and Hiroyuki Seki

Using Multi-scale Glide Zoom Window Feature Extraction Approach to Predict Protein Homo-oligomer Types 78
QiPeng Li, Shao Wu Zhang, and Quan Pan

Extraction of Binding Sites in Proteins by Searching for Similar Local Molecular Surfaces 87
Satoshi Koizumi, Keisuke Imada, Tomonobu Ozaki, and Takenao Ohkawa

Part II: Learning, Classification, and Clustering

A Clustering Based Hybrid System for Mass Spectrometry Data Analysis 98
Pengyi Yang and Zili Zhang

A Modified Markov Clustering Approach for Protein Sequence Clustering 110
Lehel Medvés, László Szilágyi, and Sándor M. Szilágyi

Feature Selection and Classification for Small Gene Sets 121
Gregor Stiglic, Juan J. Rodriguez, and Peter Kokol

Pseudoknot Identification through Learning TAG_{RNA} 132
Sahar Al Seesi, Sanguthevar Rajasekaran, and Reda Ammar

Support Vector Based T-Score for Gene Ranking 144
Piyushkumar A. Mundra and Jagath C. Rajapakse

Prediction of Transcription Factor Families Using DNA Sequence Features 154
Ashish Anand, Gary B. Fogel, Ganesan Pugalenthi, and P.N. Suganthan

g-MARS: Protein Classification Using Gapped Markov Chains and Support Vector Machines 165
Xiaonan Ji, James Bailey, and Kotagiri Ramamohanarao

Part III: Bio-Molecular Networks and Pathways Analysis

Domain-Domain Interaction Identification with a Feature Selection Approach 178
Xing-Ming Zhao and Luonan Chen

Dividing Protein Interaction Networks by Growing Orthologous Articulations 187
Pavol Jancura, Jaap Heringa, and Elena Marchiori

Constraint Minimization for Efficient Modeling of Gene Regulatory Network 201
Ramesh Ram, Madhu Chetty, and Dieter Bulach

Fusion of Gene Regulatory and Protein Interaction Networks Using Skip-Chain Models 214
Iti Chaturvedi and Jagath C. Rajapakse

TopEVM: Using Co-occurrence and Topology Patterns of Enzymes in Metabolic Networks to Construct Phylogenetic Trees 225
Tingting Zhou, Keith C.C. Chan, and Zhenghua Wang

Generating Synthetic Gene Regulatory Networks 237
Ramesh Ram and Madhu Chetty

Part IV: Microarray and Gene Expression Analysis

Gene Selection for Microarray Data by a LDA-Based Genetic Algorithm 250
Edmundo Bonilla Huerta, Béatrice Duval, and Jin-Kao Hao

Sequential Forward Selection Approach to the Non-unique Oligonucleotide Probe Selection Problem 262
Lili Wang, Alioune Ngom, and Luis Rueda

On Finding and Interpreting Patterns in Gene Expression Data from Time Course Experiments 276
Yvonne E. Pittelkow and Susan R. Wilson

Microarray Design Using the Hilbert–Schmidt Independence Criterion 288
Justin Bedo

Identifying Non-random Patterns from Gene Expression Profiles 299
Radhakrishnan Nagarajan, Meenakshi Upreti, and Mariofanna Milanova

A Study on the Importance of Differential Prioritization in Feature Selection Using Toy Datasets 311
Chia Huey Ooi, Shyh Wei Teng, and Madhu Chetty

Weighted Top Score Pair Method for Gene Selection and Classification 323
Huaien Luo, Yuliansa Sudibyo, Lance D. Miller, and R. Krishna Murthy Karuturi

Part V: Data Mining and Knowledge Discovery

Identifying Conserved Discriminative Motifs 334
Jyotsna Kasturi, Raj Acharya, and Ross Hardison

Exploratory Data Analysis for Investigating GC-MS Biomarkers 349
Ken McGarry, Kim Bartlett, and Morteza Pourfarzam

Multi-relational Data Mining for Tetratricopeptide Repeats (TPR)-Like Superfamily Members in *Leishmania spp.*: Acting-by-Connecting Proteins 359
Karen T. Girão, Fátima C.E. Oliveira, Kaio M. Farias, Italo M.C. Maia, Samara C. Silva, Carla R.F. Gadelha, Laura D.G. Carneiro, Ana C.L. Pacheco, Michel T. Kamimura, Michely C. Diniz, Maria C. Silva, and Diana M. Oliveira

Heuristic Non Parametric Collateral Missing Value Imputation: A Step Towards Robust Post-Genomic Knowledge Discovery 373
Muhammad Shoaib B. Sehgal, Iqbal Gondal, Laurence S. Dooley, and Ross Coppel

Protein Expression Molecular Pattern Discovery by Nonnegative Principal Component Analysis 388
Xiaoxu Han and Joseph Scazzero

Gene Ontology Assisted Exploratory Microarray Clustering and Its Application to Cancer 400
Geoff Macintyre, James Bailey, Daniel Gustafsson, Alex Boussioutas, Izhak Haviv, and Adam Kowalczyk

Discovery of Biomarkers for Hexachlorobenzene Toxicity Using Population Based Methods on Gene Expression Data 412
Cem Meydan, Alper Küçükural, Deniz Yörükoğlu, and O. Uğur Sezerman

Part VI: Applications of High Performance Computing

Exploiting Fine-Grained Parallelism in the Phylogenetic Likelihood Function with MPI, Pthreads, and OpenMP: A Performance Study 424
Alexandros Stamatakis and Michael Ott

Massively Parallelized DNA Motif Search on the Reconfigurable Hardware Platform COPACOBANA 436
Jan Schröder, Lars Wienbrandt, Gerd Pfeiffer, and Manfred Schimmler

GPU-MEME: Using Graphics Hardware to Accelerate Motif Finding in DNA Sequences 448
Chen Chen, Bertil Schmidt, Liu Weiguo, and Wolfgang Müller-Wittig

Accelerating BLASTP on the Cell Broadband Engine 460
Huiliang Zhang, Bertil Schmidt, and Wolfgang Müller-Wittig

Author Index 471

Sequence Based Prediction of Protein Mutant Stability and Discrimination of Thermophilic Proteins

M. Michael Gromiha[1], Liang-Tsung Huang[2], and Lien-Fu Lai[3]

[1] Computational Biology Research Center (CBRC), National Institute of Advanced Industrial Science and Technology (AIST), AIST Tokyo Waterfront Bio-IT Research Building, 2-42 Aomi, Koto-ku, Tokyo 135-0064, Japan
michael-gromiha@aist.go.jp
[2] Department of Computer Science and Information Engineering, MingDao University, Changhua 523, Taiwan
larry@mdu.edu.tw
[3] Department of Computer Science and Information Engineering, National Changhua University of Education, Changhua 500, Taiwan

Abstract. Prediction of protein stability upon amino acid substitution and discrimination of thermophilic proteins from mesophilic ones are important problems in designing stable proteins. We have developed a classification rule generator using the information about wild-type, mutant, three neighboring residues and experimentally observed stability data. Utilizing the rules, we have developed a method based on decision tree for discriminating the stabilizing and destabilizing mutants and predicting protein stability changes upon single point mutations, which showed an accuracy of 82% and a correlation of 0.70, respectively. In addition, we have systematically analyzed the characteristic features of amino acid residues in 3075 mesophilic and 1609 thermophilic proteins belonging to 9 and 15 families, respectively, and developed methods for discriminating them. The method based on neural network could discriminate them at the 5-fold cross-validation accuracy of 89% in a dataset of 4684 proteins and 91% in a test set of 707 proteins.

Keywords: Protein stability, rule generator, discrimination, prediction, thermophilic proteins, neural network, machine learning techniques.

1 Introduction

One of the most important tasks in protein engineering is to understand the mechanisms responsible for protein stability changes affected by single point mutations, which can be employed for constructing temperature sensitive mutants and used to identify a wide spectrum of drug resistance conferring mutations. Another related task is to understand the important factors for the extreme stability of thermophilic proteins and discriminating them from mesophilic ones.

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 1–12, 2008.

Several methods have been proposed for predicting the stability of proteins upon amino acid substitutions. These methods are mainly based on distance and torsion potentials [1,2], multiple regression techniques [3], energy functions [4], contact potentials [5], neural networks [6], support vector machines, SVMs [7,8], average assignment [9], classification and regression tool [10], backbone flexibility [11] etc. Further, it has been reported that the discrimination of stabilizing and destabilizing mutants is more important than its magnitude in many cases [6]. Most of these methods used the information from the three-dimensional structures of proteins for discrimination/prediction. On the other hand, prediction accuracy using amino acid sequence is significantly lower than that with structural data [12].

Several attempts have been made to understand the factors influencing the stability of thermophilic proteins using three-dimensional structural information as well as from amino acid sequence. It has been reported that increase in number of salt bridges and side chain-side chain interactions [13], counterbalance between packing and solubility [14], aromatic clusters [15], contacts between the residues of hydrogen bond forming capability [16,17], ion pairs [18], cation-π interactions [19,20], non-canonical interactions [21], electrostatic interactions of charged residues and the dielectric response [22,23], amino acid coupling patterns [24], main-chain hydrophobic free energy [25] and hydrophobic residues [26] in thermophilic proteins enhanced the stability. In addition, the amino acid sequences of genomes have been used for understanding the stability of thermophilic proteins. Das and Gerstein [27] reported that intra-helical salt bridges are prevalent in thermophiles. Fukuchi and Nishikawa [28] showed that the amino acid composition on protein surface may be an important factor for understanding the stability. Ding et al. [29] revealed the preferences of dipeptides in thermophilic proteins for extreme stability. Berezovsky et al. [30] found that the proteomes of thermophilic proteins are enriched in hydrophobic and charged amino acids at the expense of polar ones.

In spite of these studies, it is necessary to build a system, which derives stability rules for any input data and convert them into prediction. In this work, we have developed a classification rule generator to provide an online service for relating protein stability changes from the information about the mutated residue, three neighboring residues and the mutant residue. The rules can be interpreted to understand and predict protein stability changes upon point mutations. We have developed a method based on decision tree for discriminating /predicting protein mutant stability just from amino acid sequence. Using the information of a short window of seven residues (three residues on both directions of the mutant site) our method discriminated the stabilizing and destabilizing mutants with an accuracy of 82% and predicted the stability changes with a correlation of 0.70. Further, we have analyzed the performance of different algorithms, such as Bayes rules, neural network, SVM, decision trees etc for discriminating mesophilic and thermophilic proteins. We found that the 5-fold cross-validation accuracy is almost similar in most of the machine learning algorithms and the accuracy of discriminating mesophilic and thermophilic proteins using neural networks is marginally better than other methods. It could discriminate them at an accuracy of 93% and 89%, respectively, for self-consistency and 5-fold cross-validation tests in a dataset of 4684 proteins.

2 Materials and Methods

We have used different sets of data for predicting protein stability upon point mutations, and discriminating mesophilic and thermophilic proteins. Likewise, different methods have been used for these two studies.

2.1 Datasets

For the study on protein mutant stability, we have constructed a dataset of 1859 non-redundant single mutants from 64 proteins using ProTherm, the thermodynamic database for proteins and mutants available on the web [31,32]. We have removed the duplicate mutants that have same mutated and mutant residues, residue number, experimental conditions (pH and temperature, T) and $\Delta\Delta G$ values. Further, we retained only one data (the average value) for the mutants in which $\Delta\Delta G$ are reported with same T and pH, and different conditions (buffers/ions). We have used five variables for implementing the discrimination/prediction algorithm: (i) Md, mutated (deleted) residue, (ii) Mi, mutant (introduced) residue, (iii) pH, (iv) T (°C) at which the stability of the mutated protein was measured explicitly and (v) three neighboring residues of the central residue. These attributes have been selected with the balance between experimental conditions and sequence information.

Zhang and Fang [33] used 4895 mesophilic and 3522 thermophilic proteins for discriminating them using dipeptide composition. The proteins in each set contain many redundant sequences and we removed the redundancy using CD-HIT algorithm, [34] as implemented by Holm and Sander [35]. The final dataset contains 3075 mesophilic proteins and 1609 thermophilic proteins. Further, we have used a test set of 325 mesophilic and 382 thermophilic proteins belonging to Xylella fastidosa and Aquifex aeolicus families, respectively. These datasets have the proteins with less than 40% sequence identity.

2.2 Computation of Amino Acid Composition

The amino acid composition for each protein has been computed using the number of amino acids of each type and the total number of residues:

$$\text{Comp}(i) = \Sigma\, n_i/N, \tag{1}$$

where i stands for the 20 amino acid residues; n_i is the number of residues of each type and N is the total number of residues. The summation is through all the residues in the particular protein.

2.3 Methods for Discrimination and Prediction

We have used decision tree [36] along with adaptive boosting algorithm [37] for discriminating the stability of protein mutants, and classification and regression tree (CART) [38] for predicting the stability changes of proteins upon mutations. The decision tree algorithms can efficiently construct interpretable prediction models by measuring input variables directly from training data, which is suitable for large datasets and unknown data distribution. The decision tree has been selected with two

steps: in the first step, a recursive split procedure builds a tree, named maximum tree, which closely describes the training dataset and in the second step, the maximum tree is cut off for finding optimal sub tree. The adaptive boosting algorithm generates a set of classifiers from the data, each optimized to classify the correct ones that were misclassified in previous pass. Considering the exploitation of sets of hypotheses with independent errors it can improve the classification accuracy and reduce the variance as well as the bias.

We have analyzed several machine learning techniques implemented in WEKA program [39] for discriminating mesophilic and thermophilic proteins. This program includes several methods based on Bayes functions, neural networks, logistic functions, support vector machines, regression analysis, nearest neighbor methods, meta learning, decision trees and rules. The details of these methods have been explained in our earlier article [40]. We have analyzed different classifiers and datasets to discriminate mesophilic and thermophilic proteins.

2.4 Assessment of Predictive Ability

We have used different measures to assess the accuracy of discriminating mesophilic and thermophilic proteins, and stabilizing and destabilizing mutants. The term, sensitivity shows the correct prediction of thermophiles (stabilizing mutants), specificity about the mesophilies (destabilizing mutants) and accuracy indicates the overall assessment. The agreement between experimental and predicted stability changes has been assessed with correlation coefficient. These terms are defined as follows:

$$\text{Sensitivity} = TP/(TP+FN) \tag{2}$$

$$\text{Specificity} = TN/(TN+FP) \tag{3}$$

$$\text{Accuracy} = (TP+TN)/(TP+TN+FP+FN) \tag{4}$$

$$r = [N \Sigma XY - (\Sigma X\ \Sigma Y)]/\{[N \Sigma X^2 - (\Sigma X)^2]\ [N \Sigma Y^2 - (\Sigma Y)^2]\}^{1/2} \tag{5}$$

where, TP, FP, TN and FN refer to the number of true positives, false positives, true negatives, and false negatives respectively; r is the correlation coefficient, N, X, and Y are the number of data, experimental and predicted stability, respectively.

We have performed n-fold cross-validation test for assessing the validity of the present work. In this method, the data set is divided into n groups, n-1 of them are used for training and the rest is used for testing the method. The same procedure is repeated for n times and the average is computed for obtaining the accuracy of the method. We have carried out 2-fold, 3-fold, 4-fold, 5-fold and 10-fold cross validation tests.

3 Results and Discussion

3.1 Development of Classification Rules

We have developed a system composed of three components, which can sequentially develop protein sequence information to classification rules along with related analysis (Figure 1).

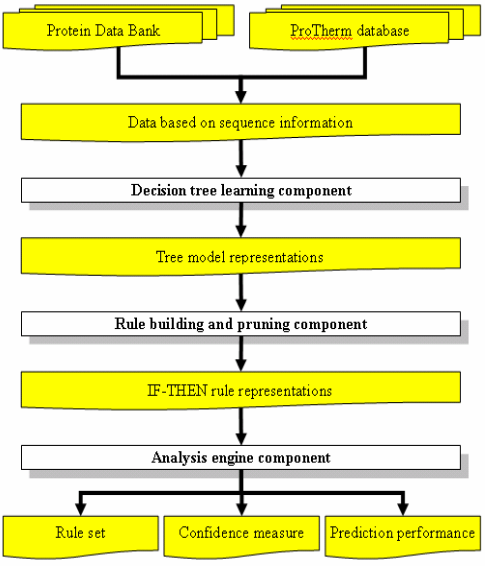


Fig. 1. Flowsheet of the learning process for depicting the relationship between components and data

The first component constructs a decision tree from the information about the mutated residue with three neighboring residues and the mutant residue. The mutation and neighboring residues information have been obtained from ProTherm database [31,32] and Protein Data Bank [41], respectively. Then the second one converts the learned tree into an equivalent set of rules, which may discriminate the stabilizing and destabilizing mutants as well as to explore the underlying concept of experimental data. The third provides further analyses from different viewpoints to clarify the characteristics of generated rules.

From the dataset of 1859 mutants, a total of 104 rules were generated. The rule size of the rule set being about 2 indicates the antecedent of these rules consist of about two statements on average. Generally, a shorter rule may make the rule easier to understand and to be examined. We further observed that 1535 samples of the dataset can match the antecedent of these rules with 175 errors, which showed the accuracy of 88.6%. It reveals that most samples in the dataset can be correctly inferred by using the rule set. In Table 1, we have given few examples of rules and their details: (i) if the mutated residue is Asp, its third neighbor at N-terminal is Glu, and its second neighbor at C-terminal is Leu, then the predicted stability change will be positive (stabilizing); we obtained an accuracy 96% in a set of 25 data; (ii) if the deleted residue is Ser and its first neighbor at N-terminal is Pro, then the predicted stability change will be negative (destabilizing), which correctly predicted all the 29 data with an accuracy of 100%; (iii) if the deleted residue is Leu, then the protein will be destabilizing; this rule is applied to 122 mutants and 115 are predicted correctly (accuracy 94%).

Table 1. Confidence measure for 5 rules with high accuracy and sufficient number of data from a dataset of 1859 non-redundant single mutants

Rule	Rule size	Number of data	Percentage of data (%)	Correctly predicted	Accuracy (%)	Predicted class
Mutated residue=D, N3=E, C2=L	3	25	1.34	24	96	Stabilizing
Mutated residue=T, C1=V	2	29	1.56	24	83	Stabilizing
Mutated residue=S, N1=P	2	29	1.56	29	100	Destabilizing
Mutated residue=L	1	122	6.56	115	94	Destabilizing
Mutant residue=G	1	66	3.55	62	94	Destabilizing

We have developed a web interface for generating rules for any set of stability data using wild type, mutant and three neighboring residue information. We have also provided the related dataset for different tests along with the generated rules on the web server.

3.2 Prediction of Protein Stability

We have utilized the rules for discriminating the stabilizing and destabilizing mutants and predicting the stability change upon mutation along with the information about pH and T. The validity of our approach has been assessed with 4-fold, 10-fold and 20-fold cross-validation procedures. The 4-fold and 20-fold cross-validation tests yielded the accuracy of 81.4% and 82.1% for discriminating the stability of protein mutants. The sensitivity and specificity are 75.3% and 84.5%, respectively [42]. Further, our method could predict the stability of protein mutants with the correlation coefficient of 0.70.

The main features of the present method are: (i) it is based on the neighboring residues of short window length, (ii) it can predict the stability from amino acid sequence alone, (iii) developed different servers for discrimination and prediction, and integrated them together, (iv) utilized the information about experimental conditions, pH and T, and (v) implemented several rules for discrimination and prediction from the knowledge of experimental stability and input conditions: (i) if the deleted residue is Ala and the neighboring residues contain Gln, then the predicted stability change will be negative (accuracy = 97.1%), (ii) if the deleted residue is Glu and its second neighbor at N-terminal is Met, the mutation stabilizes the protein (accuracy = 100%) and (iii) if the deleted-residue belongs to Y, W, V, R, P, M, L, I, G, F or C, and the introduced-residue belongs to T, S, P, K, H, G or A, then the predicted stability change will be -2.05 kcal/mol (mean absolute error = 1.57 kcal/mol).

We have developed a web server for discriminating the stabilizing and destabilizing mutants and predicting the stability of proteins upon mutations. The program takes the information about the mutant and mutated residues, three neighboring residues on both sides of the mutant residue along with pH and T. In the output, we display the predicted protein stability change upon mutation along with input conditions (Figure 2). In the case

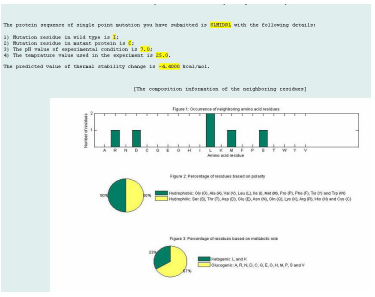


Fig. 2. The results obtained for predicting the stability change along with the related information of neighboring residues

of discrimination, we show the effect of the mutation to protein stability, whether stabilizing or destabilizing. Both discrimination and prediction services offer an option for additional sequence composition information of neighboring residues (Figure 2). The bar chart shows the number of amino acids of each type. The two pie charts below represent the percentage of residues according to polarity and the metabolic role of amino acids. The prediction/discrimination results are available at http://bioinformatics.myweb.hinet.net/iptree.htm.

In our method, we have used the balance between experimental conditions and sequence information as features for prediction. These features are different from other methods, which mainly used contact potentials, 40 different combinations of mutations, solvent accessibility, secondary structure, average stability value for each mutation, experimental conditions etc. for predicting the stability. In addition, we have used different features including the variation of window length along the sequence and we observed the best performance with the information about mutant and mutated residues as well as three neighboring residues along the sequence.

We have compared the performance of CART with neural networks (NN) and support vector machines (SVM) using same features. The ROC curve obtained for the three methods with 20-fold cross-validation test is shown in Figure 3. We observed

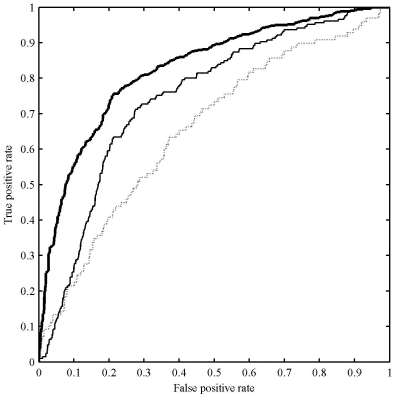


Fig. 3. ROC curves for CART (thick line), SVM (thin line) and NN (broken line)

that the performance of CART is the best among all the three methods. The areas under the curve (AUC) for CART, SVM and NN are 0.83, 0.75 and 0.66, respectively.

3.3 Discrimination of Mesophilic and Thermophilic Proteins

We have computed the amino acid composition of mesophilic and thermophilic proteins and the results are shown in Figure 4. From this figure, we observed that the composition of Ala, Leu, Gln and Thr are higher in mesophiles than thermophiles an opposite trend is observed for Glu, Lys, Arg and Val [43]. These preferences and the higher occurrence of other amino acids in thermophilic proteins reveal the implications for protein stability.

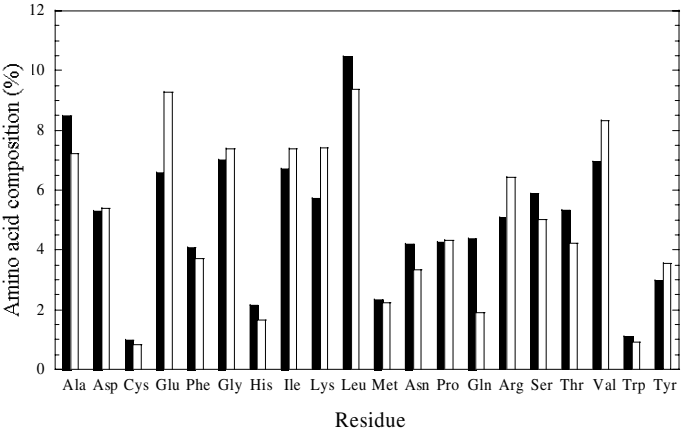


Fig. 4. Amino acid composition in mesophilic (■) and thermophilic (□) proteins

The comparative analysis on the occurrence of Cys, Ile and Val in the structural homologues of 23 mesophilic and thermophilic proteins [25] showed that the occurrence of Cys is less in thermophiles than mesophiles. On the other hand, the occurrence of Val/Ile is higher in thermophiles than mesophiles. In addition, it has been reported that Cys can be replaced by Val/Ile to enhance the stability [14]. Interestingly, these trends were reflected in the analysis of amino acid composition. Further, the charged residues, Lys, Arg and Glu have significantly higher occurrence in thermophilic proteins than mesophilic ones and the composition of Asp showed a moderate difference (Figure 4). We have analyzed the composition of charged residues in the structural homologues of thermophilic and mesophilic proteins and observed that the thermophiles have more number of charged residues than mesophiles. This result supports our observation obtained with amino acid sequence analysis.

We have analyzed the performance of different machine learning techniques for discriminating mesophilic and thermophilic proteins. In this discrimination, we have used the amino acid composition as the main attributes. We observed that most of the machine learning methods discriminated the mesophilic and thermophilic proteins with the accuracy in the range of 84-89% in a set of 4684 proteins. This analysis showed that there is no significant difference in performance between different

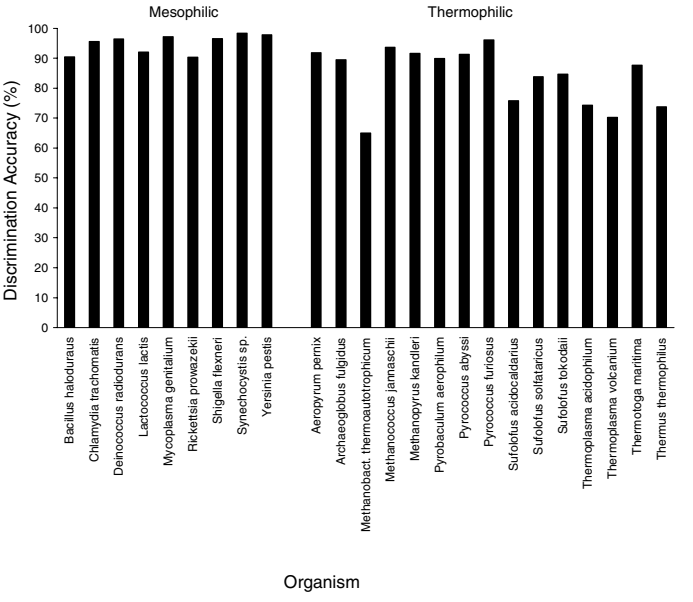


Fig. 5. Discrimination accuracy in different mesophilic and thermophilic organisms

machine learning methods. Interestingly, the methods neural networks, support vector machines and logistic functions discriminated mesophilic and thermophilic proteins at similar accuracy of 89%. The accuracy of identifying thermophilic proteins is 87% where as that of excluding mesophilic proteins is 96%. The overall accuracy is 89.4% for distinguishing mesophilic and thermophilic proteins.

The accuracy of discriminating mesophilic and thermophilic proteins in different families has been analyzed and the results are depicted in Figure 5.

We observed that the proteins in most of the mesophilic families are discriminated with the accuracy of more than 90%. On the other hand, the accuracy of discriminating thermophilic proteins showed a vide variation of 65 to 96%. Further analysis on this family of proteins revealed that the number of proteins in this family is significantly less (20 proteins) and most of the proteins are showing high sequence identity with mesophilic proteins. In addition, we have analyzed the discrimination accuracy of thermophilic (moderate) and hyper (extreme) thermophilic proteins from mesophilic proteins. Interestingly, we observed that hyper-thermophilic proteins are discriminated with higher accuracy than moderate thermophilic proteins. The accuracies of discriminating hyper-thermophilic and thermophilic proteins from mesophilic ones are, 90% and 73%, respectively.

We have assessed the reliability of the present method by discriminating mesophilic and thermophilic proteins from different families that are not considered in the work for training/ testing. We have collected the data of 325 mesophilic and 382 thermophilic proteins from Xylella fastidosa and Aquifex aeolicus families, respectively. We observed that the present method based on neural networks correctly identified the thermophilic proteins with the sensitivity of 87.6%. Further, the

mesophilic proteins are excluded with the specificity of 95.7% and the overall accuracy is 91.3%. These results demonstrated that our method is performing extremely well in distinguishing mesophilic and thermophilic proteins.

4 Conclusions

We have developed a rule generator for classifying the stabilizing and destabilizing protein mutants based on wild type, mutant and three neighboring residue information. These rules have been effectively used to discriminate the stabilizing and destabilizing mutants, and predicting the stability of a protein upon point mutation. Our method could achieve the accuracy of 82% and a correlation of 0.70 for discrimination and prediction, respectively, just from amino acid sequence. Further, different machine learning techniques have been analyzed for discriminating the mesophilic and thermophilic proteins and showed that these proteins are discriminated with the accuracy of 89%. Our method used simple features and achieved high accuracy and hence it is suitable for prediction. We suggest that our method could be effectively used in protein design.

References

1. Gilis, D., Rooman, M.: Stability changes upon mutation of solvent-accessible residues in proteins evaluated by database-derived potentials. J. Mol. Biol. 257, 1112–1126 (1996)
2. Parthiban, V., Gromiha, M.M., Hoppe, C., Schomburg, D.: Structural analysis and prediction of protein mutant stability using distance and torsion potentials: role of secondary structure and solvent accessibility. Proteins 66, 41–52 (2007)
3. Gromiha, M.M., Oobatake, M., Kono, H., Uedaira, H., Sarai, A.: Role of structural and sequence information in the prediction of protein stability changes: comparison between buried and partially buried mutations. Protein Eng. 12, 549–555 (1999)
4. Guerois, R., Nielsen, J.E., Serrano, L.: Predicting changes in the stability of proteins and protein complexes: a study of more than 1000 mutations. J. Mol. Biol. 320, 369–387 (2002)
5. Khatun, J., Khare, S.D., Dokholyan, N.V.: Can contact potentials reliably predict stability of proteins? J. Mol. Biol. 336, 1223–1238 (2004)
6. Capriotti, E., Fariselli, P., Casadio, R.: A neural-network-based method for predicting protein stability changes upon single point mutations. Bioinformatics 20(1), i63–68 (2004)
7. Capriotti, E., Fariselli, P., Casadio, R.: I-Mutant2.0: predicting stability changes upon mutation from the protein sequence or structure. Nucleic Acids Res. 33, w306–310 (2005)
8. Cheng, J., Randall, A., Baldi, P.: Prediction of protein stability changes for single-site mutations using support vector machines. Proteins 62, 1125–1132 (2006)
9. Saraboji, K., Gromiha, M.M., Ponnuswamy, M.N.: Average assignment method for predicting the stability of protein mutants. Biopolymers 82, 80–92 (2006)
10. Huang, L.T., Saraboji, K., Ho, S.Y., Hwang, S.F., Ponnuswamy, M.N., Gromiha, M.M.: Prediction of protein mutant stability using classification and regression tool. Biophys. Chem. 125, 462–470 (2007)
11. Yin, S., Ding, F., Dokholyan, N.V.: Modeling backbone flexibility improves protein stability estimation. Structure 15, 1567–1576 (2007)

12. Gromiha, M.M.: Prediction of protein stability upon point mutations. Biochem. Soc. Trans. 35, 1569–1573 (2007)
13. Kumar, S., Tsai, C.J., Nussinov, R.: Factors enhancing protein thermostability. Protein Eng. 13, 179–191 (2000)
14. Gromiha, M.M., Oobatake, M., Sarai, A.: Important amino acid properties for enhanced thermostability from mesophilic to thermophilic proteins. Biophys. Chem. 82, 51–67 (1999)
15. Kannan, N., Vishveshwara, S.: Aromatic clusters: a determinant of thermal stability of thermophilic proteins. Protein Eng. 13, 753–761 (2000)
16. Gromiha, M.M.: Important inter-residue contacts for enhancing the thermal stability of thermophilic proteins. Biophys. Chem. 91, 71–77 (2001)
17. Gromiha, M.M., Selvaraj, S.: Inter-residue interactions in protein folding and stability. Prog. Biophys. Mol. Biol. 86, 235–277 (2004)
18. Kumar, S., Tsai, C.J., Nussinov, R.: Thermodynamic differences among homologous thermophilic and mesophilic proteins. Biochemistry 40, 14152–14165 (2001)
19. Gromiha, M.M., Thomas, S., Santhosh, C.: Role of cation-pi interactions to the stability of thermophilic proteins. Prep. Biochem. Biotechnol. 32, 355–362 (2002)
20. Chakravarty, S., Varadarajan, R.: Elucidation of factors responsible for enhanced thermal stability of proteins: a structural genomics based study. Biochemistry 41, 8152–8161 (2002)
21. Ibrahim, B.S., Pattabhi, V.: Role of weak interactions in thermal stability of proteins. Biochem. Biophys. Res. Commun. 325, 1082–1089 (2004)
22. Xiao, L., Honig, B.: Electrostatic contributions to the stability of hyperthermophilic proteins. J. Mol. Biol. 289, 1435–1444 (1999)
23. Dominy, B.N., Minoux, H., Brooks, C.L.: 3rd: An electrostatic basis for the stability of thermophilic proteins. Proteins 57, 128–141 (2004)
24. Liang, H.K., Huang, C.M., Ko, M.T., Hwang, J.K.: Amino acid coupling patterns in thermophilic proteins. Proteins 59, 58–63 (2005)
25. Saraboji, K., Gromiha, M.M., Ponnuswamy, M.N.: Importance of main-chain hydrophobic free energy to the stability of thermophilic proteins. Int. J. Biol. Macromol. 35, 211–220 (2005)
26. Sadeghi, M., Naderi-Manesh, H., Zarrabi, M., Ranjbar, B.: Effective factors in thermostability of thermophilic proteins. Biophys. Chem. 119, 256–270 (2006)
27. Das, R., Gerstein, M.: The stability of thermophilic proteins: a study based on comprehensive genome comparison. Funct. Integr. Genomics 1, 76–88 (2000)
28. Fukuchi, S., Nishikawa, K.: Protein surface amino acid compositions distinctively differ between thermophilic and mesophilic bacteria. J. Mol. Biol. 309, 835–843 (2001)
29. Ding, Y., Cai, Y., Zhang, G., Xu, W.: The influence of dipeptide composition on protein thermostability. FEBS Lett 569, 284–288 (2004)
30. Berezovsky, I.N., Zeldovich, K.B., Shakhnovich, E.I.: Positive and negative design in stability and thermal adaptation of natural proteins. PLoS Comput. Biol. 3, 52 (2007)
31. Gromiha, M.M., An, J., Kono, H., Oobatake, M., Uedaira, H., Sarai, A.: ProTherm: Thermodynamic Database for Proteins and Mutants. Nucleic Acids Res. 27, 286–288 (1999)
32. Bava, K.A., Gromiha, M.M., Uedaira, H., Kitajima, K., Sarai, A.: ProTherm, version 4.0: thermodynamic database for proteins and mutants. Nucleic Acids Res. 32, D120–D121 (2004)

33. Zhang, G., Fang, B.: Application of amino acid distribution along the sequence for discriminating mesophilic and thermophilic proteins. Process biochemistry 41, 1792–1798 (2006)
34. Li, W., Jaroszewski, L., Godzik, A.: Clustering of highly homologous sequences to reduce the size of large protein databases. Bioinformatics 17, 282–283 (2001)
35. Holm, L., Sander, C.: Removing near-neighbour redundancy from large protein sequence collections. Bioinformatics 14, 423–429 (1998)
36. Quinlan, J.R.: C4.5: programs for machine learning. Morgan Kaufmann Publishers, San Mateo (1993)
37. Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and System Sciences 55, 119–139 (1997)
38. Breiman, L.: Classification and regression trees. Wadsworth International Group, Belmont (1984)
39. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann, San Francisco (2005)
40. Gromiha, M.M., Suwa, M.: Discrimination of outer membrane proteins using machine learning algorithms. Proteins 63, 1031–1037 (2006)
41. Berman, H.M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T.N., Weissig, H., Shindyalov, I.N., Bourne, P.E.: The Protein Data Bank. Nucleic Acids Res. 28, 235–242 (2000)
42. Huang, L.T., Gromiha, M.M., Ho, S.Y.: iPTREE-STAB: interpretable decision tree based method for predicting protein stability changes upon mutations. Bioinformatics 23, 1292–1293 (2007)
43. Gromiha, M.M., Suresh, M.X.: Discrimination of mesophilic and thermophilic proteins using machine learning algorithms. Proteins 70, 1274–1279 (2008)

A Method to Find Sequentially Separated Motifs in Biological Sequences (SSMBS)

Chetan Kumar[1], Nishith Kumar[1], Sarani Rangarajan[1], Narayanaswamy Balakrishnan[2], and Kanagaraj Sekar[1,2,⋆]

[1] Bioinformatics Centre (Centre of excellence in Structural Biology and Bio-computing)
Tel.: +91-80-22933059/22932469/23601409; Fax: +91-80-23600683/23600551
sekar@physics.iisc.ernet.in, chetan-k@northwestern.edu
nishith_iitd@yahoo.co.in, sarani.rangarajan@gmail.com
[2] Supercomputer Education and Research Centre, Indian Institute of Science, Bangalore 560 012, India
{balki,sekar}@serc.iisc.ernet.in
http://www.physics.iisc.ernet.in/~dichome/sekhome/index.html

Abstract. Sequence motifs occurring in a particular order in proteins or DNA have been proved to be of biological interest. In this paper, a new method to locate the occurrences of up to five user-defined motifs in a specified order in large proteins and in nucleotide sequence databases is proposed. It has been designed using the concept of quantifiers in regular expressions and linked lists for data storage. The application of this method includes the extraction of relevant consensus regions from biological sequences. This might be useful in clustering of protein families as well as to study the correlation between positions of motifs and their functional sites in DNA sequences.

Keywords: Regular expressions, protein and nucleotide sequences, sequence motifs.

1 Introduction

Research on proteins and DNA has revealed that specific motifs in biological sequences exhibit important characteristics [1]. This has spurred the development of computational methods to search for sequence motifs of biological significance. Further, the exponential rise in the volume of protein and nucleotide sequences has necessitated the development of algorithms that are both time and space efficient to make optimum use of available computational resources. Here, an efficient method is proposed that locates all occurrences of motifs of biological interest in a specific order using the concept of quantifiers in regular expressions.

⋆ Corresponding author.

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 13–27, 2008.

We refer to motifs that occur in a particular order as "sequentially separated motifs", since they could be separated by intermediate amino acid residues or nucleotides.

Recent studies have considered sequentially separated motifs as a method for classifying DNA sequences based on the presence and relative positions of a few transcription factor (TF) binding sites. These binding sites are of such importance that several algorithms and online tools are available for their detection[2],[3],[4],[5] . Binding sites are relatively short stretches of DNA, normally 5 to 35 nucleotides long and occur as consensus regions or well conserved regions called motifs. It has been established in literature that binding sites are often found in a well-ordered and regularly spaced manner [6],[7],[8] . In prokaryotic organisms, the binding sites are located predominantly in the region that extends about 300 to 600 nucleotides upstream of the transcription start site, in the promoter regions. However, in eukaryotic organisms, the binding sites, called cis-regulatory modules (CRMs), usually occur in a fixed arrangement and are distributed over very large distances. A detailed explanation of eukaryotic promoters can be found in literature [9],[6]. A eukaryotic promoter is considered to comprise of three CRMs, each having one or more TF binding sites. Since each CRM has a different function, it will be helpful to have a method that can locate the distribution of the occurrences of the three CRMs in the order in which they exist in the sequence. Further, repeated occurrences of CRMs in the DNA sequence might lead to alternate modes of binding by the same protein, thereby regulating transcriptional activity. In addition, it may lead distinct proteins to recognize the identical CRMs occurring at different positions in the sequence. Also, if the signature motifs for trans-regulatory modules are known, they too can be detected to achieve a more complete understanding of the the structure of the gene and its regulation.

Furthermore, sequentially separated conserved motifs have been used to categorize new and unknown protein structures. For instance, the classification of T6PP as a member of the haloacid dehalogenase (HAD) superfamily is based on the presence of three highly conserved motifs that are found in all enzymes belonging to the HAD family. The three motifs are DXXX(V/T), followed by (S/T)GX, and finally $K(X)_{(16-30)}$(G/S)(D/S)XXX(D/N), where X denotes a wild card symbol that can be substituted by any of the 20 amino acids and G/S signifies the presence of G or S at the particular position in the motif [10],[11]. The HAD superfamily is further subdivided into three structural groups based on the length of the sequence between the motifs [12]. Thus, it can be concluded that in proteins, the intermediate sequences that separate the sequential motifs are also biologically significant. The concept of sequentially separated motifs finds an important application in remote homology detection of proteins. Homology is generally established by sequence similarity. In the past two decades, many methods for measuring sequence similarity have been developed. The two most popular methods are the Smith-Waterman algorithm [11] and its faster counterpart, BLAST[13]. Protein sequence motifs can offer an alternative way of detecting sequence similarity. By closely studying highly conserved sequence

motifs, important clues to a protein function might be revealed even if it is not globally similar to any known protein [14]. In addition, the sequentially separated motifs for most catalytic sites and binding sites are conserved over much wider taxonomic distances and evolutionary time than the protein sequences themselves [15]. Thus, it can be deduced that motifs that are found to occur in a particular order could represent functionally important regions such as catalytic sites, binding sites, protein-protein interaction sites and structural motifs.

In view of the biological relevance of sequentially separated motifs, a need is felt to develop a method that can detect the occurrences of the motifs in large sequence databases efficiently. The performance of such a method designed to solve this problem should be judged according to the following criteria:

1. Efficiency: To analyze large nucleotide and proteins sequences (e.g. the human chromosome 1 contains 240 million nucleotide bases and the proteome of *A. thaliana* more than 7,000 protein sequences), the space and time complexity of the method must scale linearly with the sequence length and the number of sequences. Further, the method should also minimize the number of iterations and comparisons required to report all occurrences of the motifs.
2. Flexibility: The motifs should be specified using regular expressions.
3. Accuracy: To identify all locations, including degenerate occurrences and overlapping occurrences.
4. User-Friendliness: It should be simple to use, platform independent and display results in an elegant and easily comprehensible manner.

1.1 Existing Algorithms

Two types of pattern matching algorithms are commonly used in biology:

scan_for_matches. [16] brought on a series of other software and algorithms, including PatScan [17] which searches a dataset for matches against a query pattern. PatSearch [18] has added features such as the assessment of the statistical significance of pattern hits using a Markov chain simulation. The results of these programs display the entire substring that contains the motif provided by the user but do not explicitly indicate the individual occurrences of the motifs. Due to this, the user needs to manually delineate the intermediate residues that separate the motifs.

grep-based programs. An example of which is eMOTIF-SCAN, a program which uses the agrep tool that supports matching and regular expression. However, it searches only against the eMOTIF database of protein sequence motifs [19].

The program, Scansite 2.0 [20] searches for up to two motifs and looks for the occurrences of these motifs in no particular order of arrangement. Motif Scan [21] searches for motifs against protein profile databases including Prosite [1] and Pfam [22], and, thus does not provide the users with the option to enter their own motifs. Though most of the above mentioned programs work efficiently with

protein sequences, they do not perform well with large sequences. In most cases, the programs do not execute to completion for very large nucleotide sequences (150 million bp).

Furthermore, the pattern search present in the PIR database is also extremely efficient for a single motif (or when a number of motifs can be combined to a single motif). However, when the specific number of residues between two or more motifs cannot be identified, two separate PIR pattern searches must be run and the results compared either manually or through a program written specifically to obtain the required sequences from the output of the two searches. This process becomes much more complicated when more than two motifs are being searched in order in a set of sequences. Finally, in SSMBS, the database of sequences to be searched for the motif can be specified or uploaded. On the other hand, in the PIR pattern search, only two options for databases exist if a search for a user-defined motif must be carried out: UniPotKB and UniRef100. Thus, when the user wishes to find a number of motifs in order (with an unknown of large number of residues separating the motifs) in a user specified database, SSMBS is the only available option.

2 Materials and Methods

2.1 Basic Definitions

If S = {A, C, G, T, U} is the alphabet defined for nucleotide sequences (U for RNA) and S = {A, R, N, D, C, E, Q, G, H, I, L, K, M, F, P, S, T, W, Y, V} is the alphabet defined for amino acid sequences, then let S, defined over S, represents the sequence in which the sequentially separated motifs are to be located. Further, let n = |S| i.e. length of S. Let m be the number of input sequences.

S[i] denotes the i^{th} character of S, for i ∈ [1,n]. For i ≤ j ≤n, S[i,j] denotes the substring of S starting with the i^{th} and ending with the j^{th} character of S. Thus, the length of S[i,j] is j - i + 1.

Let M = { $motif_i | 1 \leq i \leq 5$ } be the set of up to five motifs entered by the user and k = |M|, i.e. number of motifs, where M is defined over S.

L denotes a linked list whose elements comprise of many other linked lists, each called L'. L' contains the starting positions of the occurrences of the M_i such that L' = { $(s_i),(s_{i+1}),\ldots,(s_k)$ | s_i is starting position of M_i; $1 \leq s_i \leq n$; $1 \leq i \leq 5$ }.

2.2 Use of Quantifiers

Quantifiers, as the name implies, express quantity i.e. how much or how many. They are used in pattern matching since they allow us to control the amount of text in a sequence that is to be matched against a pattern. Quantifiers have already been implemented in several programming languages including JAVA and Perl and they are an integral part of regular expression matching. In biological

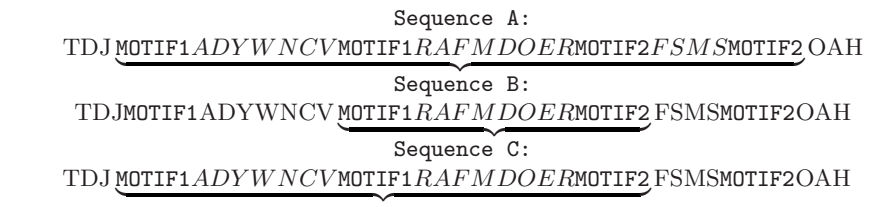


Fig. 1. Sequences A, B and C represent the three different query patterns [(.*), (.?) and (.*?) respectively] used to simultaneously locate the two motifs, motif1 (solid block) and motif2 (grey box) in that order

$$\underbrace{[MOTIF1](.*?)[MOTIF2]...[MOTIF_K]}_{EXPR}$$

Fig. 2. $EXPR$ represents the combined query pattern which is formed by appending the (.*?) quantifier between the k motifs entered by the user

sequences, they can be used to match complex motifs that are defined using regular expressions.

'*' is a greedy quantifier which tries to match as much text as possible in the query string. However, '?' is a reluctant quantifier which tries to match as less text as possible. In this method, 'minimal matching' is utilized: the two quantifiers, '*' and '?' are coupled in the order (.*?) and appended between the two motifs motif1, motif2 ∈ M , such that: [motif1](.*?)[motif2]. This enables the detection of both occurrences simultaneously (Figure 1*c*).

This concept can be further extended to simultaneously detect the first occurrences of any number of motifs. This can be achieved by appending '(.*?)' between the motifs to form an expression EXPR as shown in Figure 2.

EXPR suggests that the SSMBS (Sequentially Separated Motifs in Biological Sequences) method appends the (.*?) quantifier after every motif till the k^{th} motif. At the time of execution, the user is asked to specify whether the sequence file provided contains amino acids or nucleotides. The method exploits the technique explained above to search for motifs in a defined order in proteins sequences. However, in case of large nucleotide sequences (>100,000 bp), the method follows the divide and conquer approach, as outlined in the subsequent sections.

2.3 Amino Acid Sequences

Let us consider a case in which the user enters five sequentially separated motifs and a set of 10,000 amino acid sequences. As SSMBS reads each sequence, it first checks whether there exists, in that sequence, at least a single occurrence of the five motifs in the order specified by the user. If a match is found, then it attempts to find all occurrences of the motifs in that particular sequence. If there does not exist any match, it moves to the next sequence and performs the

same check. To find all occurrences of the five sequentially separated motifs in these sequences, SSMBS first simultaneously locates all occurrences of the last two motifs i.e. motif4 and motif5, followed by occurrences of motif3, motif2 and finally motif1. An explanation of this procedure for k motifs follows.

Locating all occurrences of k motifs. For k motifs entered by the user (k $\leq$ 5), the last two motifs are $\text{motif}_{(k-1)}$ and $\text{motif}_k \in$ M respectively. Let 'R' be the set of remaining motifs i.e. motif_1 to $\text{motif}_{k-2} \in$ M. The terms 'last two motifs' and 'R' hold significance as they divide the method into two fundamental parts: first, finding all occurrences of the last two motifs and second, finding all occurrences of the R motifs in desired order. To locate all occurrences of the last two motifs, the method appends the '(.*?)' quantifier between the motifs to locate their occurrences simultaneously in the order, (k-1)$^{\text{th}}$ motif followed by the k$^{\text{th}}$ motif. The matching performed by the method returns the starting index of $\text{motif}_{(k-1)}$ and the end index of $\text{motif}_{(k)}$. Further, a series of iterations are performed to extract all occurrences of the two motifs in the specified order. The procedure of the first step is illustrated in the form of a pseudo code as shown below:

```
 Pat = [motif_(k−1)](.*?)[motif_k], SEQ = S
do {
if(find(Pat)) // returns true if a match is found.
{ start-index-m_(k−1) = start(); // returns starting index of m_(k−1).
end-index-m_k = end(); // returns ending index of m_k.
start-index-m_k = start(matcher(motif_k,SEQ.substring(start(),end());
// searches for motif_k only at the end of the substring.
SEQ.substring(start(),end());
Linked list changes ();
SEQ = S.substring(start-index-m_k−1, start-index-m_k-1)
+ S.substring(start-index-m_k+1,n);
```

$$\underbrace{\underbrace{MOTIF_{K-1}NDRKEMOTIF_{K-1}LVAY}\underbrace{MOTIF_kAMTEMOTIF_kLGL}}_{SEQ}$$

```
}
else { SEQ = S.substring(start-index-m_k−1 + 1, n);
```

$$\underbrace{MOTIF_{k-1}NDRKEMOTIF_{k-1}LVAYMOTIF_KAMTELPMOTIF_KLGL}$$

```
}
} while(no more matches of Pat can be found)
}
```

The second step begins when no more occurrences of the last two motifs can be found. In this step, all occurrences of the k motifs are found in the following order: (k-p)$^{\text{th}}$ motif (where p = 2,3,. . . ,(k-2)) to 1$^{\text{st}}$ motif. Thus, while searching for the occurrences of the (k-p)th motif, the method has already obtained all the occurances of the (k-p+1)th to kth motifs. All occurrences of (k-p+1)$^{\text{th}}$ to k$^{\text{th}}$ motifs are stored as a linked list L' in the form (start positions of (k-p+1)$^{\text{th}}$, (k-p+2)$^{\text{th}}$,. . . . . , k$^{\text{th}}$ motifs). To update these ordered sets by appending the

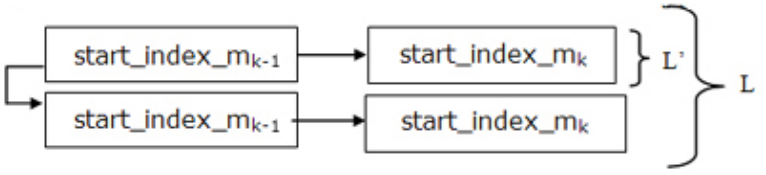


Fig. 3. Linked List appending and changes

start position of the (k-p)th motif at the beginning of L', it goes on comparing the end index of the position of the motif being dealt with, which is the (k-p)thmotif, with the first entry of every ordered set. The comparison is made at every iteration in which a new occurrence of motif$_{(k-p)}$ is detected. After successfully attaching the start index of motif$_{(k-p)}$ to L', the method appends L' to L.

For the subsequent iterations of this step, SSMBS retains only those elements or ordered sets to which the append was carried out successfully. Thus, at every iteration, the unwanted sets are eliminated, thereby shortening the size of the linked list L, to be searched in the iterations to follow (Figure 3).

2.4 Nucleotide Sequences

Unlike amino acids sequences, nucleotide sequences are very large often comprising of millions of bases. Their large size poses a major challenge in locating sequentially separated motifs because it is a memory exhaustive process. Accordingly, SSMBS adopts a divide and conquer strategy, breaking down the large sequence into small fragments comprising of 3,500 nucleotides. The value of 3,500 nucleotides per fragment is an optimal value that was heuristically determined after considering the time taken by the program implementing this method for varying sizes of the fragments. Let the fragments be denoted by F_s where s ranges from 1 to (n/3500 + 1). The method begins locating the occurrences of the sequentially separated motifs by traversing each fragment starting from F_1. The fragment in which the first occurrence of motif$_1$ is detected is marked F_m. Attempts to detect the occurrences of other motifs are carried out only in the fragments that follow F_m. In addition, the method also checks for any occurrences of the motifs that might overlap between regions common to two consecutive fragments, say F_a and F_{a+1} where a < (n/3500 + 1). It does so by searching for an occurrence of either of the k motifs in the string F_a + F_{a+1}('+' denotes concatenation) and confirming whether the starting position of the substring that matches any of the motifs is less than the length of F_a i.e. $|F_a|$ and the ending position is greater than $|F_a|$. Finally, the method collates all occurrences of the k motifs and displays the results by traversing the linked list L, which stores the individual occurrences of the k motifs as in the case of amino acids sequences explained earlier.

Locating overlapping occurrences of motifs. The proposed method locates all overlapping occurrences of a motif as well and thus misses no occurrence. For instance, the motif M_{exmpl} = ATA{3,5} can be found to occur six times in a sequence of the form ATAATAATAATAATA i.e three occurrences of ATAATAATA, two occurrences of ATAATAATAATA and single occurrence of ATAATAATAATAATA. To report such overlapping occurrences, the method initially attempts a greedy match to find an occurrence of the M_{exmpl} in the sequence. For every occurrence of M_{exmpl}, SSMBS then attempts a reluctant match to find occurrences of the motif that might exist within or that overlap with the matched string that was returned as a result of the greedy search. This is achieved by appending the reluctant quantifier '?' to M_{exmpl} to form the new expression M_{exmpl}' = ATA{3,5}? Now, SSMBS matches M_{exmpl}' against the substring that matches M_{exmpl}. Thus, the reluctant match returns (ATAATAATA: 1 to 9) as the first overlapping occurrence. Successive iterations of this step return all possible overlapping occurrences.

2.5 Time Complexity

The computational complexity of SSMBS method is explained based on the following points:

1. **Complexity with regard to number of proteins sequences:** The SSMBS method searches for occurrences of k motifs only in those sequences that have at least one occurrence of EXPR. As explained earlier, EXPR detects the ordered occurrence of k motifs in O(n) time, where n is the length of the sequence. If there are m sequences in all, then in O(mn) time, the method searches for all sequences that have at least one occurrence of EXPR. Hence, the method scales linearly with the number of input sequences. This is notable especially in the context of the exponential rise in the size of sequence databases.
2. **Complexity with regard to locating all occurrences in a given sequence:** The method is able to detect all ordered occurrences of k motifs in k-1 scans of the sequence, as compared to k scans in a brute force approach. Further, as the computation grows, it optimizes by reducing the length of the query sequence based on motif positions located in previous iterations. For instance, while searching for the $motif_R$, the algorithm searches only till the last occurrence of $motif_{R+1}$ in the sequence. Specifically, the performance of the method is bounded polynomially by $O(n^{k-1})$.
3. **Complexity specifically for nucleotide sequences:** By following the divide and conquer strategy in nucleotide sequences, the method successfully avoids the out of memory problem no matter how large the nucleotide sequence is. As in the case of proteins sequences, the complexity of the algorithm scales linearly with the size of nucleotide sequence. Thus, the algorithm can be applied to search for specific regions in entire genome of different organisms.

3 Biological Applications

3.1 Motifs Specified Using Regular Expressions

Motifs with biological importance often occur with some mutations or substituted residues in the sequence. Thus, regular expressions are used to specify such motifs in SSMBS. This process is quite similar to that found in the PIR pattern search. However, one main difference between the SSMBS algorithm and the PIR pattern search is that SSMBS can search for multiple motifs in a particular order, while PIR's pattern search is limited to those patterns in which the number of intervening residues between two motifs is at least approximately known. A few examples are:

1. String motifs: Motifs such as CXXCXXC will match any substring that has first, fourth and last characters as C. 'X' denotes a wild card residue that can match any amino acid or nucleotide.
2. Range motifs: Motifs such as SEK{2,5}XXC would match SEKKAEC, SEKKKAEC... SEKKKKKAEC.
3. Either/or motifs: Certain amino acid residues or nucleotides in motifs can be specified using the '|' operator. For instance, AA(B|C)DE will match AABDE as well as AACDE. B and C can also be replaced by complex motifs to form a motif of the form AA(SEKXXAF)|(SEKP{2,4}DFX)DE.
4. Start of Sequence motifs: If the motifs are prefixed by ^, the match will be performed at the start of the sequence. Example, ^CDG will match only a CDG occurring at the start of the sequence and nowhere else in the string.
5. End of Sequence motifs: The motifs that are suffixed by $ will be matched only at the end of the sequence.
6. Class motifs: For motifs in which amino acid residues or nucleotides are enclosed in square brackets, the method will match any of them in any order against the sequence. For example, ABC[EFGH] will match ABCEFGH, ABCEFHG, ABCE, ABCGH etc.
7. Negative class motifs: If ^ is prefixed to the characters that are inside the [ ], SSMBS will ignore all matches of substrings that have the characters placed in [ ]. For example, ABC[^EFGH] will match ABC, ABCD but not ABCE i.e. all substrings beginning with ABC and not ending with E or F or G or H.
8. Multiple motifs can also be combined to form a single motif and searched accordingly. For instance, a motif of the form AATAX{3,10}GACATTX{20,30} TCACTG will attempt to match three smaller motifs in the order $motif_1$=AA TAX, $motif_2$=GACATT and $motif_3$=TCACTG such that 3-10 nucleotides separate $motif_1$ and $motif_2$, and 20-30 nucleotides separate $motif_2$ and $motif_3$.
9. Motifs with hydrophobic or polar residues: Hydrophobic or polar residues can be substituted by the single characters B or Z respectively.

3.2 Case Study: Members of the Haloacid Halogenase HAD Family

Based on the presence of three sequentially separated motifs, DXXX(V|T), (S|T)GX, KX_{16-30}(G|S)(D|S)XXX(D|N), protein sequences can be categorized

to belong to the HAD family of proteins [10],[7]. Thus, the proposed method was executed over a set of 15 protein sequences that belong to the enzyme trehalose-6-phosphatase. A sample of the output generated by SSMBS is shown below.

```
------------------------------------------------------------------------
Input FileName              :   fasta.txt
No of motifs to be searched :   3
Motif 1                     :   DXXX(V|T)
Motif 2                     :   (S|T)GX
Motif 3                     :   KX{16,30}(G|S)(D|S)XXX(D|N)
OutPut FileName             :   filename1.doc
------------------------------------------------------------------------
OUTPUT OF SSMBS
------------------------------------------------------------------------
>1L6R:A|PDBID|CHAIN|SEQUENCE
Motif positions for occurrence number: 1
(DGNLT: 13 to 17)
(SGN: 45 to 47)
(KAFAVNKLKEMYSLEYDEILVIGDSNND: 154 to 181)
Position of motifs with intermediate residues for occurrence number: 1
(DGNLT: 13 to 17)
(DRDRLISTKAIESIRSAEKKGLTVSLL)
(SGN: 45 to 47)
(VIPVVYALKIFLGINGPVFGENGGIMFDNDGSIKKFFSNEGTNKFLEEMSKRTSMRSILTNRWREASTG
FDIDPEDVDYVRKEAESRGFVIFYSGYSWHLMNRGED)
(KAFAVNKLKEMYSLEYDEILVIGDSNND: 154 to 181)
------------------------------------------------------------------------
Motif positions for occurrence number: 2
(DGNLT: 13 to 17)
(TGF: 115 to 117)
(KAFAVNKLKEMYSLEYDEILVIGDSNND: 154 to 181)
Position of motifs with intermediate residues for occurrence number: 2
(DGNLT: 13 to 17)
(DRDRLISTKAIESIRSAEKKGLTVSLLSGNVIPVVYALKIFLGINGPVFGENGGIMFDNDGSIKKFFSN
EGTNKFLEEMSKRTSMRSILTNRWREAS)
(TGF: 115 to 117)
(DIDPEDVDYVRKEAESRGFVIFYSGYSWHLMNRGED)
(KAFAVNKLKEMYSLEYDEILVIGDSNND: 154 to 181)
------------------------------------------------------------------------

------------------------------------------------------------------------
Total number of occurrences: 5
------------------------------------------------------------------------
>1L6R:B|PDBID|CHAIN|SEQUENCE
Motif positions for occurrence number: 1
(DGNLT: 13 to 17)
(SGN: 45 to 47)
(KAFAVNKLKEMYSLEYDEILVIGDSNND: 154 to 181)
Position of motifs with intermediate residues for occurrence number: 1
(DGNLT: 13 to 17)
```

```
(DRDRLISTKAIESIRSAEKKGLTVSLL)
(SGN: 45 to 47)
(VIPVVYALKIFLGINGPVFGENGGIMFDNDGSIKKFFSNEGTNKFLEEMSKRTSMRSILTNRWREASTG
FDIDPEDVDYVRKEAESRGFVIFYSGYSWHLMNRGED)
(KAFAVNKLKEMYSLEYDEILVIGDSNND: 154 to 181)
----------------------------------------------------------------------
**********************************************************************
93 hits were found in 15 sequences.
**********************************************************************
----------------------------------------------------------------------
Total sequences in file         :   15
Running on machine              :   igraph9.physics.iisc.ernet.in
Program stated at (h:m:s:ms)    :   3:40:6:559 on 2/3/07 3:40 AM
Program stop at   (h:m:s:ms)    :   3:40:6:994 on 2/3/07 3:40 AM
Executed Time     (h:m:s:ms)    :   0:0:0:435
----------------------------------------------------------------------
```

The output reports occurrences of the three motifs in each of the 15 sequences in the particular order as specified. This is in accordance with the results published in literature [12]. Hence, it can be concluded that all of the 15 sequences belong to the HAD family.

This test, however, could not be run directly on the PIR pattern search as three different motifs are be specified simultaneously, for which there is no provision on the web-server. On checking PROSITE for HAD, haloacid halogenase and combinations thereof, no signature motifs were found that could be used to provide a pattern to the PIR search.

3.3 Case Study: Transcription Activation of CRP in *E.coli*

The proposed method was tested to run over the genome sequence of *E.coli* to locate the occurrences of the CRP binding complex. According to the literature [2], the consensus for the activating regions of the CRP protein is given by the sequence S_1 = TGTGAX{5,7}TCACA. The whole complex inclusive of the CRP with the core promoter sites is specified by the consensus sequence S_2 = TGTGAX{5,7}TCACAX{15,23}TATAA [2]. SSMBS located 28 identical matching occurrences of S_1 and a single identical occurrence of S_2 in the genome sequence. A section of the output for S_2 search is shown below.

```
----------------------------------------------------------------------
No of motifs to be searched :   1
Motif 1             :   TGTGAX{5,7}TCACAX{15,23}TATAA
----------------------------------------------------------------------
OUTPUT OF SSMBS
----------------------------------------------------------------------
>gi|49175990|ref|NC_000913.2| Escherichia coli K12, complete genome
Motif positions for occurrence number: 1
...CAATCTTTA
(TGTGATACAAATCACATAAATACCCCTTTAATGTTATAA: 1986066 to 1986104)
AAATGATAAT...
```

```
**********************************************************************
1 hit(s) was found in 1 sequence(s).
**********************************************************************
Running on machine              :   igraph9.physics.iisc.ernet.in
Program stated at (h:m:s:ms)    :   21:39:12:101 on 2/6/07 9:39 PM
Program stop at   (h:m:s:ms)    :   21:39:13:585 on 2/6/07 9:39 PM
Executed Time     (h:m:s:ms)    :   0:0:1:484
----------------------------------------------------------------------
```

3.4 Case Study: Zinc Finger Binding Motif

In order to compare SSMBS with with the PIR Pattern Search in terms of speed and accuracy, the extremely well knowm Zinc Finger Binding Motif HX_3 HX_{23} CXXC was considered. SSMBS was used to search for this motif in the 90% non-redundant dataset of PDB chains containing 14,423 chains. It found 33 hits in 33 sequences in 7 seconds. A section of the output for the search from SSMBS is shown below.

```
----------------------------------------------------------------------
No of motifs to be searched :   1
Motif 1             :   HX{3}HX{23}CXXC
----------------------------------------------------------------------
OUTPUT OF SSMAS
----------------------------------------------------------------------
>1jrx_B mol:protein length:571     Flavocytochrome C
Motif positions for occurrence number: 1
...EVAETTKHE(HYNAHASHFPGEVACTSCHSAHEKSMVYCDSC: 54 to 85)HSFDFNMPYA...
----------------------------------------------------------------------
Total number of occurrences: 1
----------------------------------------------------------------------
>1wjd_B mol:protein length:55     Hiv-1 Integrase
Motif positions for occurrence number: 1
...DGIDKAQEE(HEKYHSNWRAMASDFNLPPVVAKEIVASCDKC: 12 to 43)QLKGEAMHGQ...
----------------------------------------------------------------------

----------------------------------------------------------------------
Total number of occurrences: 1
----------------------------------------------------------------------
**********************************************************************
33 hits were found in 33 sequences.
**********************************************************************
----------------------------------------------------------------------
Running on machine              :   igraph9.physics.iisc.ernet.in
Program stated at (h:m:s:ms)    :   7:28:58:273 on 6/14/08 7:29 AM
Program stop at   (h:m:s:ms)    :   7:29:5:279 on 6/14/08 7:29 AM
Executed Time     (h:m:s:ms)    :   0:0:7:6
----------------------------------------------------------------------
```

3.5 Case Study: Eukaryotic DNA Topoisomerase II

Following the results of the previous case study, another was carried out with the signature motif of the eukaryokic DNA Topoisomerase II protein: (L|I|V|M|A) R_{0-1} EG(D|N)SAF_{0-1} (S|T|A|G). A single sample output is shown, where the source file is the same as the earlier case study.

```
----------------------------------------------------------------------
No of motifs to be searched :   1
Motif 1             :   (L|I|V|M|A)R{0,1}EG(D|N)SAF{0,1}(S|T|A|G)
----------------------------------------------------------------------
OUTPUT OF SSMAS
----------------------------------------------------------------------
>1z0w_A mol:protein length:207     Putative Protease La Homolog Type
Motif positions for occurrence number: 1
...IQFVGTYEG(VEGDSAS: 91 to 97)ISIATAVISA...
----------------------------------------------------------------------
Total number of occurrences: 1
----------------------------------------------------------------------
**********************************************************************
25 hits were found in 25 sequences.
**********************************************************************
----------------------------------------------------------------------
Running on machine              :   igraph9.physics.iisc.ernet.in
Program stated at (h:m:s:ms)    :   8:6:56:702 on 6/14/08 8:07 AM
Program stop at   (h:m:s:ms)    :   8:7:4:77 on 6/14/08 8:07 AM
Executed Time     (h:m:s:ms)    :   0:0:8:375
----------------------------------------------------------------------
```

The same searches for the last two case studies (outlined in sections 3.4 and 3.5) performed by the PIR pattern search over the the UniRef100 database (with its several thousand sequences) timed out after 43 and 22 minutes respectively. This search was not attempted for *E. coli* genome case study since PIR pattern search cannot be used for nucleotides. The web-server has the additional disadvantage of depending upon the internet connectivity of the user, rather than being freely available and utilized. Thus, for simple common motif searches over large databases, perhaps the SSMBS algorithm is easier to use.

4 Implementation

SSMBS requires three input: a file of protein or nucleotide sequences in FASTA format, the number of motifs to be searched and the motifs of interest. The program will generate a detailed output containing the location of the motifs and the residues which separate the motifs occurring in the given order. An option is also provided to the user to specify the maximum number of occurrences to be reported per sequence. This is particularly helpful in case this method reports a large number of occurrences for the specified motifs. The number of motifs that can be detected in a particular sequence is restricted to five due to

the high time complexity of the method for more motifs. A standalone version of SSMBS can be obtained upon request by sending an e-mail to Dr. K. Sekar (sekar@serc.iisc.ernet.in or sekar@physics.iisc.ernet.in). We plan to create a web-based computing server to locate the sequentially separated motifs in various biological sequence databases such as SWISS-PROT, PDB, PIR and Genome Database.

The SSMBS method has been implemented using JAVA since it has an in-built garbage collector that works with commendable efficiency. It improves the performance of the program by releasing occupied portions of the memory that are no more in use during run time. Since JAVA is also a platform independent language, the program can be executed on any operating system. The program has been successfully tested on Microsoft Windows (XP), Linux (Red Hat 9.0) and Sun Solaris.

5 Conclusion

Sequentially Separated Motifs in Biological Sequences (SSMBS) is a motif localization method used to locate user-defined motifs in both nucleotide and protein sequences. It has been developed to provide a comprehensive solution to the task of locating sequence motifs occurring in a particular order in large biological sequence databases. The method also provides the option for the user to specify motifs using regular expressions. By default, the method locates all the overlapping occurrences of the motifs. The method has the advantage of locating the ordered occurrences of up to five motifs in any user-defined database in FASTA format. It is a rapid method and clearly indicates the location and occurrence of the motifs.

Acknowledgments. The authors gratefully acknowledge the use of the Bioinformatics Centre, the Interactive Graphics Based Molecular Modeling facility and the Supercomputer Education and Research Centre. The methodology presented here is supported by a research grant provided by the Department of Information Technology, Governement of India. Part of this work is supported by the Institute-wide computational biology programme.

References

1. Hulo, N., Sigrist, C.J.A., Bairoch, A.: Recent improvements to the PROSITE database. Nucl. Acids Res. 32, D134–D137 (2004)
2. Carvalho, A.M., Freitas, A.T., Oliveira, A.L., Sagot, M.: An Efficient Algorithm for the Identification of Structured Motifs in DNA Promoter Sequences. IEEE/ACM Transactions on Computational Biology and Bioinformatics 03, 126–140 (2006)
3. Cartharius, K., Frech, K., Grote, K., Klocke, B., Haltmeier, M., Klingenhoff, A., Frisch, M., Bayerlein, M., Werner, T.: MatInspector and beyond: promoter analysis based on transcription factor binding sites. Bioinformatics 21, 2933–2942 (2005)

4. Wingender, E., Chen, X., Fricke, E., Geffers, R., Hehl, R., Liebich, I., Krull, M., Matys, V., Michael, H., Ohnhaeuser, R., Prueb, M., Schacherer, F., Thiele, S., Urbach, S.: Match - a tool for searching transcription factor binding sites in DNA sequences. Nucl. Acids Res. 29, 281–283 (2001)
5. Akiyama, Y.: TFSEARCH: Searching Transcription Factor Binding Sites, http://www.rwcp.or.jp/papia/
6. Werner, T.: Model for prediction and recognition of eukaryotic promoters. Mammalian Genome 10, 168–175 (1999)
7. Wang, W., Kim, R., Jancarik, J., Yokota, H., Kim, S.H.: Crystal structure of phosphoserine phosphatase from Methanococcus jannaschii, a hyperthermophile, at 1.8 A resolution. Structure 9, 65–71 (2001)
8. VanHelden, J., André, B., Collado-Vides, J.: Extracting Regulatory Sites from the Upstream Region of Yeast Genes by Computational Analysis of Oligonucleotide Frequencies. J. Mol. Biol. 281, 827–842 (1998)
9. Pavlidis, P., Furey, T.S., Liberto, M., Haussler, D., Grundy, W.N.: Promoter region-based classification of genes. In: Proceedings of the Pacific Symposium on Biocomputing, pp. 151–163 (2001)
10. Collet, J.F., Stroobant, V., Pirard, M., Delpierre, G., Schaftingen, E.V.: A new class of phosphotransferases phosphorylated on an aspartate residue in an amino-terminal (DXDX(T/V)) motif. J. Biol. Chem. 273, 14107–14112
11. Smith, T.F., Waterman, M.S.: Identification of common molecular subsequences. J. Mol. Biol. 147, 195–197 (1981)
12. Rao, K.N., Kumaran, D., Swaminathan, S.: Crystal structure of trehalose-6-phosphate phosphatase-related protein: Biochemical and biological implications. Protein Sci. 15, 1735–1744 (2006)
13. Altschul, S.F., Gish, W., Miller, W., Myers, W.E., Lipman, D.J.: Basic local alignment search tool. J. Mol. Biol. 215, 403–410 (1990)
14. Nevill-Manning, C.G., Wu, T.D., Brutlag, D.L.: Highly specific protein sequence motifs for genome analysis. JOURNAL NAME HERE 95, 5865–5871 (1998)
15. Ben-Hur, A., Brutlag, D.: Remote homology detection: a motif based approach. Bioinformatics 19, i26–i33 (2003)
16. Russ Overbeek: scan_for_matches, http://iubio.bio.indiana.edu/soft/molbio/pattern/scan_for_matches
17. Dsouza, M., Larsen, N., Overbeek, R.: Searching for patterns in genomic data. Trends Genet. 13, 497–504 (1997)
18. Pesole, S., Liuni, S., D'Souza, M.: PatSearch: a pattern matcher software that finds functional elements in nucleotide and protein sequences and assesses their statistical significance. Bioinformatics 16, 439–450 (2000)
19. Huang, J.Y., Brutlag, S.: The eMOTIF Database. Nucl. Acids Res. 29, 202–204 (2001)
20. Obenauer, J.C., Cantley, L.C., Yaffe, M.B.: Scansite 2.0 Proteome-wide prediction of cell signaling interactions using short sequence motifs. Nucl. Acids Res. 31, 3635–3641 (2003)
21. MOTIF SCAN, http://myhits.isb-sib.ch/cgi-bin/motif_scan
22. Bateman, A., Coin, L., Durbin, R., Finn, R.D., Hollich, V., Griffiths-Jones, S., Khanna, A., Marshall, M., Moxon, S., Sonnhammer, E.L.L., Studholme, D.J., Yeats, C., Eddy, S.R.: The Pfam protein families database. Nucl. Acids Res. 32, D138–D141 (2004)

Predicting SUMOylation Sites

Denis C. Bauer, Fabian A. Buske, and Mikael Bodén

Institute for Molecular Bioscience, University of Queensland, Brisbane, Qld. 4072, Australia
d.bauer@imb.uq.edu.au

Abstract. Recent evidence suggests that SUMOylation of proteins plays a key regulatory role in the assembly and dis-assembly of nuclear sub-compartments, and may repress transcription by modifying chromatin. Determining whether a protein contains a SUMOylation site or not thus provides essential clues about a substrate's intra-nuclear spatial association and function.

Previous SUMOylation predictors are largely based on a degenerate and functionally unreliable consensus motif description, not rendering satisfactory accuracy to confidently map the extent of this essential class of regulatory modifications. This paper embarks on an exploration of predictive dependencies among SUMOylation site amino acids, non-local and structural properties (including secondary structure, solvent accessibility and evolutionary profiles).

An extensive examination of two main machine learning paradigms, Support-Vector-Machine and Bidirectional Recurrent Neural Networks, demonstrates that (1) with careful attention to generalization issues both methods achieve comparable performance and, that (2) local features enable best generalization, with structural features having little to no impact. The predictive model for SUMOylation sites based on the primary protein sequence achieves an area under the ROC of 0.92 using 5-fold cross-validation, and 96% accuracy on an independent hold-out test set. However, similar to other predictors, the new predictor is unable to generalize beyond the simple consensus motif.

1 Introduction

SUMOylation is a post-translational modification attaching a small ubiquitin-like modifier (SUMO) covalently to a target protein. It has been shown that SUMO plays an important role in many essential biological functions, such as preserving the integrity and function of intra-nuclear compartments, chromatin organization and ultimately gene regulation [9, 14]. By modifying histones, dynamically competing with acetylation and ubiquitylation, SUMOylation appears to play a pivotal role in repressing transcription. Dysfunction of the SUMOylation pathway is related to several neurodegenerative diseases in human, such as Huntington's disease [5]. The significance of the SUMO conjugation system is further underscored by the apparent conservation through evolution among eukaryotic organisms.

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 28–40, 2008.

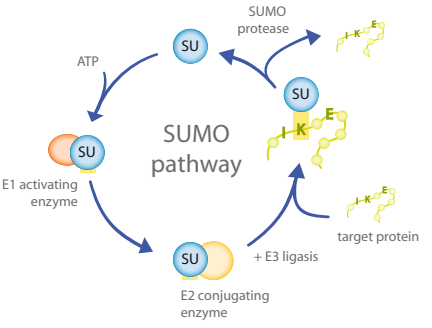


Fig. 1. SUMOylation pathway. The figure shows the role of the involved proteins in the SUMOylation pathway. E1 activates SUMO in an ATP requiring process. E2 attaches SUMO to the Lysine in the target protein, supported by E3. SUMO-protease removes SUMO from the protein, now free to be re-used in another cycle.

The SUMOylation pathway comprises four proteins: E1 activating enzyme, E2 conjugating enzyme, E3 ligase and SUMO-protease (illustrated in Fig. 1). E1 prepares SUMO for binding to the target protein (the substrate). E2 and E3 interact directly (in a concerted fashion) with the substrate at the SUMOylation site, usually conforming to the consensus motif, ΨKxE (where Ψ is a large hydrophobic residue, K is Lysine and E is Glutamic acid). E2 and E3 mediate the binding between SUMO and the central Lysine [11]. Finally, the SUMO-protease disassociates SUMO from the target protein.

Unfortunately, the motif is an unreliable predictor. Some substrates are modified on sites not matching the consensus motif [8]. Furthermore, not every consensus site in a protein is modified by SUMO. It has been suggested that there are additional factors, such as the appropriate presentation of the substrate sequence and protein sub-cellular location, which determine whether modification is completed [8].

To date, three specialized SUMOylation site predictors have been published. SUMOplot[1] is commercial. SUMOsp [17] combines two algorithms originally designed for phosphorylation site prediction (the scoring-based function GPS [19] and an iterative statistical approach MotifX [13]). SUMOpre [16] is based on a probabilistic method that optimizes the entropy of the motif.

An immediate application for *in silico* prediction is to determine the putative SUMOylation sites in the four core histones of *S. cerevisiae*. Nathan *et al.* [10] demonstrated that H2A, H2B, H3 and H4 are frequently SUMOylated. However, Nathan and colleagues were only able to experimentally identify the exact location of a fraction of the expected sites. The SUMOylation sites of the histones do not conform to the consensus motif. SUMOpre and SUMOsp both fail to predict a single SUMOylation site in protein sequences of the four histones. This exemplifies the need for a SUMOylation site predictor that captures dependencies beyond the consensus motif.

[1] http://www.abgent.com/doc/sumoplot

It is understood that SUMOylation site recognition by E2 and E3 depends mainly on the amino acid composition in the immediate neighbourhood of the central Lysine. However, it is unclear (1) if there are relevant dependencies between central residues and surrounding residues not captured by simpler models, (2) if the site's structural presentation influences binding and the computational recognition of it, and (3) if sequence conservation can be used to improve the recognition of functional sites. In this study, we investigate the ability of two machine learning techniques in predicting SUMOylation sites. Support-Vector-Machine (SVM) [15] and Bidirectional Recurrent Neural Networks (BRNN) [2] have both been successful in incorporating a range of dependencies into biological sequence models. To evaluate the contribution of dependencies putatively relevant to SUMOylation, we explore a range of features and functions for presenting our data to these machine learning algorithms.

SVMs use kernels to map samples into a high dimensional feature space to find the best separating decision hyperplane between the two classes (by maximizing the margin between them). In this study we investigate standard vector-based kernels as well as sequence-adapted kernels, including the string P-kernel [7] and the local alignment kernel [12], all acting on a fixed sequence-window around Lysine residues.

In the BRNN, the sequence input is instead fed iteratively into a network of interconnected nodes with feedback connections incorporating a trace of past sequence inputs. A BRNN is thus capable of accounting for sequence information beyond that of a current input (here coming from both a downstream and an upstream direction). The BRNN uses a gradient-based learning algorithm [2], which involves updating network "weights" to minimize the difference between predicted and target values.

We investigate the usefulness of secondary structure (SS) and solvent accessibility (SolvAcc) for SUMOylation site recognition. Unfortunately, experimentally resolved structures are available for only a fraction of known SUMOylated proteins, hence both SS and SolvAcc are obtained from predictors. We use the continuum secondary structure predictor, CSSP (with a reported $Q_3 = 77\%$) [3] and the solvent accessibility predictor, ASAP (with a correlation coefficient of 0.69) [18].

The present paper is organized as follows. First, we give an overview of the SUMOylation sites and analyse their distribution in our dataset. Second, we investigate the abilities of the different machine learning approaches when applied directly on the primary data and then with additional features. In the last section, we compare the best model with previous predictors, SUMOplot, SUMOsp and SUMOpre.

2 Methods

2.1 Dataset

This study uses the dataset of Xu *et al.* [16] only containing proteins with at least one SUMOylation site. Using the same strategy as Xu *et al.* for dividing

the data results in 144 proteins used for training and testing, and 14 proteins set aside for final validation.

The 144 proteins contain a total of 241 validated SUMOylation sites, which collectively form the positive class. Roughly 68% of the SUMOylation sites contain the consensus motif. The set of 5,741 Lysines which are not modified by SUMO form the negative class. The 13 proteins in the hold-out set contain 27 sites of which 48% match the consensus. Noteworthy, the resulting dataset is strongly unbalanced and could bias the method to prioritize the larger (negative) class. Steps are taken to investigate any effects of this imbalance.

Redundancy reduction of sequence similarity is not performed. Standard redundancy reduction targets the overall sequence similarity within a dataset and does not reduce the similarity of the relatively short SUMOylation sites.

When a numerical encoding is required (e.g. when using vector-based kernels), each amino acid in the sequence is represented by a one-hot bit vector ("plain") or the position-specific score profile produced by psi-Blast [1] for the protein ("profile"). The "plain" encoding is neutral in that no similarities are incorporated *a priori*. The "profile" encoding reflects the evolutionary divergence between homologous proteins, making available information about sequence conservation. Such "profiles" have found great utility for predicting structural features from sequence. In either case, the full sequence is represented by concatenating the position-specific vectors.

We apply CSSP (using default setting) to predict the secondary structure from primary sequence. The secondary structure is represented by the probability of a residue to adopt each of the three considered classes (helix, sheet, coil). ASAP provides predicted residue-wise relative solvent accessibility (using default settings). The predicted value is normalized to range between zero and one (with one indicating a maximally exposed residue). In either case, each residue-wise prediction is concatenated to the "plain" or "profile" encoding.

2.2 Cross-Validation and Evaluation

We evaluate every predictor configuration using 5-fold cross validation, where the dataset is randomly divided into five subsets. All but one of the five are used for training with the remaining one used for testing. This routine is repeated until all five subsets have been used for testing exactly once. In most cases, each evaluation is then repeated five times, with averages and standard deviations reported. To evaluate the performance we compare the predictions with the known positives and report on the correlation coefficient (CC), the sensitivity (SN), specificity (SP), and, the area under the ROC (AUC) (see e.g. [6] for standard definitions). Only the AUC is not influenced by the arbitrary setting of a specific classification threshold and we thus use this as the primary measure. The large number of negatives makes it easy to reach high specificity by simply predicting all but a few certain as negatives. We do revert to CC, SN and SP to discuss specific issues and to compare with previous results.

Finally, trying a large number of configurations and selecting parameter values on basis of test results will impart some selection bias. We therefore report and rely on results for the hold-out set, which has not influenced any predictor settings.

2.3 BRNN

A BRNN first centered on a particular position in a sequence (in our case this is always a Lysine). Then, in an iterated fashion it processes w_n residues on the N-terminal side and w_c residues on the C-terminal side, from both flanks and working towards the centre (in steps of w_n and w_c residues, respectively). The hidden nodes in this network are divided into two "wheels", serving as feedback modules in the N-terminal and C-terminal direction, respectively. Each wheel is equipped with a specified number of nodes, effectively controlling the trace of input from the flanks. The influence decays with the distance to the centered Lysine.

Tuning the internal weights of the BRNN is an iterative process, requiring many passes through the training set. With an independent test set left for the final evaluation, we monitor the performance on the cross-validation test set of each fold and stop training when the performance starts to deteriorate.

The unbalanced dataset could potentially also compromise performance. In addition to the original training set we create a balanced set by sampling positive and negative training data with equal probability. However, during testing all positives and negatives from the test set in the particular fold are evaluated.

2.4 SVM and Kernels

To train the SVM we extract a sequence window covering w_n residues towards the N-terminus of the protein and w_c residues to the C-terminus surrounding every Lysine in the dataset. To account for the imbalance of the dataset, we evaluate the influence of class-specific soft margin parameters, C_+ and C_-, for positives and negatives, respectively.

Apart from window size and C-values, the performance also depends on the choice of the kernel. Here, we evaluate five different kernels, the three standard kernels: linear, radial basis function (RBF) and polynomial kernel, all requiring numerical input ("plain" or "profile" encoding) and two sequence based kernels which operate directly on the sequence data in the window.

Haussler proposed a string kernel known as the string P-kernel that probabilistically evaluates (by convolution) the similarity between sequences by exploring their alignment with all ancestral sequences [7]. Since we are only dealing with fixed-length ($N = w_n + 1 + w_c$) amino acid sequences without gaps, the string P-kernel is computed as $K_P(\mathbf{x}, \mathbf{y}) = \prod_{i=1}^{N} \sum_{\alpha \in A} P(\alpha) P(x_i|\alpha) P(y_i|\alpha)$ where A is the amino acid alphabet and $\mathbf{x}$ and $\mathbf{y}$ are the two amino acid patterns being evaluated. The prior and conditional probabilities of amino acids are taken from the data used to create the BLOSUM62 substitution matrix.

In contrast, the local alignment kernel compares two sequences by exploring *all* their alignments including those with gaps [12]. An alignment between the two sequences is quantified using an amino acid substitution matrix (here BLOSUM62) and a gap penalty setting (we use the default setting). The contribution of non-optimal alignments to the final score is controlled (we use $\beta = 0.1$ which implies that many local alignments influence the result). All kernels are normalized.

3 Results

3.1 Dataset Analysis

This section illustrates the discrepancy between the dominant consensus motif and alternative SUMOylation sites.

165 out of the 241 sites in the training set have the consensus motif of ΨKxE. The motif seems to be direction dependent, reading in direction of the C-terminus. However, there are four validated SUMOylation sites which show the reverse motif. As shown in Tab. 4 a simple regular expression parser for the consensus motif can achieve a CC of 0.68 – exceeding the 0.64 reported for SUMOpre – by identifying the 165 SUMOylated sites containing the consensus motif and missing 76. However, it wrongly predicts 88 sites to be SUMOylated. It should be noted that on a proteomic scale the dataset contains an unrealistically high proportion of SUMOylation sites so the estimates are optimistic.

The difficulty of discriminating between SUMOylated and non-SUMOylated sites on basis of the consensus is illustrated in Fig. 2**a-b** using sequence Logos of both positives and negatives that match the motif [4]. A Logo of the known SUMOylation sites not matching the consensus motif is shown in Fig. 2**c**. The

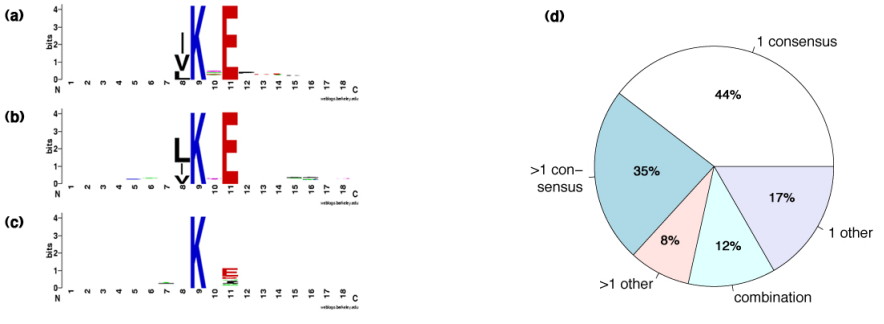


Fig. 2. Comparison between the sequence Logos of SUMOylated and non-SUMOylated sites as well as site distribution in the dataset. Panel **a** shows the sequence Logo created from 165 SUMOylated sites containing the consensus motif (positive class). Panel **b** shows the Logo of 88 non-SUMOylated sites which contain the consensus motif. Panel **c** shows the Logo of the remaining 76 non-consensus sites of the positive class. Panel **d** shows a pie diagram of the SUMOylation distribution in the dataset.

central Lysine is still predominantly flanked by Glutamic acid (E) on the C-terminal side, however the N-terminal hydrophobic residue is missing.

Fig. 2d shows the distribution of consensus vs non-consensus SUMOylation sites in the dataset of 144 proteins. 56% of the proteins have a single SUMOylation site only of which two thirds are consensus sites. A similar ratio can be observed for proteins, which contain more than one SUMOylation site. Only 12 proteins contain consensus as well as non-consensus sites. This indicates that there is no cascade effect, where the "strong" consensus site is SUMOylated first and then aids in the SUMOylation of "weaker" non-consensus sites.

3.2 Performance of BRNN

The optimal parameter setting of the BRNN was determined empirically. The window size of $w_n = 1$ and $w_c = 3$ has the highest AUC. Smaller windows give worse accuracy while larger windows do not bring any improvements. Tab. 1 summarizes the performance of several settings of hidden nodes, and on balanced and unbalanced presentation of data.

The performance is rather even across all settings. The BRNN performs slightly better when trained on the unmodified, unbalanced dataset. Increasing the number of hidden nodes appears to only decrease accuracy – suggesting that the site is simple to represent. The simplest topology with one hidden node in each wheel, trained without compensating for the class imbalance provides the best result with an average AUC of 0.93 (henceforth referred to as BRNN^{Best}). The BRNN^{Best} model contains 125 parameters to be optimized during training on approx. 4,800 samples. We do not observe a trend to overfit, which indicates a sufficient amount of training samples.

Table 1. Overview of the performance of examined BRNN settings. Average area under the ROC (AUC) of different benchmark settings for BRNN (five times repeated).

Dataset	*hidden nodes*	*AUC (sd)*
unbalanced	2	0.923 (0.006)
unbalanced	10	0.919 (0.004)
unbalanced	20	0.914 (0.007)
balanced	2	0.895 (0.012)
balanced	10	0.906 (0.007)
balanced	20	0.906 (0.010)

3.3 Performance of SVMs

In this section the performance of several SVM-settings are evaluated. Kernel, C-values and window size are problem specific and thus determined empirically. Fig. 3 exemplifies the influence of the choice of window size, as well as C-values for the linear, RBF and string P-kernel respectively. The optimal window sizes

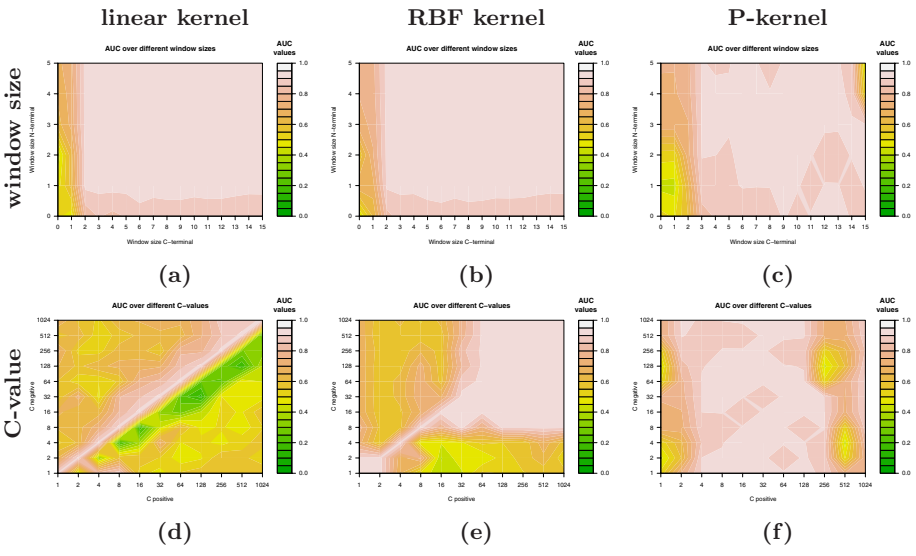


Fig. 3. Performance of different SVM settings. Each panel exemplifies the AUC on the test set for different configurations with varying window sizes (upper panels) and C-values (lower panels) for the linear, RBF and P-kernel respectively.

Table 2. Overview of the performance of the examined machine learning methods. The methods are ordered according to the average AUC, achieved by the different kernels and BRNN^{Best}. Each SVM and BRNN is represented by its best performing parameter setting regarding test error, 5-fold CV, five times repeated.

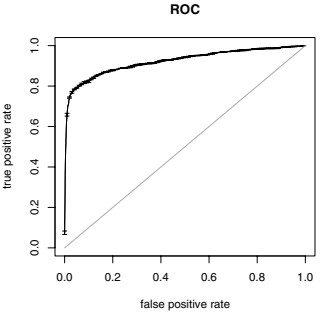
ML method	*AUC* (*sd*)	*parameter settings*				
		w_c	w_n	C_+	C_-	*method specific*
RBF kernel	0.923 (0.001)	6	4	2	1	$\sigma = 0.014$
String P-kernel	0.923 (0.004)	6	4	8	1	$\gamma = 0.1$, BLOSUM62
BRNN^{Best}	0.923 (0.006)	3	1			hidden nodes=2
Linear kernel	0.920 (0.004)	6	1	2	2	
Polynomial kernel	0.920 (0.004)	12	1	4	1	$order = 3$
Local alignment kernel	0.913 (0.002)	3	2	1	1	$\beta = 0.1$, BLOSUM62

agree with the information content visualized in the Logos (Fig. 2): while there seems to be some conservation towards the C-terminus the performance drops when more than three residues are included towards the N-terminus. The best C-values for the linear kernel put equal weight for the negative and positive classes. For the RBF and the string P-kernel there seems to be specific C-value pairs, which perform better than others.

Tab. 2 summarizes the performance of the best setting for each kernel in terms of C-values, window sizes and kernel specific parameters.

Once the optimal parameter setting is determined, all kernels seem to be able to recognize SUMOylation sites quite accurately, since the average AUC is

Table 3. Influence of evolutionary and structural features on the performance of SUMOsvm. Panel **a**: average ROC for SUMOsvm using plain encoding (five times repeated). Panel **b**: Performance of SUMOsvm with additional features input. The structural features are secondary structure (SS), solvent accessibility (SolvAcc) or both. Evolutionary features are psi-Blast profiles.



(a) ROC of SUMOsvm

encoding	*AUC*
Plain	0.92
Profile	0.93
SS	0.92
SolvAcc	0.92
SS+SolvAcc	0.92

(b) Enriched encoding

around 0.92 (with no statistical significant difference using t-test p-value < 0.05). The RBF and string P-kernel achieve the highest average AUC (both at 0.923) and have the same predictive power as the BRNN, albeit with a smaller standard deviation.

We choose the SVM with RBF kernel ($w_c = 6, w_n = 4, C_+ = 2, C_- = 1$) as our final predictor. Though not statistical significantly better its performance is more robust than the BRNN approach. Compared to the string P-kernel the RBF-kernel is much faster to train and test. We refer to the SVM-RBF kernel as SUMOsvm.

3.4 Assessing Enhanced Input Data and Multi-SVM Architecture

In this section we evaluate the impact of incorporating structural features and evolutionary information into the predictor, as well as combining several kernels into one "committee"-like SVM.

The results from the extended input features are summarized in Tab. 3. We observe no performance increase when incorporating secondary structure or solvent accessibility. The small increase using psi-Blast profiles is not statistically significant.

A multi-SVM committee yields no observable performance increase. An improvement in performance due to a committee-style prediction is expected only when the kernels deliver qualitatively different predictions. This is not the case here as we observe at least 90% of the false predictions are shared amongst the majority of all kernels.

3.5 Comparison and Discussion

In this section we compare SUMOsvm with the previously reported SUMOylation site predictors. In Tab. 4, we show the testing error measured on the 144

Table 4. Performance overview of the existing predictors and SUMOsvm. The values for the area under the ROC (AUC), correlation coefficient (CC), sensitivity (SN), specificity (SP) and accuracy (AC) are obtained from the original publications of SUMOpre and SUMOsp. The threshold chosen for SUMOsp was 18. *Though reported by Xu *et al.* as CV and hold-out error, the values are understand to be training error because "self-consistency test was used as the testing strategy" [16].

Method	*Validation*	*AUC*	*CC*	*SN*	*SP*	*AC*
SUMOsvm	CV	0.92	0.67	0.62	0.99	0.97
	hold-out	-	0.56	0.44	0.99	0.96
RegularExp	training	NA	0.68	0.69	0.99	0.98
	hold-out	-	0.54	0.48	0.99	0.97
SUMOpre	CV*	0.87	0.64	0.74	0.98	0.97
	hold-out*	-	0.66	0.54	1.0	0.97
SUMOsp	CV	0.73	0.26	0.83	0.93	0.93
	hold-out	-	0.37	0.61	0.93	0.91
SUMOplot	training	NA	0.48	0.80	0.93	0.90
	hold-out	-	0.35	0.57	0.93	0.91

proteins during cross-validation and the prediction error on the 14 proteins in the hold-out set. To obtain the hold-out error, we perform a voted prediction of the SVMs trained during the 5-fold cross-validation. The performance measures from the other methods are obtained from the original publications.

The comparison with other methods for predicting SUMOylation sites is complicated by the use of different validation methods. For SUMOpre, only three different test protocols are used: self-consistency (where "the SUMOylation state for each motif in the entire dataset is predicted by the rules derived from the same dataset" [16]), K-fold cross-validation and Leave-one-out cross-validation (which is identical to K-fold CV when K equals the size of the dataset minus one). The hold-out set is inspected only in the context of these protocols (all of which involve training on this set).

The AUC is not explicitly reported for SUMOpre, but here estimated from their ROC curve. Sensitivity and specificity are altered by simply changing the classification threshold. The threshold setting similarly affects the correlation between observed and predicted sites. We thus assume that all reported results are achieved when the threshold is the best possible.

SUMOsvm is not significantly better than the previously published methods, which in turn are not more powerful than a simple regular expression scan with `[LVI]K.E`. Neither the motif-flanking residues nor structural features appear to aid prediction. This begs the question how non-consensus sites are processed by SUMO.

One hypothesis is that sites are SUMOylated by different means (corresponding to different SUMOylation pathways). We would then expect that SUMOylation sites of proteins group in accordance with shared means. To identify such groups, we performed a kernel hierarchical cluster analysis, where the distances in the feature space (as seen by the RBF kernel) are used to generate a distance

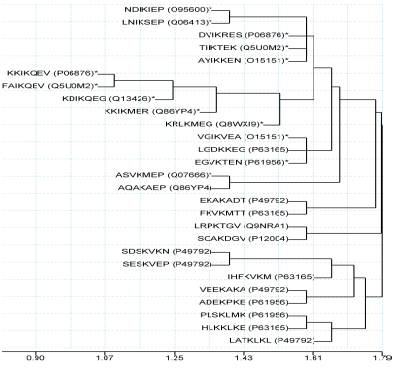


Fig. 4. Hierarchical clustering of the SUMOylation sites in the hold-out set. We use the RBF-kernel with $w_c = w_n = 3$ to obtain the hierarchical clustering plot. Sites SUMOsvm predicts correctly are marked with *.

map between the different sites. The resulting map of the SUMOylation sites in the hold-out set is shown in Fig. 4. The correctly predicted sites (all conform to the consensus motif) are clustered and form the largest entity. There is only one other cluster formed containing a putative KxK motif in the hold-out set.

To investigate if SUMOylation binding is species or compartment dependent, we extracted all proteins in the dataset that belong to human and are localized to the nucleus. If the SUMOylation pathway is species and/or compartment dependent, one would expect to see a correlation of either with sequence motif. However, a similar fraction of the consensus motif appears amongst human nuclear proteins as in the original set, and no alternative motifs were obvious when Logos were used from this smaller group of binding sites. Also, no performance gain could be observed when retraining on this subset.

4 Conclusion

We developed a SUMOylation site predictor, SUMOsvm, based on support vector classification and the RBF kernel. Several other configurations performed equally well including models based on alternative kernels and the bidirectional recurrent neural network. However, in the comparison to previously published SUMOylation site predictors we found that neither SUMOsvm nor the previously published methods are significantly better than a simple regular expression scanner.

The disappointing result is particularly noteworthy because we presented SUMOsvm with sequence data which were enriched with predicted structural features (secondary structure and relative solvent accessibility) and evolutionary information (psi-Blast profiles).

No predictor to date is able to identify the SUMOylation sites in the four core histones of yeast–a group of proteins which are known to be regulated by SUMO

but for which we still have only partial understanding of actual sites involved. All predictors tend to rely on the consensus motif that describe a majority of known SUMOylated sites but do not include the sites on the histones. Until more of the SUMOylation pathway is uncovered, SUMOylation site prediction from the current paucity of data remains challenging.

Acknowledgment

The authors would like to thank Jialin Xu and colleagues for providing us with the SUMOylation dataset.

References

1. Altschul, S.F., Madden, T.L., Schäffer, 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 Res. 25(17), 3389–3402 (1997)
2. Baldi, P., Brunak, S., Frasconi, P., Soda, G., Pollastri, G.: Exploiting the past and the future in protein secondary structure prediction. Bioinformatics 15, 937–946 (1999)
3. Bodén, M., Yuan, Z., Bailey, T.L.: Prediction of protein continuum secondary structure with probabilistic models based on NMR solved structures. BMC Bioinformatics 7, 68 (2006)
4. Crooks, G.E., Hon, G., Chandonia, J.-M., Brenner, S.E.: WebLogo: a sequence logo generator. Genome Res. 14(6), 1188–1190 (2004)
5. Dorval, V., Fraser, P.E.: SUMO on the road to neurodegeneration. Biochim Biophys Acta. 1773(6), 694–706 (2007)
6. Fawcett, T.: Roc graphs: Notes and practical considerations for researchers. Machine Learning (2004)
7. Haussler, D.: Convolution kernels on discrete structures. Technical Report UCSC-CRL-99-10, Department of Computer Science, University of California, Santa Cruz, CA 95064 (1999)
8. Hay, R.T.: SUMO: a history of modification. Mol Cell 18(1), 1–12 (2005)
9. Heun, P.: SUMOrganization of the nucleus. Curr Opin Cell Biol. 19(3), 350–355 (2007)
10. Nathan, D., Ingvarsdottir, K., Sterner, D.E., Bylebyl, G.R., Dokmanovic, M., Dorsey, J.A., Whelan, K.A., Krsmanovic, M., Lane, W.S., Meluh, P.B., Johnson, E.S., Berger, S.L.: Histone sumoylation is a negative regulator in saccharomyces cerevisiae and shows dynamic interplay with positive-acting histone modifications. Genes Dev. 20(8), 966–976 (2006)
11. Rodriguez, M.S., Dargemont, C., Hay, R.T.: SUMO-1 conjugation in vivo requires both a consensus modification motif and nuclear targeting. J Biol Chem. 276(16), 12654–12659 (2001)
12. Saigo, H., Vert, J., Ueda, N., Akutsu, T.: Protein homology detection using string alignment kernels. Bioinformatics 20(11), 1682–1689 (2004)
13. Schwartz, D., Gygi, S.P.: An iterative statistical approach to the identification of protein phosphorylation motifs from large-scale data sets. Nat Biotechnol. 23(11), 1391–1398 (2005)

14. Shen, T.H., Lin, H.-K., Scaglioni, P.P., Yung, T.M., Pandolfi, P.P.: The mechanisms of PML-nuclear body formation. Mol. Cell 24(3), 331–339 (2006)
15. Vapnik, V.: Statistical learning theory. Wiley, Chichester (1998)
16. Xu, J., He, Y., Qiang, B., Yuan, J., Peng, X., Pan, X.: A novel method for high accuracy sumoylation site prediction from protein sequences. BMC Bioinformatics 9, 8 (2008)
17. Xue, Y., Zhou, F., Fu, C., Xu, Y., Yao, X.: SUMOsp: a web server for sumoylation site prediction. Nucleic Acids Res. 34, W254–W257 (2006)
18. Yuan, Z., Huang, B.: Prediction of protein accessible surface areas by support vector regression. Proteins 57(3), 558–564 (2004)
19. Zhou, F., Xue, Y., Chen, G., Yao, X.: GPS: a novel group-based phosphorylation predicting and scoring method. Biochem. Biophys. Res. Commun. 325(4), 1443–1448 (2004)

DFS Based Partial Pathways in GA for Protein Structure Prediction

Md Tamjidul Hoque[1], Madhu Chetty[2], Andrew Lewis[1], and Abdul Sattar[1]

[1] Institute for Integrated and Intelligent Systems (IIIS),
Griffith University, Nathan QLD 4108, Australia
T.Hoque@griffith.edu.au, {A.Lewis,A.Sattar}@griffith.edu.au
[2] Gippsland School of Information Technology (GSIT)
Monash University, Churchill VIC 3842, Australia
Madhu.Chetty@infotech.monash.edu.au

Abstract. Nondeterministic conformational search techniques, such as Genetic Algorithms (GAs) are promising for solving protein structure prediction (PSP) problem. The crossover operator of a GA can underpin the formation of potential conformations by exchanging and sharing potential sub-conformations, which is promising for solving PSP. However, the usual nature of an optimum PSP conformation being compact can produce many invalid conformations (by having non-self-avoiding-walk) using crossover. While a crossover-based converging conformation suffers from limited pathways, combining it with depth-first search (DFS) can partially reveal potential pathways. DFS generates random conformations increasingly quickly with increasing length of the protein sequences compared to random-move-only-based conformation generation. Random conformations are frequently applied for maintaining diversity as well as for initialization in many GA variations.

Keywords: Depth-first search, protein structure prediction, genetic algorithm, lattice model.

1 Introduction

We are seeking to solve the *ab initio* (meaning 'from the origin') or the *de novo* protein structure prediction problem [1]. In an *ab initio* approach, the building of a 3D conformation (structure) is essentially based on the properties of amino acids, where protein is a three dimensionally folded molecule composed of amino acids [2] linked together (called the primary structure) in a particular order specified by the DNA sequence of a gene. Particular folded structures are essential for the functioning of living cells as well as for providing body structure. Protein structure prediction (PSP) is a problem of determining the native state of a protein from its primary structure and is of great importance because three dimensionally folded structures determine the biological function [3] and hence proves extremely useful in applications like drug design [4].

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 41–53, 2008.

For investigating the underlying principles of protein folding, lattice protein models introduced by Dill [5] are widely used [6]. Protein conformation as a *self-avoiding walk* in the lattice model has been proven to be *NP-complete* [7, 8]. Therefore a deterministic algorithm for folding prediction is not feasible. So, a nondeterministic approach with robust strategies that can extract minimal energy conformations efficiently from these models becomes necessary. Still, this is a very challenging task as there exists an astronomical number of possible conformations even for a very short sequence of amino acids [9, 10].

We have chosen the Genetic Algorithm (GA) as a vehicle for providing solutions to the PSP problem for better performance, where crossover is regarded as the key operation of GA [11]. The core concepts of GAs and their components are often adapted by many PSP solving algorithms for the effectiveness [12-16]. While crossover can be very effective in joining two different potential sub-conformations, it can be repeatedly unsuccessful as the converging conformations (hence the sub-conformations), being compact in nature, leave limited pathways to a valid (i.e., self-avoiding-walk) conformation. This means many potential conformations may be lost, which motivates us to apply partial pathways based on depth first search (DFS) [17] to regain potential conformations, leading to effective PSP solution.

2 Background and Preliminaries

In nature, a protein folds remarkably quickly, requiring between a tenth of a millisecond and one second in general, whereas any algorithm on any modern computer is still unable to simulate this task in anything approaching similar time[11, 18]. For the immensely complex protein structure prediction problem, there are several issues and approaches which are yet to be considered [11, 19, 20]:

First, the energy function, which is a combination of several factors that determines the free energy of a folded protein, is not fully understood. Therefore, existing formulations for energy functions do not suggest any obvious path to solution of the PSP problem.

Second, conformational search algorithms are promising approaches toward this hard optimization problem, but the PSP problem still needs considerable research to find an effective algorithm. The aim of the search is to identify an optimum conformation within a huge and very convoluted search landscape.

Third, Cyrus Levinthal postulated, in what is popularly known as the Levinthal paradox, that proteins fold into their specific 3D conformations in a time-span far shorter than it would be possible for the molecule to actually search the entire conformational space (which is astronomically large) for the lowest energy state [21]. As proteins cannot, while folding, be sampling all possible conformations, therefore folding pathways must exist.

While focusing on the second issue [22-27], we are utilizing DFS strategies, developing novel search algorithms in a form to address the pathway hypothesis. It has been concluded that conformational searching is the bottleneck in protein folding prediction and the observed folding rates have been found to be proportional to the number of microscopic folding routes [28]. These routes can be captured by the crossover operation from suboptimal conformations and then partial DFS can mimic

the existing microscopic path guided by the converging sub-conformation, whereas a crossover operation alone can encounter more collisions [13] (while mating dissimilar converging conformations) before having a SAW conformation and thus often can reject the potential sub-conformation as being unfit when paired with the available counterpart of the crossover portion (from a dissimilar conformation). To determine the effect of DFS in such situations we will rely on empirical results.

2.1 The HP Lattice Model

The simplified HP lattice model [29, 30] is based on *hydrophobicity* [31], dividing the amino acids into two different beads – *hydrophobic* (H) and *hydrophilic* (or *polar* (P)). The model allows HP protein sequences to be configured as self-avoiding walks (SAW) on the lattice path favoring an energy free state due to HH interaction. The energy of a given conformation is defined as the number of *topological neighboring* (TN) contacts between those Hs, which are not adjacent in the sequence. This contact between two neighboring H residues (or HH contact) is TN and is assigned a value for the potential, termed *interaction potential* which is define as -1 for the regular HP model [32]. Further, the HP interaction and PP interaction potential value is assigned 0, which basically implies that there is no interaction between an H and a P of HP contact or between the Ps of PP contacts.

To define PSP formally, assume for an amino-acid sequence $s = s_1, s_2, s_3, \cdots, s_n$, a conformation *c* needs to be formed where $c^* \in C(s)$, $C(s)$ is the set of all valid (i.e., SAW) conformations of *s, n* is the total number of amino acids in the sequence and energy $E^* = E(C) = \min\{E(c) \mid c \in C\}$ [15]. If the number of TNs (for HH contact) in a conformation *c* is *q* then the value of $E(c)$ is defined as $E(c) = -1 \times q = -q$ and the *fitness function* is $F = -q$. The optimum conformation will have a maximum possible value of |F|. In a 2D HP square lattice model (Figure 1), a non-terminal and a terminal residue, each with 4 neighbours, can have a maximum of 2 TNs and 3 TNs, respectively. In this paper, we will confine ourselves to using the 2D HP square lattice model only, as this model will be sufficient for our needs. However, its simplicity may encourage interested readers to do further research, which would otherwise be very difficult. The HP lattice model is also very popular with the research community [11, 23, 29, 30, 33-39], since it allows easy development, validation and comparison of new techniques for protein structure prediction (PSP) [22-24, 26, 27, 40].

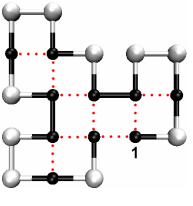


Fig. 1. HP conformation in the 2D HP model shown by a solid line. 2D square lattice having fitness = - (TN Count) = -9. ● indicates a hydrophobic and ● indicates a hydrophilic residue. The dotted line indicates a TN. Starting residue is indicated by a '1' in the figure.

2.2 Complexity of the Lattice Model

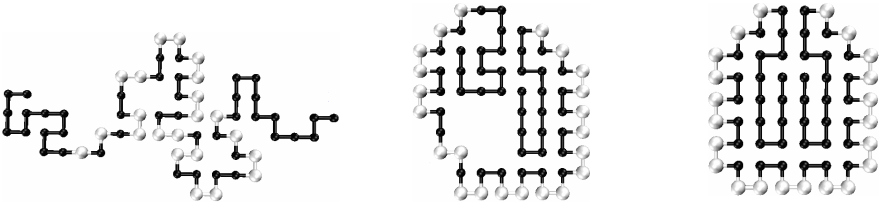
Even if we use this simplified model we have an inordinate number of valid (i.e., SAW) conformations, even for a shorter sequences [9, 10, 41]. For instance, for a sequence of n amino acids, the number of valid conformations is proportional to μ^n, where the connective constant or the effective coordinate number μ, is lattice dependent [10]. Prediction of the optimal conformation using the lattice model is also an *NP-complete* problem [7, 8]. To predict the backbone conformation of the folded protein from its amino acid sequence based on global interactions such as *hydrophobicity*, lattice models are used for approximation [29, 30, 33-35]. For *ab initio* prediction in *Critical Assessment of Structure Prediction* (CASP) [33-35], most successful approaches followed the hierarchical paradigm where the lattice-based, backbone conformational sampling works very effectively at the top of the hierarchy. With further advancement toward all-atom or full modeling from the lattice, the energy functions include atom-based potentials from molecular mechanics packages such as CHARMM, AMBER, ECEPP and so on [42, 43]. Conformational search algorithms built on lattice models, which play a key role in solving PSP, are discussed next.

2.3 Nondeterministic Conformational Search Algorithms

For solving *ab initio* PSP using the lattice model numerous nondeterministic approaches have been investigated: *Monte Carlo* (MC) simulation, *Evolutionary* MC (EMC) [12, 13], *Simulated Annealing* (SA), *Tabu Search* with *Genetic Algorithm* (GTB) [14], *Ant Colony Optimisation* [15], and *Immune Algorithm* (IA) based on *Artificial Immune System* (AIS) [44]. Due to their simplicity and search effectiveness, *Genetic Algorithms* (GAs) [11, 26, 32, 45-48] are the most attractive. They also provided superior performance over MC [46, 47]. The concepts of GAs are also widely adapted within these algorithms. For instance, a new MC algorithm [12] adopted the population-based cut-and-paste (i.e. crossover) operation to achieve higher fitness. The evolutionary Monte Carlo (EMC) [13] algorithm incorporated the evolutionary features of genetic algorithms, such as a population which is updated by crossover and mutation operations. Jiang *et al.* applied the GA with *Tabu* (GTB) search to solve PSP using lattice models [14]. Also, the *conformational space annealing* (CSA) [16, 49] algorithm is based on GA concepts, where the population is renamed as a "bank".

2.4 Focus of the Paper

Given the widespread adaptation of GAs for PSP, the heart of a GA, i.e. the crossover operation, can be made more effective by combining it with DFS which can have a significant positive impact on solving the PSP problem. In a conventional GA, since the optimum conformation is mostly compact physically (see Figure 2), a crossover-based converging conformation suffers from limited pathways and the algorithm increasingly generates invalid conformations. Our hypothesis is that the combination of depth-first search (DFS) with crossover can instead reveal potential pathways in solving PSP. Thus, a repeatedly failing crossover with a congested but potential sub-conformation can be allowed a limited number of pathways for possible candidate crossover counterparts obtained by using DFS if there exists at least one path.



(a): Fitness = -15 (b): Fitness = -33 (c): Fitness = - 42

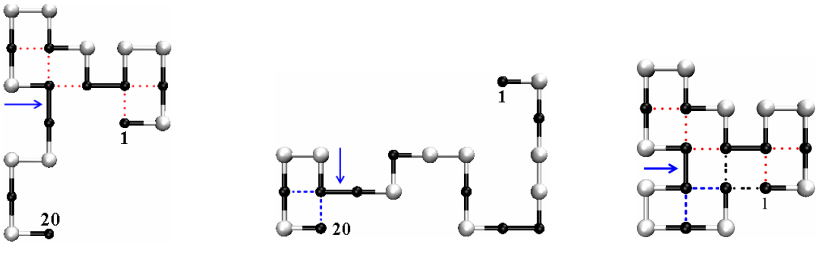
Fig. 2. As the search proceeds the conformation gets more compact: For a typical run, conformations at generation 1, 1434 and 5646 have been shown in (a), (b) and (c) respectively, showing the fitter conformation is relatively more compact.

2.5 Defining the GA Operators for PSP Problem

Here, we define the GA operators for the PSP problem based on the HP lattice model:

Crossover operation: For PSP, this aids the construction of global solutions by the cooperative combination of many local substructures [11]. We particularly followed the commonly-used crossover operation pioneered by Unger *et al.* [46], as illustrated in Figure 3, a single-point crossover. We follow this single-point crossover, since otherwise the converging conformation, being compact in nature, would generate more collisions or invalid conformations [13]. The ability to rotate before joining within the crossover, in addition, provides a mutation-equivalent operation. With the help of *relative encoding* [40], this can be seen easily. For example, if we emulate the crossover in Figure 3 without the rotation, we can write using relative encoding that:

Crossover (a:'LFLLRRLRLLFLRFRLFL', b:'RFFFRFRFLFLRFRLLFL') → would output, c': 'LFLLRRLRLLFL*RLLFL' without the rotation before joining. (Here, '*' indicates an undefined move in relative encoding but here it indicates a non-SAW move.) But, with rotation, the conformation can have SAW, i.e. c: 'LFLLRRLRLLFL**R**RLLFL'.



(a) (b) (c)

Fig. 3. An example of the crossover operation [46]. Conformations are randomly cut and pasted with the cut point chosen randomly between residues 14 and 15. The first 14 residues of (a) are rotated first as needed (as allowed by the degree of freedom by the model configuration) and then joined with the last 6 residues of (b) to form (c), where fitness, F = -9. '→' indicates crossover positions.

Comparing c':'LFLLRRLRLLFL*RLLFL'and c:'LFLLRRLRLLFL**R**RLLFL', it becomes clear that the '*' is replaced by an 'R' after the rotation, which is genotypically a single-point mutation.

Crossover failure: This implies that before joining two parts all, possible, rotated positions at the joining point have been tried but failed to produce at least one valid conformation (i.e., a SAW).

Combination of crossover and DFS: For generating a conformation this implies that a DFS-generated random and partial path has been joined with the first half of the sub-conformation.

DFS after crossover failed: This implies that 'combination of crossover and DFS' has been performed after an occurrence of 'crossover failure'.

Mutation operation: This involves *pivot rotation* (Figure 4) as basically pioneered by Unger *et al.* [46]. We employed single-point mutation to avoid more collisions.

(a) (b)

Fig. 4. An example of the mutation operation [46]. Dotted lines indicate TN. Residue number 11 is chosen randomly as the pivot. For the move to apply, a 180° rotation (among a number of possible degree of freedom defined by the model configuration) alters (a) with F = -4 to (b) F = -9. '→' indicates the mutation residue.

Ordinary random conformation generation: This implies the generation of a SAW conformation based on random-move-only (RMO). In a 2D square lattice model *Left*, *Right* and *Forward* moves are permissible but *Backward* move is prohibited. For a conformation, once a path search has failed after looking in the three possible degrees of the freedom the whole process restarts.

Random conformation generation by DFS: This implies that we apply DFS to generate a SAW conformation. As the DFS proceeds, it stores the possible pathways using a stack-memory [17] and, upon total failure after trying all possible degrees of freedom on a particular location (i.e. lattice point), it can backtrack to restart from the stored options instead of restarting the creation of the whole conformation.

3 Experiments and Results

We carried out experiments to empirically verify our hypothesis that combining DFS with crossover will be advantageous. The simple GA (SGA) applied for PSP is

1. Initialize the fixed size current population (Pop_z) of randomly generated conformations.
2. Obtain a new solution (S_{new}) from the current population by using ***Crossover*** and ***Mutation*** operations at the pre-specified rates (p_c and p_m respectively).
3. Assess the quality or fitness, *F,* of S_{new}.
4. Promote the obtained S_{new}, and elite and untouched chromosomes, to the next generation and assign the new generation as the current population.
5. IF END-OF-SOLUTION is not reached THEN repeat from Step 2.

Fig. 5. Genetic Algorithm for solving PSP problem[†]

(a)	(b)
1. DO single-point ***Crossover.*** 2. IF '***Crossover failure***' = TRUE then 3. REPLACE one of the parents. 4. DO single-point ***Crossover***. END IF	1. DO single-point ***Crossover.*** 2. IF '***Crossover failure***' = TRUE then 3. DO '***DFS after crossover failed***'. END IF
(c)	(d)
1. DO single-point '***Combination of crossover and DFS***'.	1. DO apply option: (a). 2. IF no improvement for 5 consecutive generations, 3. DO apply option: (b). END IF

Fig. 6. ***Crossover*** operation and variation details

illustrated in Figure 5 and the crossover variations with the possible implementation have been shown in Figure 6. As shown in Figure 6, we have experimented with four variations of the crossover operation. *Crossover* (a) (see Figure 6(a)) represents a conventional crossover operation for PSP without DFS. *Crossover*(b) (see Figure 6(b)) applies DFS-based partial path generation with the sub-conformation immediately the sub-conformation fails to join with its counterpart sub-conformation after trying all possible degrees of freedom. *Crossover*(d) (see Figure 6(d)) is similar to *Crossover*(b) in operation but allows more time to a failed crossover to search for a suitable counterpart sub-conformation to match. *Crossover*(c) is a the most dissimilar variation of *Crossover*(d) where, instead of a sub-conformation looking for its counterpart sub-conformation in the population, *Crossover*(c) directly uses DFS to generate the rest of the path to complete the conformation. This alternative was investigated to determine an effective rate of DFS.

The default GA parameters for all experiments were set as population size (Pop_z) to 200, crossover rate (p_c) to 0.85 or 85%, mutation rate (p_m) to 5% and for elitism the elite rate was set to 5% [50, 51].

The fold for longer PSP problems generally has complex energy landscapes [30, 52-57], and hence those sequences will take longer to converge. So we chose those longer sequences to highlight the true benefit of this approach. A maximum of 2000

[†] Terms in **bold** and *italic* are explained in section 2.5.

Table 1. Benchmark protein sequences for 2D HP model

Length	Sequences	Ref.
50	H2(PH)3PH4PH(P3H)2P4H(P3H)2PH4P(HP)3H2	[59]
60	P2H3PH8P3H10PHP3H12P4H6PH2PHP	[59]
64	H12(PH)2(P2H2)2P2HP2H2PPHP2H2P2(H2P2)2(HP)2H12	[59]
85	4H4P12H6P12H3P12H3P12H3P1H2P2H2P2H2P1H1P1H	[58]
100	3P2H2P4H2P3H1P2H1P2H1P4H8P6H2P6H9P1H1P2H1P11H2P3H1P2H 1P1H2P1H1P3H6P3H	[58]

'H' and 'P' in the sequence indicate hydrophobic and hydrophilic amino acids, respectively.

Table 2. Run results of 10 iterations on each PSP sequence (see Table 1 for the sequences). GA runs with four different crossover options (shown in Figure 6), have been compared.

Length	*X(a)*	*X(b)*	*X(c)*	***X(d)***	*CSA*	*UGA*
50	-17.3/-**20**	-17.6/-**20**	-14.5/-17	-**18**/-**20**	-17 / -19	-16.6 / -18
60	-29.2/-**32**	-29.8/-**32**	-27.8/-31	-**30.5**/-**32**	-30.4/-**32**	-29/-31
64	-29.1/-31	-29.3/-31	-25.2/-29	-**32**/-**35**	-29/-30	-27.8/-31
85	-39.4/-44	-39.6/-45	-34.5/-38	-**43.4**/-**46**	-43.2/-**46**	-41.4/-**46**
100	-37.1/-39	-37.6/-41	-30.2/-37	-**38.5**/-**42**	-37.2/-38	-37.4/-40

The format of column entries is 'Average / Minimum'. The *X* implies ***Crossover*** operation. Thus, *X*(a) indicates ***Crossover***(a) as described above, and so on. CSA and UGA indicate Conformational Space Annealing Algorithm [16] and Unger's GA [46], respectively. **Bold** entries indicate the row-wise best values obtained.

generations was allocated for each of the 10 iterations carried out per sequence, per category of experiments. Benchmark PSP sequences shown in Table 1 for the 2D square HP lattice model [5], length ranging from 50 to 100 were used [58, 59]. The results are shown in Table 2.

It may be noted that in Table 2, we include two other algorithms in their generic form: Unger's GA (UGA [46]) and Conformational Space Annealing (CSA) algorithm [16, 49] with our proposed algorithm for solving the PSP problem. UGA has already outperformed many MC variations, as reported in [11, 46]. We emulated UGA in our experiment with the same parameter for cooling, i.e. the cooling temperature was set to 2 at the start and decreased by 0.99 every 200000 steps until the temperature became 0.15.

We abstracted the general form of the CSA algorithm by removing the heuristic-based special moves, keeping the generic form intact, to provide a fair comparison in our experiment. Comparison with CSA algorithm is particularly important for our work, since the CSA approach has recently been applied in the PSP software ROSETTA [33, 60-63]. Both UGA and CSA ran 2000 GA generation equivalent runs per iteration.

4 Discussion of the Experimental Results

We have introduced the concept of finding potential partial pathways using a depth-first search (DFS) strategy when a converging, potential sub-conformation in a crossover failed to find a matching counterpart to produce a valid (i.e., having a self-avoiding-walk) conformation. Crossover variation *X*(c) has the worst result in Table 2. *X*(c) involves applying DFS constantly at the same rate as the crossover operation to generate the other half of the crossover portion, which is misguiding the optimum results more that guiding them. *X*(a) represents the crossover-only approach, that is, crossover with DFS, and *X*(b) is the variant where DFS is applied whenever a crossover fails. *X*(b) is a slight improvement over *X*(a). *X*(d) performed the best, with results comparable to the UGA and CSA algorithms. This is because, in *X*(d), crossover was applied exhaustively by allowing a failed crossover to look for more counterparts to match and when there is no improvement at all in the whole population for consecutive few generations, the failed crossover is combined with DFS to generate the possible potential pathways. It is interesting to note that, in our experiment we find *DFS has zero failure in finding pathways*. Thus, a constantly failing sub-conformation in a crossover operation, which is likely to have few possible pathways, can be salvaged using DFS to unravel the hidden paths effectively. As an alternative to DFS, breadth-first search (BFS) [17] could have been used; however, BFS is both memory and time intensive.

5 Supplementary Applications of DFS in PSP

It is important to remember that ***ordinary random conformation generation***† takes exponential time (fitted curve: $y = 2.8723\ e^{0.0326x}$ with square of coefficient of determination, $R^2 = 0.9832$) with increasing sequence length using the random-move-only

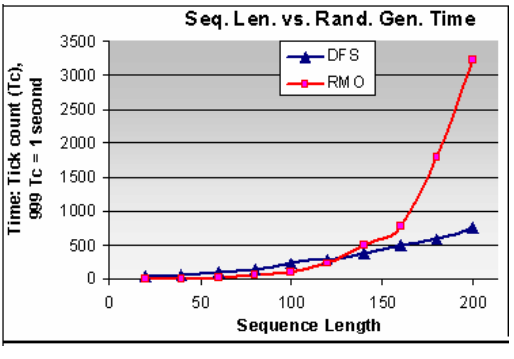


Fig. 7. Random conformation generation: DFS approach versus random-move-only (RMO) approach. An average of 100 iterations is taken for a particular length of a single random conformation generation.

† Terms in **bold** and *italic* are explained in section 2.5.

(RMO) approach. In contrast, the run-time for ***random conformation generation by DFS*** remains quadratic (fitted curve: $y = 0.02\ x^2\text{-}0.5717x\text{+}54.789$, with $R^2 = 0.9996$) (see Figure 7).

The application of ***random conformation generation by DFS*** may have a generally lower impact because totally random conformations are only generated for initialization of the population. To maintain diversity many GA approaches replenish the population a considerable amount and at frequent intervals [64, 65]. For example, Hoque *et al.* have shown removal of chromosomes having 80-90% or greater similarity from a GA population helps it to perform better [64]. After removal it is necessary to replenish the population by random conformations of 20 to 30% in each generation. Thus, in such a case, for longer sequences, ***random conformation generation by DFS*** would make the GA search far more efficient.

6 Conclusions

A depth-first search (DFS) strategy at a low rate has been applied in combination with a powerful crossover operation. Together they revealed convoluted and microscopic pathways in solving protein structure prediction problem. Experiments using a variety of longer, standard benchmark sequences from the literature have demonstrated the efficacy and improved performance characteristics of this approach. The search strategy developed was inspired by the pathway hypothesis. Further work will be directed to exploring the biological significance and relevance of this novel approach.

Acknowledgement

Support from Australian Research Council (grant no DP0557303) is thankfully acknowledged.

References

1. Chivian, D., Robertson, T., Bonneau, R., Baker, D.: AB INITIO METHODS. In: Bourne, P.E., Weissig, H. (eds.) Structural Bioinformatics. Wiley-Liss, Inc., Chichester (2003)
2. Allen, F., et al.: Blue Gene: A vision for protein science using a petaflop supercomputer. IBM System Journal 40, 310–327 (2001)
3. Pietzsch, J.: The importance of protein folding. Nature (2003) (last access, 2007), `http://www.nature.com/horizon/proteinfolding/background/importance.html`
4. Petit-Zeman, S.: Treating protein folding diseases. Nature (last access, 2008), `http://www.nature.com/horizon/proteinfolding/background/treating.html`
5. Dill, K.A.: Theory for the Folding and Stability of Globular Proteins. Biochemistry 24, 1501–1509 (1985)
6. Backofen, R., Will, S.: A Constraint-Based Approach to Fast and Exact Structure Prediction in Three-Dimensional Protein Models. Constraints Journal 11 (2006)

7. Crescenzi, P., Goldman, D., Papadimitriou, C., Piccolboni, A., Yannakakis, M.: On the complexity of protein folding (extended abstract). In: 2nd Intl. conference on Computational molecular biology, pp. 597–603. ACM, New York (1998)
8. Berger, B., Leighton, T.: Protein Folding in the Hydrophobic-Hydrophilic (HP) Model is NP-Complete. Journal of Computational Biology 5, 27–40 (1998)
9. Schiemann, R., Bachmann, M., Janke, W.: Exact Enumeration of Three – Dimensional Lattice Proteins. In: Computer Physics Communications, p. 166. Elsevier Science, Amsterdam (2005)
10. Guttmann, A.J.: Self-avoiding walks in constrained and random geometries. Elsevier, Amsterdam (2005)
11. Unger, R., Moult, J.: On the Applicability of Genetic Algorithms to Protein Folding. The Twenty-Sixth Hawaii International Conference on System Sciences 1, 715–725 (1993)
12. Bastolla, U., Frauenkron, H., Gerstner, E., Grassberger, P., Nadler, W.: Testing a new Monte Carlo Algorithm for Protein Folding. National Center for Biotechnology Information 32, 52–66 (1998)
13. Liang, F., Wong, W.H.: Evolutionary Monte Carlo for protein folding simulations. J. Chem. Phys. 115 (2001)
14. Jiang, T., Cui, Q., Shi, G., Ma, S.: Protein folding simulation of the hydrophobic-hydrophilic model by computing tabu search with genetic algorithms. ISMB, Brisbane Australia (2003)
15. Shmygelska, A., Hoos, H.H.: An ant colony optimization algorithm for the 2D and 3D hydrophobic polar protein folding problem. BMC Bioinformatics 6 (2005)
16. Lee, J.: Conformational space annealing and a lattice model Protein. Journal of the Korean Physical Society 45, 1450–1454 (2004)
17. Cormen, T.H., leiserson, C.E., Rivest, R.L.: Introduction to Algorithms. MIT Press, Cambridge (1998)
18. Toma, L., Toma, S.: Folding simulation of protein models on the structure based cubo-octahedral lattice with the Contact interactions algorithm. Protein Science 8, 196–202 (1999)
19. Baker, D.: A surprising simplicity to protein folding. Nature 405, 39–42 (2000)
20. Pande, V.S., Rokhsar, D.: Folding pathway of a lattice model for proteins. PNAS 96, 273–278 (1999)
21. Levinthal, C.: Are there pathways for protein folding? Journal of Chemical Physics 64, 44–45 (1968)
22. Hoque, M.T., Chetty, M., Dooley, L.: A Guided Genetic Algorithm for Protein Folding Prediction Using 3D Hydrophobic-Hydrophilic Model. In: IEEE CEC (2006)
23. Hoque, M.T., Chetty, M., Dooley, L.S.: Significance of Hybrid Evolutionary Computation for Ab Initio Protein Folding Prediction. In: Grosan, C., Abraham, A., Ishibuchi, H. (eds.) Hybrid Evolutionary Algorithms, vol. 75, pp. 241–268. Springer, Berlin (2007)
24. Hoque, M.T., Chetty, M., Dooley, L.S.: A Hybrid Genetic Algorithm for 2D FCC Hydrophobic-Hydrophilic Lattice Model to Predict Protein Folding. In: Sattar, A., Kang, B.-h. (eds.) AI 2006. LNCS (LNAI), vol. 4304. Springer, Heidelberg (2006)
25. Hoque, M.T., Chetty, M., Dooley, L.S.: Fast computation of the fitness function for protein folding prediction in a 2D hydrophilic-hydrophobic model. Journal published in the special issue of the International Journal of Simulation Systems, Science and Technology 6, 27–37 (2005)
26. Hoque, M.T., Chetty, M., Dooley, L.S.: A New Guided Genetic Algorithm for 2D Hydrophobic-Hydrophilic Model to Predict Protein Folding. In: IEEE CEC, Edinburgh, UK (2005)

27. Hoque, M.T., Chetty, M., Sattar, A.: Protein Folding Prediction in 3D FCC HP Lattice Model Using Genetic Algorithm Bioinformatics special session. In: IEEE CEC, Singapore (2007)
28. Ghosh, K., Ozkan, S.B., Dill, K.A.: The Ultimate Speed Limit to Protein Folding Is Conformational Searching. Journal of American Chemical Society 129, 11920–11927 (2007)
29. Lau, K.F., Dill, K.A.: A lattice statistical mechanics model of the conformational and sequence spaces of proteins. Macromolecules 22, 3986–3997 (1989)
30. 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)
31. Dill, K.A., Ozkan, S.B., Weikl, T.R., Chodera, J.D., Voelz, V.A.: The protein folding problem: when will it be solved? Current Opinion in Structural Biology 17, 246–342 (2007)
32. Corne, D.W., Fogel, G.B.: An Introduction to Bioinformatics for Computer Scientists. In: Fogel, G.B., Corne, D.W. (eds.) Evolutionary Computation in Bioinformatics, pp. 3–18 (2004)
33. Baker, D.: Prediction and design of macromolecular structures and interactions. Phil. Trans. R. Soc. B361, 459–463 (2006)
34. Schueler-Furman, O., Wang, C., Bradley, P., Misura, K., Baker, D.: Progress in Modeling of Protein Structures and Interactions. Science 310, 638–642 (2005)
35. Xia, Y., Huang, E.S., Levitt, M., Samudrala, R.: Ab Initio Construction of Protein Tertiary Structures using a Hierarchical Approach. J. Mol. Biol. 300, 171–185 (2000)
36. Backofen, R., Will, S., Clote, P.: Algorithmic approach to quantifying the hydrophobic force contribution in protein folding. Pacific Symp. On Biocomputing 5, 92–103 (2000)
37. Yue, K., Dill, K.A.: Sequence-structure relationships in proteins and copolymers. Phys. Rev. E48, 2267–2278 (1993)
38. Toma, L., Toma, S.: Contact interactions methods: A new Algorithm for Protein Folding Simulations. Protein Science 5, 147–153 (1996)
39. Bornberg-Bauer, E.: Chain Growth Algorithms for HP-Type Lattice Proteins. In: RECOMB, USA (1997)
40. Hoque, M.T., Chetty, M., Dooley, L.S.: Non-Isomorphic Coding in Lattice Model and its Impact for Protein Folding Prediction Using Genetic Algorithm. In: IEEE CIBCB, Toronto, Canada (2006)
41. Chen, M., Lin, K.Y.: Universal amplitude ratios for three-dimensional self-avoiding walks. Journal of Physics A: Mathematical and General 35, 1501–1508 (2002)
42. Roterman, I.K., Lambert, M.H., Gibson, K.D., Scheraga, H.: A comparison of the CHARMM, AMBER and ECEPP potentials for peptides. II. Phi-psi maps for N-acetyl alanine N'-methyl amide: comparisons, contrasts and simple experimental tests. J. Biomol. Struct. Dynamics 7, 421–453 (1989)
43. Cornell, W.D., Cieplak, P., Bayly, C.I., Gould Jr., I.R., Merz Jr., K.M., Ferguson, D.M., Spellmeyer, D.C., Fox, T., Caldwell, J.W., Kollman, P.A.: A second generation force field for the simulation of proteins and nucleic acids. J. Am. Chem. Soc. 117, 5179–5197 (1995)
44. Cutello, V., Nicosia, G., Pavone, M., Timmis, J.: An Immune Algorithm for Protein Structure Prediction on Lattice Models. IEEE Transactions on Evolutionary Computation 11 (2007)
45. Takahashi, O., Kita, H., Kobayashi, S.: Protein Folding by A Hierarchical Genetic Algorithm. AROB (1999)
46. Unger, R., Moult, J.: Genetic Algorithms for Protein Folding Simulations. J. of Mol. Bio. 231, 75–81 (1993)

47. Unger, R., Moult, J.: Genetic Algorithm for 3D Protein Folding Simulations. In: Conference on GAs, pp. 581–588 (1993)
48. König, R., Dandekar, T.: Refined Genetic Algorithm Simulation to Model Proteins. Journal of Molecular Modeling 5 (1999)
49. Lee, J., Scheraga, H.A., Rackovsky, S.: New Optimization Method for Conformational energy Calculations on Polypeptides: Conformational Space Annealing. J. of Comp. Chemistry 18, 1222–1232 (1997)
50. Haupt, R.L., Haupt, S.E.: Practical Genetic Algorithms (2004)
51. Digalakis, J.G., Margaritis, K.G.: An experimental Study of Benchmarking Functions for Genetic Algorithms Intern. J. Computer Math. 79, 403–416 (2002)
52. Flores, S.D., Smith, J.: Study of Fitness Landscapes for the HP model of Protein Structure Prediction. In: IEEE CEC (2003)
53. Mousseau, N., Barkema, G.T.: Exploring High-Dimensional Energy Landscape. Computing in Science & Engineering 1, 74–80, 82 (1999)
54. Hansmann, U.H.E.: Protein Folding in Silico: An Overview. In: IEEE CS and the AIP (2003)
55. Skolnick, J., Kolinski, A.: Computational Studies of Protein Folding. IEEE COMPUTING IN SCIENCE & ENGINEERING 3, 40–50 (2001)
56. Cui, Y., Wong, W.H., Bornberg-Bauer, E., Chan, H.S.: Recombinatoric exploration of novel folded structures: A heteropolymer-based model of protein evolutionary landscapes. PNAS 99, 809–814 (2002)
57. Schreiner, K.: Distributed Project Tackle Protein Mystery. Computing in Science & Engineering. IEEE 3, 13–16 (2001)
58. Lesh, N., Mitzenmacher, M., Whitesides, S.: A Complete and Effective Move Set for Simplified Protein Folding. In: RECOMB, Berlin, Germany (2003)
59. Hart, W.E., Istrail, S.: HP Benchmarks vol. 2005 (2005)
60. Rosetta, Y.: 2.1.0., Copyright © 2007-2008 The Rosetta Commons (last access, 2008), http://www.rosettacommons.org/tiki/tiki-index.php?page=Change+Log
61. Bonneau, R., Tsai, J., Ruczinski, I., Chivian, D., Rohl, C., Strauss, C.E.M., Baker, D.: Rosetta in CASP4: Progress in Ab Initio Protein Structure Prediction. PROTEINS: Struct. Func. and Genetics 5, 116–119 (2001)
62. Bradley, P., et al.: Rosetta Predictions in CASP5: Success, Failure, and Prospects for Complete Automation. PROTEINS: Structure, Function, and Genetics 53, 457–468 (2003)
63. Simons, K.T., Bonneau, R., Ruczinski, I., Baker, D.: Ab Initio Protein Structure Prediction of CASP III Target Using ROSETTA. PROTEINS: Structure, Function, and Genetics 3, 171–176 (1999)
64. Hoque, M.T., Chetty, M., Dooley, L.S.: Generalized Schemata Theorem Incorporating Twin Removal for Protein Structure Prediction. In: PRIB, Singapore (2007)
65. Koumousis, V.K., Katsaras, C.P.: A Saw-Tooth Genetic Algorithm Combining the Effects of Variable Population Size and Reinitialization to Enhance Performance. TEVC 10, 19–28 (2006)

Evaluation of the Stability of Folding Nucleus upon Mutation

Mathieu Lonquety[1,2], Zoé Lacroix[1,3], and Jacques Chomilier[2]

[1] Scientific Data Management Laboratory, Arizona State University
Tempe AZ 85281-5706, USA
[2] IMPMC, Université Pierre et Marie Curie, UMR 7590 CNRS
140 rue de Lourmel, 75015 Paris, France
[3] Pharmaceutical Genomics Division, Translational Genomics Research Institute
13400 E Shea Blvd, Scottsdale AZ 85259, USA

Abstract. The development of a method that accurately predicts protein folding nucleus is critical at least on two points. On one hand, they can participate to misfolded proteins and therefore they are related to several amyloid diseases. On the other hand, as they constitute structural anchors, their prediction from the sequence can be valuable to improve database screening algorithms. The concept of Most Interacting Residues (MIR) aims at predicting the amino acids more likely to initiate protein folding. An alternative approach describes a protein 3D structure as a series of Tightened End Fragments (TEF). Their spatially close ends have been shown to be mainly located in the folding nucleus. While the current sequence-driven approach seems to capture all MIR, the structure-driven method partially fails to predict known folding. We present a stability-based analysis of protein folding to increase the recall and precision of these two methods.

Results: Prediction of the folding nucleus by MIR algorithm is in agreement with mutation stability prediction.

Availability: The database is available at: http://bioinformatics.eas.asu.edu/Stability/index.php. The MIR calculation program is available at: http://bioserv.rpbs.univ-paris-diderot.fr/cgi-bin/MIR and the TEF program at: http://bioserv.rpbs.univ-paris-diderot.fr/TEF.

Contact: jacques.chomilier@impmc.jussieu.fr

Keywords: Protein folding, folding nucleus, structure stability, point mutations.

1 Introduction

Structural bioinformatics has been particularly productive for the past decade partially thanks to the contribution of physics disciplines. One of its main focuses is the study of protein folding and, in particular, the prediction of folding nuclei. Modeling and predicting protein folding mechanisms is critical because

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 54–65, 2008.

a misfolded protein may result in the formation of aggregates that may play a role in most misfolding diseases such as amyloid ones [1,2,3,4].

The folding nucleus model [5,6,7,8,9] is based on the assumption that protein folding begins with just a few amino acids that strongly interact with each other. These strong interactions initiate the folding that is completed by a successive folding of the remaining parts of the structure to constitute a compact globule. Within this model, a precise and accurate prediction of the main amino acids responsible for initiating the folding provides enough constraints to simulate the whole folding mechanism. For that purpose, Papandreou *et al.* developed an algorithm devoted to the search of the Most Interacting Residues (MIR) [10]. The current algorithm has a good recall however its precision needs improvement as several studies consider that the minimal number of amino acids needed to initiate a folding process is significantly less than the 15 % found in average with the MIR prediction algorithm [8,11,12,13].

Proteins can be described as a succession of Tightened End Fragments or TEF [14,15,16,17] which spatially close ends (lower than 10 Å) are deeply buried in the cores of globular domains. Their ending positions could represent the folding nucleus while TEF would correspond to the final fold for these portions. Indeed, we have previously demonstrated that the TEF ends correspond statistically to hydrophobic residues highly conserved in multiple alignments of proteins of common function [17]. These particular positions have been called topohydrophobic, and they are clearly related to amino acids belonging to the folding nucleus [18]. They are derived from multiple alignments of distantly related sequences, typically less than 30 % identity. It constitutes a limitation of the prediction process since most of the available algorithms for multiple alignments of highly divergent sequences produce controversial results [19]. We have shown that MIR and tophohydrophobic positions match in two thirds of the cases which confirms a reasonable recall of the MIR prediction algorithm. In other words, one has a mean to predict, from the single information of the sequence, positions (MIR) including the folding nucleus.

In this paper we present a stability-based analysis that was conducted to better characterize MIRs. The expected results were to improve the precision of the MIR method by refining the algorithm with constraints related to the prediction of the stability changes induced by point mutations. We assume that the folding nucleus is the deep core of the structure and thus should be very sensitive to point mutations. For example, if a keystone substitutes another one with a different shape, the vaulting will collapse almost every time.

2 Material and Methods

2.1 MIR Prediction Algorithm

A Monte Carlo algorithm is used to simulate the early steps of protein folding on a (2,1,0) lattice. An amino acid is randomly selected and displaced to a new available position on the lattice. The energy of both initial and final conformations is computed from the Miyazawa and Jernigan potential of mean force [20]

and the Metropolis criterion is then applied [21,10]. The starting point is the protein structure in a random coil conformation and the simulation is typically conducted on 10^6 Monte Carlo steps.

This simulation is repeated 100 times with different initial conformations. The number of first neighbors is recorded after each series of 10 Monte Carlo steps, and at the end of the process, an average Number of Contact Neighbors (NCN) is calculated for each amino acid of the sequence. Actually, amino acids surrounded by many others play a role in the compactness of the protein and thus are called Most Interacting Residues (MIR). In contrast, the ones with few neighbors are called Less Interacting Residues (LIR).

2.2 TEF Assignment

Along the backbone of a protein, some pairs of amino acids can be very close in several places, with a typical distance between their alpha carbons below 10 Å. The histogram of the sequence separation between these "contact" amino acids is not smooth, and presents a maximum around 25 amino acids [15]. These sequence fragments were initially called closed loops [14].

Later on, it has been shown that the ends of these closed loops are mainly occupied by hydrophobic amino acids. A thorough analysis demonstrated that these hydrophobic amino acids were highly conserved among structures of the same family, although containing distantly related sequences: these positions were called topohydrophobic [22].

The concept of TEF emerged from the junction between closed loops and topohydrophobic positions mainly located at their ends.

2.3 Free Energy Calculation

Gibbs free energy change due to mutation is a good approximation to characterize the stability of a given structure. It consists of a succession of energetic terms that attempt to capture all the properties and forces that drive the conformation of a protein. In our study we focus on the difference of these energies for the wild type structure ΔG_{wild} and for the mutant structure ΔG_{mutant}. Considering that in the literature various stability prediction methods use different nomenclature, $\Delta\Delta G$ is defined as follows:

$$\Delta\Delta G = \Delta G_{\text{mutant}} - \Delta G_{\text{wild}} \ . \tag{1}$$

The unit is kcal/mol. $\Delta\Delta G$ describes whether it costs more in energy to have the mutated amino acid or the wild type one. For example, if $\Delta\Delta G < 0$ then it costs more in energy to have the wild type structure than the mutant one thus the mutation is more favorable to the structure stability. Conversely, if $\Delta\Delta G > 0$, the mutant structure ΔG is higher than the wild type one thus the mutation is less favorable to the structure stability.

2.4 Stability Analysis

Many methods have been implemented to predict stability changes induced by point mutations. MUpro [23] and I-Mutant (sequence version) [24] both predict stability changes on a protein sequence whereas DFIRE [25], I-Mutant (sequence + structure version) [24] and PoPMuSiC [26] use protein sequence and structure to predict these changes. Other methods exist but have been rejected due to some restrictions: CUPSAT[1] [27] was not available in a standalone version and the current version of FoldX[2] [28] only computes mutations to Alanine. To avoid biases from one or the other method, we present a comprehensive analysis with five existing tools (the two versions of I-Mutant are considered as two different tools).

We use the Protherm database [29] that collects thermodynamic data published in the scientific literature and thus includes measured values of $\Delta\Delta G$ to compare our prediction to experimental data. It is available at the following URL: http://gibk26.bse.kyutech.ac.jp/jouhou/Protherm/protherm.html.

2.5 Data Set

Our analysis was conducted on a dataset published by the Protein Folding Fragments European consortium that can be found at the URL: http://bioserv.rpbs.univ-paris-diderot.fr/PFF/. MIR predictions and TEF calculations were already performed on the selected 116 protein sequences.

The experimental dataset consisted of 116 protein sequences for a total of 15,183 amino acids. Each sequence was processed with each of the five stability prediction tools, for each amino acid, and for each of the 19 possible mutations. We computed 1,442,385 different $\Delta\Delta G$ values. In order to manage and publish our produced data and results in a more efficient way than output flat files, a database was created and is available to the community at http://bioinformatics.eas.asu.edu/Stability/ where more information about the data can be found.

3 Results

3.1 MIR and TEF

The MIR concept aims at characterizing the main amino acids involved in the early steps of the protein folding process. The TEF method splits a structure into fragments with spatially close ends that interact with each other. A previous study [10] has demonstrated that TEF ends (within a range of ± 5 positions) correspond to MIR in 57 % of the cases. As we are looking at coherent methods to determine the folding nucleus, we observe that MIR over predict the TEF ends, therefore we hypothesize that restricting the MIR to the ones in agreement with TEF ends would capture the expected amino acids responsible for protein folding.

[1] CUPSAT is available at http://cupsat.tu-bs.de/.
[2] FoldX is available at http://foldx.crg.es//.

Then, separation between "good" MIR and "bad" MIR emerges, and we define them as TEF related and TEF independent MIR. The TEF independent MIR are expected to be the noise in the MIR prediction algorithm. The TEF related ones are those in a ± 3 amino acids window around a TEF end. As the experimental validation of folding nucleus is rather difficult, one way to validate MIR prediction (i.e., TEF related MIR are the nucleus residues) is the comparison with other structural data. Indeed, there is nowadays no experimental technique able to determine which residues constitute the folding nucleus.

The Φ value experimental determination [30] attests whether one amino acid is in the folding nucleus or not, but all the mutants need to be constructed to validate the hypothesis. Nevertheless in some cases, such as CI2 for instance, a low Φ value can be obtained for a residue attributed to the folding nucleus by convergent experiments [31]. We propose here to add constraints derived from energy stability evaluation in order to increase the agreement between prediction and experiment; these constraints are restrictred to thermodynamics experiments as they are not supposed to be suspicious.

3.2 Stability Changes upon Point Mutations

Because they are structurally compulsory for the complete folding, we assume that folding nucleus positions are very sensitive to mutation, in the sense that a mutation would destabilize the protein. We thus decide to verify this assumption by computing stability changes upon point mutations for all the sequences which already have been processed for the MIR and TEF predictions (See Material and Methods section). There exist numerous software devoted to this task among which we focused on: DFIRE, two versions of I-Mutant, MUpro, and PoPMuSiC.

We first start by calculating $\Delta\Delta G$ resulting from mutations at each position for the five tools. We retrieve all experimental values for proteins either present in our database and in Protherm [29]. 1409 different mutations with their experimental $\Delta\Delta G$ were gathered. A correlation then appear between experimental $\Delta\Delta G$ and predicted $\Delta\Delta G$. The two versions of I-Mutant obtain the best score (represented in Fig. 1) with 0.96 correlation coefficient, just followed by MUpro with 0.86. The remaining tools PoPMuSiC and DFIRE show an average correlation with 0.53 and 0.48 respectively. The goal of these correlations is to verify that the tools used in this work are accurate and truthfully. The excellent correlation for I-Mutant and MUpro can be explained by the fact that both software used data extracted from the Protherm database as a training set for their algorithm. No other experimental data was available for this study.

Three tools can be considered as efficient and two others have to be carefully apprehended. Nevertheless, for a better overview of the stability changes concept, all five tools are kept in this study and a comparison between the two types of MIR and their relative stability changes upon mutation can be performed. We then verify if stability upon mutation would allow to discriminate among the two types of MIR, according to their location relative to TEF ends. Stability prediction is characterized by $\Delta\Delta G$ on each amino acid and for each possible mutation. We compute a stability score to compare with MIR prediction as

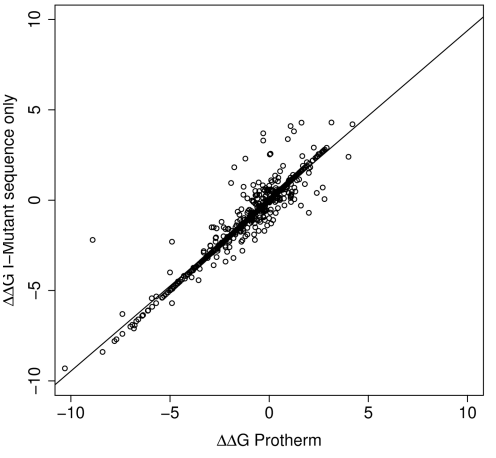


Fig. 1. Graph of $\Delta\Delta G$ (kcal/mol) predicted by I-Mutant (sequence only version) as a function of the experimental one, taken from the Protherm database. The line represents the correlation between both experiment and prediction.

follows. To synthesize the mean stability change tendency for a given amino acid, the 19 $\Delta\Delta G$ have been summed in one value. Actually, to normalize this result, instead of summing the $\Delta\Delta G$, a score has been given to each $\Delta\Delta G$. If $\Delta\Delta G < 0$, the mutation is considered as stabilizing and it is granted a value of $+1$. Conversely, if $\Delta\Delta G > 0$, the mutation is considered as destabilizing and the value is -1. This procedure produces a score in the range of $[-19,+19]$ which reflects the global stability change for an amino acid upon its mutation. The lower the score, the more sensitive to mutation, i.e., the native residue is the most stable. This stability score is computed for each amino acid of all the sequences in the data set and for the five different tools. Moreover, a consensus tool has been created which corresponds to the mean of the five programs. Graphs are plotted, upon request on the server, to get an overview of the stability score over a whole sequence. One example is given in Fig. 2 where the stability scores along a whole sequence have been represented for the five tools in the case of the engrailed homeodomain (PDB code: 1enh). Stability scores curves have also been smoothened for an easier interpretation.

We observe that along the homeodomain sequence, stability changes score ranges from -19 to $+15$. One can notice that most values are under 0 which means that there are more positions destabilized by mutations than stabilized ones. This observation is in agreement with the principles of Evolution which tend to favor stable protein structures.

It is thus possible to detect the most sensitive positions to a mutation. Indeed, the minima of stability scores are positions for which mutations induce the most destabilizing changes, regarding free energy, along the structure. We can locate these positions and compare them with the MIR.

To bear evidence of an eventual co-localisation of MIR and stable positions, we compute the distance in sequence that separates each MIR (TEF related and TEF independent) from the nearest minimum of stability score. These differences of positions have been computed for all the sequences of the database and for each tool. Figure 3 shows an example of these deltas of positions for the consensus tool. It appears that there is no distinction between the two classes of MIR as both have their respective peak centered on the same position.

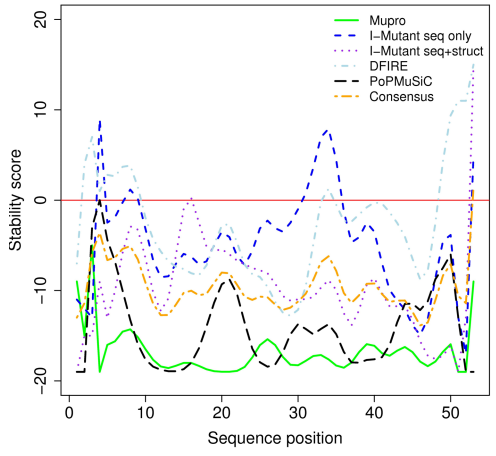


Fig. 2. Representation of the stability scores for each amino acid of the 1enh sequence. The five lines represent each one a different tool. The consensus graph is also represented.

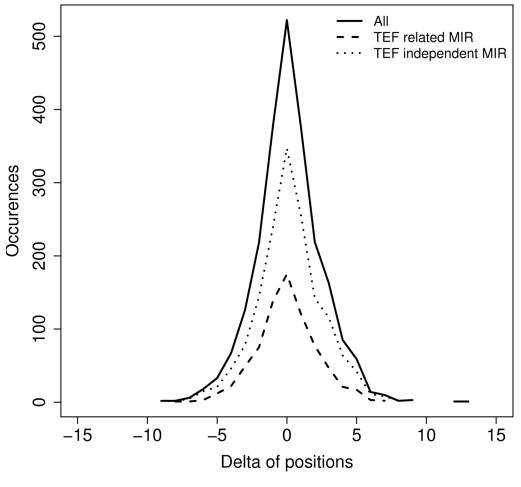


Fig. 3. The origin of the abscissa corresponds to the position of each MIR (TEF related and independent) and one calculates the distance to the closest minimum of stability scores on the whole dataset and for the consensus tool.

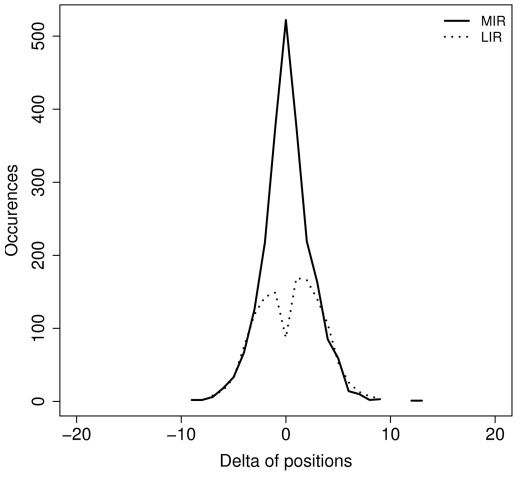


Fig. 4. Sequence separation (Delta) between MIR, LIR, and minima of stability scores on the whole data set and for the consensus tool. The origin is taken as for Fig. 3.

However, MIR have been shown to statistically match the topohydrophobic positions, corresponding in several experiments to the folding nucleus [18]. The conservation of the folding nucleus among species is still under a strong debate. If we assume, in agreement with Shakhnovich [31], that the folding nucleus is the subject of an additional evolution pressure, considering simulations on distantly related sequences of the same fold, instead of single one, may help in a better definition of the folding nucleus.

Two hypotheses emerge: the MIR algorithm is not accurate enough or the TEF assignment has to be improved. A direct comparison of each of these methods with minimua of protein stability scores has been processed.

The determination of MIR accuracy relies on their good match with minima of stability scores but also on the comparison between their antagonists i.e., LIR for Less Interacting Residues and stability scores under the same protocol. These amino acids are the ones with the smallest number of contact neighbors and are thus assumed to be mainly located at the interface of the protein and the solvent. Results are shown in Fig. 4. As already seen in Fig. 3, MIR are statistically located at minima of stability scores with a peak for a delta of 0. If one takes a window centered on 0, on the $[-1, +1]$ range, 55 % of the MIR correspond to a minimum of stability score. For the LIR, we observe that there is a clear minimum on the 0 position and two peaks centered on the -2 and +2 positions. The conclusion is that the MIR concept is in good agreement with the concept of sensitivity to the structure stability. This prediction method succeeds in correctly locating the most stable residues, that can be either located at the ends of TEF or elsewhere, because both classes of MIR match the location of the lowest scores.

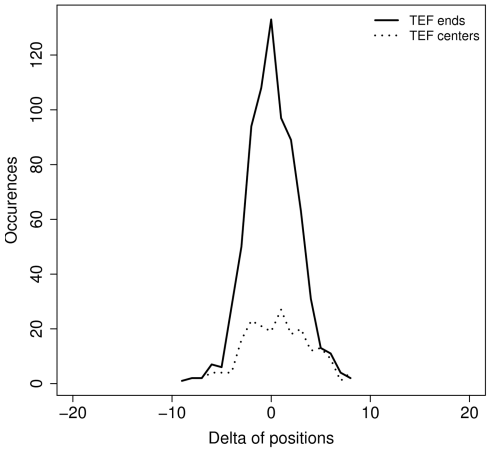


Fig. 5. Sequence separation (Delta) between TEF ends, TEF centers, and minima of stability scores on the whole data set and for the consensus tool.

We then compare specific TEF positions to the positions with the lowest stability scores. We consider on one hand TEF ends, as they are assumed to be in the folding nucleus, and TEF centers on the other hand. Figure 5 represents the distance in terms of amino acids (delta) between the TEF ends/TEF centers and minima of stability score on the other side. The results are compatible with the ones presented for Fig. 4 with MIR and LIR. TEF ends match the positions where stability is the highest (lowest scores), while TEF centers do not. Therefore, one can conclude that MIR predictions capture some physics of the folding process, by finding residues forming the core, evaluated here on the basis of the most stable positions toward mutation, independently of their location relative to the TEF.

The relative efficiency of this method has been confirmed by the calculation of solvent accessible surface for all amino acids and for each sequence of the database. The mean value is of $53\,\text{Å}^2$ for one amino acid among all the sequences of the dataset. If we now consider amino acids which are characterized as MIR, this mean drops to $33\,\text{Å}^2$. For the LIR, we obtain a rise to $64\,\text{Å}^2$. This observation also gives another evidence of the efficiency of the MIR method as low solvent accessible surface induces that the considered amino acid is buried inside the globular domain.

For the results observed for the TEF the conclusion is less evident. TEF ends are centered on the positions of the highest stability, but TEF centers graph is more ambiguous as there is kind of a plateau in the range $[-3, +3]$. It thus means that TEF ends are quite in agreement with stability scores minima but TEF centers do not show any tendency to be reluctant to stability scores minima.

4 Conclusion

Protein folding is nowadays one of the biggest challenges in structural bioinformatics. The MIR method is devoted to the prediction of the residues forming the

folding nucleus of proteins. Some refinement has been proposed to improve the accuracy of the current algorithm such as the use of additional input based on topology and stability. The structural analysis of proteins in terms of TEF was a relevant choice as it captures ends of fragments buried in the core of the protein. MIR are constituted of two families, the ones present at the ends of TEF and the other ones found elsewhere. It was hypothesized that folding nucleus would preferentially be located at TEF ends. We checked the relevance of this separation by comparing the presence of MIR with positions known to be stable upon point mutation. We actually evidenced that both classes of MIR are highly stable positions with respect to mutations. This result may be interpreted in the following way: if we admit the assumption that most stable positions toward mutation are indicative of the inclusion in the folding nucleus, then MIR is a rather satisfactory method to predict this nucleus. In addition, we assume that split of the protein structures into TEF should be improved, and in particular, one might think of secondary contacts, i.e. two residues located in the middle of a TEF, and close from one each other.

Although we probably overestimate the number of amino acids involved in the folding nucleus, our approach might be a help for selecting positions susceptible of experimental muations in order to perform Φ_F determination. A long term application of this prediction nucleus algorithm is its inclusion in database screening tools, in order to give a stronger weight once a residue has been postulated as belonging to the nucleus. One might guess that this would help in retrieving more distantly related sequences than present methods.

Acknowledgement. RPBS resources have been used in order to produce all these results. The authors would like to thank Dr. Nikolaos Papandreou for his contribution to the MIR calculation program. Many thanks for the authors of the different software devoted to stability changes and especially to Dr. Jean-Marc Kwasigroch for his help on using the PoPMuSiC software. Dr. Pierre Tufféry is acknowledged for his help in implementing the MIR and TEF programs on the RPBS server.

This research was partially supported by the National Science Foundation[3] (grants IIS 0431174, IIS 0551444, and IIS 0612273) and by an invitation of the Université Pierre et Marie Curie. J.C. and M.L. benefited of a EU grant QLG2-2002-01298.

References

1. Brockwell, D.J., Smith, D.A., Radford, S.E.: Protein folding mechanisms: new methods and emerging ideas. Curr. Opin. Struct. Biol. 10(1), 16–25 (2000)
2. Steward, R.E., MacArthur, M.W., Laskowski, R.A., Thornton, J.M.: Molecular basis of inherited diseases: a structural perspective. Trends Genet. 19(9), 505–513 (2003)

[3] Any opinion, finding, and conclusion or recommendation expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

3. Mogensen, J.E., Ipsen, H., Holm, J., Otzen, D.E.: Elimination of a misfolded folding intermediate by a single point mutation. Biochemistry 43(12), 3357–3367 (2004)
4. Cerdà-Costa, N., Esteras-Chopo, A., Avilés, F.X., Serrano, L., Villegas, V.: Early kinetics of amyloid fibril formation reveals conformational reorganisation of initial aggregates. J. Mol. Biol. 366(4), 1351–1363 (2007)
5. Wetlaufer, D.B.: Nucleation, rapid folding, and globular intrachain regions in proteins. Proc. Natl. Acad. Sci. USA 70(3), 697–701 (1973)
6. Abkevich, V.I., Gutin, A.M., Shakhnovich, E.I.: Specific nucleus as the transition state for protein folding: evidence from the lattice model. Biochemistry 33(33), 10026–10036 (1994)
7. Fersht, A.R.: Optimization of rates of protein folding: the nucleation-condensation mechanism and its implications. Proc. Natl. Acad. Sci. USA 92(24), 10869–10873 (1995)
8. Shakhnovich, E., Abkevich, V., Ptitsyn, O.: Conserved residues and the mechanism of protein folding. Nature 379(6560), 96–98 (1996)
9. Fersht, A.R.: Nucleation mechanisms in protein folding. Curr. Opin. Struct. Biol. 7(1), 3–9 (1997)
10. Papandreou, N., Berezovsky, I.N., Lopes, A., Eliopoulos, E., Chomilier, J.: Universal positions in globular proteins. Eur. J. Biochem. 271(23-24), 4762–4768 (2004)
11. Sacile, R., Ruggiero, C.: Hunting for "key residues" in the modeling of globular protein folding: an artificial neural network-based approach. IEEE Trans Nanobioscience 1(2), 85–91 (2002)
12. Religa, T.L., Markson, J.S., Mayor, U., Freund, S.M.V., Fersht, A.R.: Solution structure of a protein denatured state and folding intermediate. Nature 437(7061), 1053–1056 (2005)
13. Alexander, P.A., He, Y., Chen, Y., Orban, J., Bryan, P.N.: The design and characterization of two proteins with 88% sequence identity but different structure and function. Proc. Natl. Acad. Sci. 104(29), 11961–11963 (1968)
14. Ittah, V., Haas, E.: Nonlocal interactions stabilize long range loops in the initial folding intermediates of reduced bovine pancreatic trypsin inhibitor. Biochemistry 34(13), 4493–4506 (1995)
15. Berezovsky, I.N., Grosberg, A.Y., Trifonov, E.N.: Closed loops of nearly standard size: common basic element of protein structure. FEBS 466(2-3), 283–286 (2000)
16. Berezovsky, I.N., Kirzhner, V.M., Kirzhner, A., Trifonov, E.N.: Protein folding: looping from hydrophobic nuclei. Proteins 45(4), 346–350 (2001)
17. Lamarine, M., Mornon, J.P., Berezovsky, N., Chomilier, J.: Distribution of tightened end fragments of globular proteins statistically matches that of topohydrophobic positions: towards an efficient punctuation of protein folding? Cell Mol. Life Sci. 58(3), 492–498 (2001)
18. Poupon, A., Mornon, J.P.: Predicting the protein folding nucleus from sequences [correction of a sequence]. FEBS Lett. 452(3), 283–289 (1999)
19. Baussand, J., Deremble, C., Carbone, A.: Periodic distributions of hydrophobic amino acids allows the definition of fundamental building blocks to align distantly related proteins. Proteins 67(3), 695–708 (2007)
20. 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. J. Mol. Biol. 256(3), 623–644 (1996)
21. Chomilier, J., Lamarine, M., Mornon, J.P., Torres, J.H., Eliopoulos, E., Papandreou, N.: Analysis of fragments induced by simulated lattice protein folding. C. R. Biol. 327(5), 431–443 (2004)

22. Poupon, A., Mornon, J.P.: Populations of hydrophobic amino acids within protein globular domains: identification of conserved "topohydrophobic" positions. Proteins 33(3), 329–342 (1998)
23. Cheng, J., Randall, A., Baldi, P.: Prediction of protein stability changes for single-site mutations using support vector machines. Proteins 62(4), 1125–1132 (2006)
24. Capriotti, E., Fariselli, P., Casadio, R.: I-Mutant2.0: predicting stability changes upon mutation from the protein sequence or structure. Nucleic Acids Res. 33(Web Server issue), W306–W310 (2005)
25. Zhou, H., Zhou, Y.: Distance-scaled, finite ideal-gas reference state improves structure-derived potentials of mean force for structure selection and stability prediction. Protein Sci. 11(11), 2714–2726 (2002)
26. Gilis, D., Rooman, M.: PoPMuSiC, an algorithm for predicting protein mutant stability changes: application to prion proteins. Protein Eng. 13(12), 849–856 (2000)
27. Parthiban, V., Gromiha, M.M., Schomburg, D.: CUPSAT: prediction of protein stability upon point mutations. Nucleic Acids Res. 34(Web Server issue), W239–W242 (2006)
28. Schymkowitz, J., Borg, J., Stricher, F., Nys, R., Rousseau, F., Serrano, L.: The FoldX web server: an online force field. Nucleic Acids Res. 33(Web Server issue), W382–W388 (2005)
29. Kumar, M.D.S., Bava, K.A., Gromiha, M.M., Prabakaran, P., Kitajima, K., Uedaira, H., Sarai, A.: ProTherm and ProNIT: thermodynamic databases for proteins and protein-nucleic acid interactions. Nucleic Acids Res. 34(Database issue), D204–D206 (2006)
30. Fersht, A.R., Daggett, V.: Protein folding and unfolding at atomic resolution. Cell 108(4), 573–582 (2002)
31. Shakhnovich, E.: Protein folding thermodynamics and dynamics: where physics, chemistry, and biology meet. Chem. Rev. 106(5), 1559–1588 (2006)

Prediction of Protein Beta-Sheets: Dynamic Programming versus Grammatical Approach

Yuki Kato[1], Tatsuya Akutsu[1], and Hiroyuki Seki[2]

[1] Bioinformatics Center, Institute for Chemical Research, Kyoto University, Gokasho, Uji, Kyoto 611-0011, Japan
{ykato,takutsu}@kuicr.kyoto-u.ac.jp
[2] Graduate School of Information Science, Nara Institute of Science and Technology, 8916-5 Takayama, Ikoma, Nara 630-0192, Japan
seki@is.naist.jp

Abstract. Protein secondary structure prediction is one major task in bioinformatics and various methods in pattern recognition and machine learning have been applied. In particular, it is a challenge to predict β-sheet structures since they range over several discontinuous regions in an amino acid sequence. In this paper, we propose a dynamic programming algorithm for some kind of antiparallel β-sheet, where the proposed approach can be extended for more general classes of β-sheets. Experimental results for real data show that our prediction algorithm has good performance in accuracy. We also show a relation between the proposed algorithm and a grammar-based method. Furthermore, we prove that prediction of planar β-sheet structures is NP-hard.

Keywords: β-sheet, dynamic programming, formal grammar, computational complexity.

1 Introduction

Protein structure prediction is one of the central problems in bioinformatics and computational biology, and various approaches have so far been proposed. Secondary structure prediction is one of the major approaches. It asks which type of secondary structure (α-helix, β-strand, or others) each residue belongs to. Since it is a kind of classification problem, various machine learning and pattern recognition techniques have been applied, including hidden Markov models [3,16], logic programming [20], neural networks [22], stochastic tree grammars [1] and support vector machines [12]. Although the overall prediction accuracy of existing methods is around 75% [18], it is recognized that β-strand regions are more difficult to predict than α-helix regions. This discrepancy may come from the fact that β-sheet structures typically range over several discontinuous regions, whereas α-helices are continuous and thus depend more on local sequence patterns.

Protein threading is another major approach for protein structure prediction. In this approach, alignment between an input amino acid sequence and a template protein structure is computed. It is known that protein threading is

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 66–77, 2008.

NP-hard if pairwise interactions of residues must be taken into account [2,17]. However, several optimal algorithms have been developed for protein threading with pairwise residue-residue interactions under an assumption that insertions or deletions do not occur in core regions (i.e., α-helices and β-strands) [26]. Although it is usually overlooked in literature, there is a similarity between protein secondary structure prediction and protein threading. In protein threading (with pairwise interactions), configuration of core regions is given in advance (from a template 3D structure) and each core (α-helix or β-strand) region is searched for in an input protein sequence. In secondary structure prediction, configuration of core regions is not given in advance and each residue is assigned to one of the three classes of secondary structures.

Although we have discussed about protein structure prediction, RNA secondary structure prediction is another important problem in bioinformatics and computational biology. One of the common approaches of RNA secondary structure prediction is use of (stochastic) grammars, which include stochastic context-free grammar [10,23], stochastic multiple context-free grammar [15], parallel communicating grammar [7], crossed-interaction grammar [21] and tree adjoining grammar [25]. These grammars may also be useful to model other pattern recognition problems.

Recently, Chiang et al. [8] proposed some grammar-based methods for protein secondary structure prediction. In particular, they proposed use of *range concatenation grammar* (RCG) [5] for β-sheet modeling. They suggested that linearly ordered β-sheets can be modeled by using a simple RCG and can be predicted in $O(n^5)$ time, where n is the number of residues in a given protein sequence. They also suggested that β-barrels and more complex β-sheet structures can be modeled by using RCG, while the time complexity increases to $O(n^7) \sim O(n^{12})$ depending on the complexity of β-sheet structures. However, they did not show how to incorporate residue-residue interaction preferences into the RCG-based methods. Furthermore, they posed the following question for proving NP-hardness of β-sheet prediction: "it remains to be seen whether such dependencies might be needed, for example, in calculating conformation counts for β-sheets."

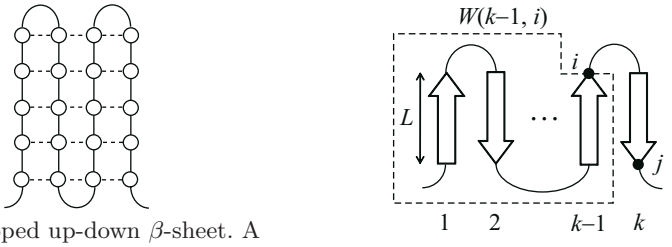
In this paper, we propose a simple and flexible dynamic programming algorithm for prediction of antiparallel up-down β-sheets. This algorithm is based on RCG approach [8], where no experimental results on structure prediction were provided. It is noteworthy that our method explicitly takes pairwise interaction preferences into account and thus can be applied to real protein sequences. Hubbard [13] also used interstrand residue pairing preferences to predict β-strand contact maps, but did not show an original prediction algorithm specific for β-sheet prediction. Our prediction algorithm achieved good performance of overall per-residue accuracy $Q_3 \approx 80\%$ for nonhomologous protein sequences with up-down topology, where there are only two secondary structural states. Although types of β-sheet structures that can be handled by our method are restricted, the technique is extensible to more complex β-sheet structures including β-barrel. We also provide insight into an existing grammar-based method. Furthermore,

we show that prediction of planar β-sheet structures is NP-hard. This result gives an answer to the question posed by Chiang et al. [8].

2 Methods

2.1 Ungapped Antiparallel β-Sheet

β-sheets are formed by pairwise interaction of several (consecutive) amino acids, called β-strands, in parallel and/or antiparallel way. Antiparallel β-structure is a fundamental topology of β-sheet, and many proteins include it in their domain. Although there are a large number of combinations of β-strands, it is known that the number of topologies of the class of antiparallel β-sheets is relatively few [6]. In this section, we are concerned with the simplest topology among them, called *up-down β-sheet*, where all strands have antiparallel topology via hydrogen bonding and they are connected by hairpin. In addition, suppose that every amino acid of β-strands is involved in hydrogen bonding, which we call *ungapped β-sheet*. Fig. 1 (a) illustrates an ungapped up-down β-sheet. This assumption enables us to design more efficient prediction algorithm in terms of computational complexity.



(a) An ungapped up-down β-sheet. A white circle represents an amino acid and a dashed line indicates a hydrogen bond.

(b) A schematic diagram of the dynamic programming algorithm

Fig. 1. Illustration of an ungapped up-down β-sheet

Let $a = a_1 a_2 \cdots a_n$ denote an amino acid sequence to be analyzed. We consider an ungapped up-down β-sheet that have N strands of the same length L where $N \leq \lfloor \frac{n}{L} \rfloor$. The reason why we can assume L is fixed is that we are concerned with only ungapped β-sheets. Because of this assumption, a β-sheet can be represented by an N-tuple of the start positions of β-strands $(p_1, p_2, \ldots, p_N)$ in the amino acid sequence a. Note that $p_i + L \leq p_{i+1}$ must be satisfied to prevent adjacent strands from overlapping each other. Let $s : (a_i, a_j) \to \mathbb{R}$ be a score (energy) function between two amino acid residues. Then, the ungapped up-down β-sheet prediction problem can be defined as follows:

Definition 1. ***(Ungapped up-down β-sheet prediction problem)***
Input: *An amino acid sequence $a = a_1 a_2 \cdots a_n$, the number of strands N, their common length L and a score function s.*

Output: *An ungapped up-down β-sheet $(p_1, p_2, \ldots, p_N)$ that minimizes the following score:*

$$\sum_{i=1}^{N-1} \sum_{j=1}^{L} s(a_{p_i+j-1}, a_{p_{i+1}+L-j}),$$

subject to $p_i + L \leq p_{i+1}$ $(i = 1, 2, \ldots, N)$.

2.2 Dynamic Programming Algorithm

We provide a dynamic programming (DP) algorithm for predicting ungapped up-down β-sheets. In the experiments described later, we predict β-sheet by changing the value of N, though N is fixed in the algorithm described below. Let $W(k, j)$ be the minimum free energy of up-down β-sheet for $a_1 \cdots a_j$, where j is the last position of the kth β-strand (see Fig. 1 (b)). $W(k, j)$ can be calculated by the following simple recursion formula:

$$W(k, j) = \min_i \{W(k-1, i) + S(i, j, L)\},$$

where

$$S(i, j, L) = \sum_{h=1}^{L} s(a_{i-L+h}, a_{j-h+1}).$$

The detailed description of the DP algorithm is presented below.

Initialization:
for $j = L$ **to** n **do** $W(1, j) = 0$.
Recursion:
for $k = 2$ **to** N **do**
for $j = kL$ **to** n **do**

$$W(k, j) = \min_{(k-1)L \leq i \leq j-L-2} \{W(k-1, i) + S(i, j, L)\}.$$

Note that this algorithm takes the length of hairpin into consideration by restricting the range of i in the recursive step.

A simple inspection of the recursive step yields the time complexity of the algorithm. Since the double "for loop" takes $O(n^2)$ time and the minimum operation takes $O(n)$ time, the time complexity is evaluated as $O(n^3)$. Obviously, the algorithm requires $O(n^2)$ space. Note that the optimal β-sheet itself can be constructed by a simple traceback procedure.

Although our DP algorithm can only handle up-down β-sheets, we can easily extend our method to predict more complicated structures, including consecutive parallel β-sheets, β-barrels as well as gapped structures.

In order to extend the algorithm for β-barrels, we compute the following:

$$W(k, j, i_0) = \min_i \{W(k-1, i, i_0) + S(i, j, L)\}$$

for each i_0 under the condition that

$$W(1, j, i_0) = \begin{cases} 0 \ \ (j = i_0), \\ \infty \ \text{(otherwise)}. \end{cases}$$

Then, we compute the minimum of

$$W(N, j, i_0) + \sum_{h=1}^{L} s(a_{i_0-L+h}, a_{j-h+1}).$$

In this case, the time complexity increases from $O(n^3)$ to $O(n^4)$. More complex β-sheet structures may be treated by using the divide-and-conquer approach proposed by Xu et al. [26]. However, the time complexity would increase as the complexity of β-sheet increases as suggested by the NP-hardness result in Section 5.

In order to extend the algorithm for gapped antiparallel β-sheets, it is enough to modify the definition of $S(i, j, L)$ so that it denotes the score of an optimal *alignment* between $a_{i-L+1} \cdots a_i$ and $a_j \cdots a_{j-L+1}$. In this case, the total time complexity increases to $O(n^4)$. Of course, we can extend it for prediction of gapped β-barrels. In that case, the time complexity remains $O(n^4)$. Capability of handling gapped β-sheets is one of the big advantages of our proposed method since gaps in core regions are not allowed in protein threading with residue-residue pairwise interactions [26].

3 Experimental Results

3.1 Data

In our experiments on prediction of up-down β-sheets with β-barrels, we used real protein sequences with known structure available in PDB_SELECT (2007) [11] as the test sets (see Table 1). The criteria for selecting test data are as follows:

(1) The test sequences are contained in the 25% threshold list of PDB_SELECT, where no two proteins have more than 25% sequence identity.
(2) They have at least four β-strands specified in DSSP [14]. Note that we do not count a residue involved in an isolated β-bridge as one strand.
(3) All but at most one pair of adjacent β-strands in the primary sequence are involved in hydrogen bonding. This constraint results from lack of a perfect set of up-down β-sheets in the list.

3.2 Tests

Since the sequences selected above actually have different strand lengths, we set the strand length constant L by rounding the mean of their actual lengths. We used a contact potential table derived from 785 proteins described in [9] as

Table 1. Accuracy of antiparallel β-sheet prediction

(a) Up-down β-sheet prediction

PDBID	N	n	L	Q_3 [%]	Q_E [%]	Q_E^{pred} [%]
2B9K	4	47	7	72.34	77.78	75.00
1AUU	4	55	4	83.64	70.59	75.00
1NY4	4	82	6	84.15	72.00	75.00
1TPN	5	50	4	68.00	61.11	55.00
2E6Z	5	59	4	74.58	61.90	65.00
2DIG	5	68	5	82.35	74.07	80.00
2JN4	6	66	5	87.88	89.29	83.33
2BT9	8	90	8	80.00	88.33	82.81
Average				79.12	74.38	73.89

(b) β-barrel prediction

PDBID	N	n	L	Q_3 [%]	Q_E [%]	Q_E^{pred} [%]
1Q9F	7	148	10	70.95	69.01	70.00
1MM4	8	170	9	64.71	58.11	59.72
1G90	8	176	11	82.39	81.32	84.09
1FW3	12	269	12	63.20	65.73	65.28
1PHO	16	330	11	66.36	67.96	69.89
Average				69.52	68.43	69.80

the score function s. Implementation of the prediction algorithms for up-down β-sheet and β-barrel was carried out in Java (version 1.6.0_03) on a machine with Intel Core2 CPU 6700 2.66GHz, 1.57GHz and 2.99GB RAM. To evaluate prediction accuracy of our algorithms, we measured per-residue accuracy Q_3, Q_E and Q_E^{pred}. Q_3 is the ratio of correctly predicted residues in overall secondary structural elements. Note that there are only two secondary structural states in this case (i.e., strand and other), and observed structures that we referred to are specified in DSSP. Q_E is defined as the ratio of the number of correctly predicted residues of the β-strands to the total number of residues of the strands in the observed structure, which corresponds to sensitivity. Q_E^{pred}, corresponding to specificity, is the ratio of the number of correctly predicted residues of the β-strands to the total number of predicted residues of the strands. Prediction results on up-down β-sheet prediction are shown in Table 1 (a) and Fig. 2, and results on β-barrel prediction are shown in Table 1 (b). Computation time of up-down β-sheet prediction was 0.19 seconds on average, whereas computation time of β-barrel prediction was 480.04 seconds on average. Note that this discrepancy arises from the difference of time complexity (i.e., $O(n^3)$ vs. $O(n^4)$).

Observed beta sheet (E: extended strand, participates in beta ladder):
MKVMIRKTATGHSAYVAKKDLEELIVEMENPALWGGKVTLANGWQLELPAMAADTPLPITVEARKL
..EEEEE....EEEEE....EEEEEEEE........EEEE....EEE...........EEE.....

Predicted beta sheet:
MKVMIRKTATGHSAYVAKKDLEELIVEMENPALWGGKVTLANGWQLELPAMAADTPLPITVEARKL
..EEEEE....EEEEE.......EEEEE.......EEEEE...EEEEE........EEEEE.....

Fig. 2. Comparison of the observed structure with the predicted one for 2JN4. Underlined residues indicate that they agree with correct residues of the β-strands.

3.3 Discussion

Experimental results on up-down β-sheet prediction show that our prediction algorithm has good performance in accuracy for several real protein sequences. One reason for high accuracy is that the contact potentials computed in [9] are good in quality. In fact, we performed the same prediction tests using other contact potentials presented in [4,24,27], where average prediction accuracy is 76.62% in Q_3, 71.71% in Q_E and 71.52% in Q_E^{pred} for [4], 72.52% in Q_3, 66.14%

in Q_E and 65.68% in Q_E^{pred} for [24], and 67.86% in Q_3, 60.01% in Q_E and 59.73% in Q_E^{pred} for [27]. These values are lower than the average accuracy when using the contact potentials in [9]. It should be noted that a few protein structures (1AUU and 1TPN) used to compute the contact potentials in [9] were also used for our experiments. However, most accuracy assessment for these two proteins is lower than the average (see Table 1 (a)), and there seems to be no positive bias that improves the accuracy of the algorithm.

It can be seen that the choice of the number of β-strands N is important to achieve good prediction accuracy. After we performed the test shown in Table 1 where N was actually chosen as the observed number of strands N_{obs}, we developed a simple method of selecting N during computation of the DP table W. More specifically, we calculated the average of $W(k, j)$ for each k ($2 \leq k \leq \lfloor \frac{n}{L} \rfloor$), denoted by $W_{avg}(k)$, and then selected N as the first k such that $W_{avg}(k) < W_{avg}(k+1)$ holds while calculating in an increasing order of k. Although the average prediction accuracy for up-down β-sheets drops to 72.52% in Q_3, 74.41% in Q_E and 65.17% in Q_E^{pred}, the value N determined by this method ranges from $N_{obs}-1$ to $N_{obs}+2$, which shows a relatively good tendency in choice of N.

As Table 1 (b) indicates, prediction accuracy for β-barrels is not so good as compared with the results on up-down β-sheet prediction. This may suggest that achieving good accuracy is difficult if the topology of the β-sheet to be analyzed becomes complex. To achieve higher accuracy than the present accuracy for β-barrels, it would be interesting to incorporate "torsion changes" into our algorithms, which is considered to be important for the stability of a protein.

As compared to another approach for β-sheet prediction, accuracy of a method using *ranked node rewriting grammar* (RNRG) [1] is roughly 74% in Q_E, which is comparable to the performance of our method. Although the test data used in our experiments are different from the data used in the RNRG-based method, we tested more sequences than they did. Furthermore, it should be noted that we never used a training algorithm to estimate score parameters, whereas the RNRG approach performed training of probability parameters using an inside-outside algorithm, which is prohibitively time-consuming.

4 Remarks on Grammatical Modeling

4.1 Definitions

Range concatenation grammar [5] is defined as a deductive system on sequences. A (positive) range concatenation grammar (RCG) is a 5-tuple $G = (N, T, V, P, S)$, where N, T, V and P are finite sets of predicate names, terminals, variables, rules, respectively, and $S \in N$ is the start predicate. For each predicate name $A \in N$, a nonnegative integer $\dim(A)$ is specified. Each rule in P has the shape $\psi_0 \rightarrow \psi_1 \cdots \psi_k$. This rule means that ψ_0 holds when all of $\psi_1, \ldots, \psi_k$ hold. Each ψ_i ($0 \leq i \leq k$) in the rule is a predicate of the shape $A_i(\alpha_{i1}, \ldots, \alpha_{i\,\dim(A_i)})$, where $A_i \in N$

and each α_{ij} $(1 \leq j \leq \dim(A_i))$ is just a variable in V if $1 \leq i \leq k$. The following is a simple example of rules:

$$S(xyz) \to A(x,y)B(z), \qquad A(axb,cyd) \to A(x,y), \qquad B(ez) \to B(z),$$
$$A(ab,cd) \to \varepsilon, \qquad B(\varepsilon) \to \varepsilon.$$

Let $\Rightarrow$ denote the one-step derivation relation. For example,

$$S(aabbccdde) \Rightarrow A(aabb,ccdd)B(e) \Rightarrow A(ab,cd)B(e) \Rightarrow B(e) \Rightarrow B(\varepsilon) \Rightarrow \varepsilon.$$

Let $\overset{+}{\Rightarrow}$ denote the transitive closure of $\Rightarrow$. The language generated by an RCG G is defined as $L(G) = \{w \mid S(w) \overset{+}{\Rightarrow} \varepsilon\}$. For the above example, $L(G) = \{a^m b^m c^m d^m e^n \mid m \geq 1, n \geq 0\}$. We also say that A *generates* w when $A(w) \overset{+}{\Rightarrow} \varepsilon$.

If every variable occurs at most once in the left-hand side (rsp. right-hand side) of a rule, the rule is called *left linear* (rsp. *right linear*). For example, $S(x) \to S_1(x)S_2(x)$ is left linear but not right linear.

4.2 Modeling by RCG

Chiang et al. [8] presented the following RCG to generate linearly ordered β-sheets:

$$Beta(xy) \to B(x,y), \qquad B(xyz,y') \to B(x,y)Adj(y,y'),$$
$$B(yz,y') \to Adj(y,y'),$$
$$Adj(x,y) \to Anti(x,y), \qquad Adj(x,y) \to Par(x,y),$$
$$Anti(ax,y\overline{a}) \to Anti(x,y), \qquad Anti(\varepsilon,\varepsilon) \to \varepsilon,$$
$$Par(ax,\overline{a}y) \to Par(x,y), \qquad Par(\varepsilon,\varepsilon) \to \varepsilon,$$

where $a, \overline{a} \in T$ stand for amino acid residues that are connected with each other by hydrogen bond. (We extend the notion $\overline{u}$ for a sequence u.) Par and $Anti$ generate parallel and antiparallel strands, respectively. $B(u,v)$ means that uv is a β-sheet where the second argument v is the "last" strand. Thus, the second rule says that if xy is a β-sheet (with y the last strand) and (y,y') constitutes a pair of adjacent strands, then $xyzy'$ is also a β-sheet (with y' the last strand) for an unpaired subsequence z. In this rule, the right nonlinearity plays a crucial role that expresses the constraints that the last strand y should be one component y of pair strands (y,y'). The time complexity of the structure prediction based on parsing of RCG is easily derived by counting the independent positions that appear in the arguments of the left-hand side for each rule and taking the maximum of them. For example, the independent positions are marked by $*_i$ $(1 \leq i \leq 5)$ for the second rule as $B(*_1 x_{*_2} y_{*_3} z_{*_4}, y'_{*_5})$. This is the maximum among all the above rules, thus the complexity is $O(n^5)$ where n is the length of an input sequence.

Returning to the problem of this paper, we assume that the length of each strand is L. This means that $|y| = |y'| = L$ in the second rule, implying that

the position $*_3$ and $*_5$ is determined by $*_2$ and $*_4$, respectively. Thus, the time complexity becomes $O(n^3)$, which is the same order as our algorithm for up-down β-sheet in Section 2. Note that the formalism in [8] does not incorporate residue-residue interaction preferences. Implementation or experimental results on β-sheet prediction based on RCG has not been reported as far as the authors know. On the other hand, we have performed experiments with real protein sequences. Although our algorithms currently consider only antiparallel β-sheets, it is not difficult to extend our proposed algorithms so that parallel structures can be treated, as described in Section 2.2.

5 Hardness Result

Although we have presented an $O(n^3)$ time dynamic programming algorithm in Section 2, it remains a question whether *generalized* ungapped β-sheets can be predicted in polynomial time or not. To discuss the complexity of such a prediction problem, we define the corresponding decision problem as follows:

Definition 2. ***(Ungapped β-sheet prediction problem, UGBETA)***
Input: *An amino acid sequence, a topology diagram and a real number e.*
Output: *"Yes" if and only if there exists an ungapped β-sheet with some free energy e or less.*

In the following, we will show that UGBETA is NP-complete by reducing the longest common subsequence problem that is known to be NP-complete [19]:

Definition 3. ***(Longest common subsequence problem, LCS)***
Input: *m sequences over an alphabet and a positive integer k.*
Output: *"Yes" if and only if there exists a common subsequence of length k or more, which is not necessarily consecutive.*

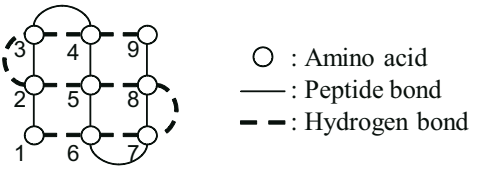
Theorem 1. *UGBETA is NP-complete even if the topology diagram is planar.*

Proof. Assume that each β-strand consists of exactly one amino acid (i.e., $L = 1$) (see Fig. 3). We can also show that NP-completeness result holds for $L \geq 2$.

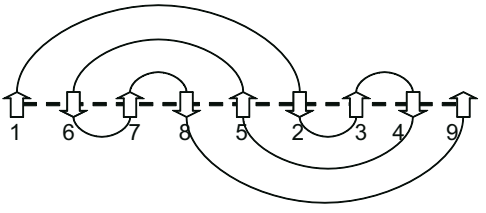
First, it is easy to see that UGBETA belongs to $\mathcal{NP}$. Guess an ungapped β-sheet from the amino acid sequence, and check that it has at most e of free energy value under some energy function.

Next, let us show how to reduce LCS to UGBETA for the proof of NP-hardness. Let $w_1, w_2 \ldots, w_m \in \{0,1\}^*$ be instance sequences of LCS. Without loss of generality, we assume that m is an even number. If it is odd, we simply add a new sequence w_{m+1} that is the same as w_m. Also, we can assume that a positive integer k is an odd number. If it is even, we simply add 0 at the end of each w_i $(i = 1, 2, \ldots, m)$. We construct from $w_1, w_2, \ldots, w_m$ an amino acid sequence $A = B_0 B_1 B_2 \cdots B_m B_{m+1} \in \{0, 1, x, y\}^*$, where

$$B_0 = x(xyx)^{(k+1)/2}y, \qquad B_{2i-1} = xw_{2i-1}xy \quad (i = 1, 2, \ldots, m/2),$$
$$B_{2i} = xw_{2i}^R xy \quad (i = 1, 2, \ldots, m/2), \qquad B_{m+1} = x(xyx)^{(k+1)/2}$$



(a) A β-sheet from the viewpoint of sequence



(b) The topology diagram of β-sheet of (a)

Fig. 3. A simplified ungapped β-sheet ($L = 1$). For simplicity of illustration, we allow hydrogen bond to be compatible with peptide bond.

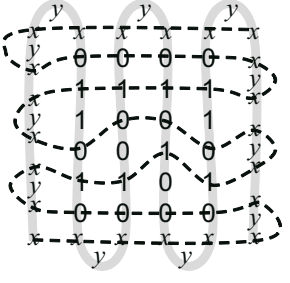


Fig. 4. Example of an amino acid sequence over $\{0, 1, x, y\}$ constructed from an LCS instance, where $w_1 = 011010$, $w_2 = 010010$, $w_3 = 010100$, $w_4 = 011010$, $m = 4$ and $k = 5$.

(see Fig. 4). Note that w_i^R denotes the reverse sequence of w_i, and $B_{i,j}$ that will be used below denotes the jth symbol of B_i. The score (energy) function s is defined in such a way that $s(0,0) = s(1,1) = -1$, $s(x,x) = -\alpha$ where α is set at some positive constant times nm, and defined as 0 for the other pairs. It is obvious that this transformation can be accomplished in polynomial time. Then, we must show the following:

– There exists a common subsequence of length k in $w_1, w_2, \ldots, w_m$ if and only if there exists an ungapped β-sheet of A with free energy $-k(m + \alpha - 1) - \alpha(2m + 3)$.

We omit a detailed proof of the above statement in this version as space is limited. It should be noted that the topology diagram used in this proof is planar (see Fig. 3 (b)). □

6 Concluding Remarks

We presented dynamic programming algorithms for predicting ungapped up-down β-sheet and its extensions. Experimental results on ungapped up-down β-sheet prediction showed that performance is good enough to distinguish β-sheet regions from non-β-sheet ones. However, we have not presented complete comparison with other models for β-sheet prediction, which is left as our future work.

Computational models that predict biomolecule structure with high accuracy are needed in bioinformatics. When we develop a model for prediction, it is important to assign some biologically appropriate score to the model. In our experiments using the dynamic programming algorithms, we used contact potentials and did not perform training from the sequence sets. It might be possible to design a training algorithm based on the EM algorithm, in which case, the prediction accuracy would be higher. If we choose a grammatical approach, training has to be carried out due to the difficulty in assigning optimal probabilities.

As shown in Section 5, arbitrary ungapped planar β-sheet prediction is NP-hard. However, this claim does not always imply that efficient algorithms never exist for small input sets. Most protein sequences consist of at most a few hundred amino acid residues, and there is room for further investigation into the development of efficient algorithms even if topologies that we wish to handle are complex. Furthermore, it is a challenging task to develop an efficient algorithm for predicting protein structures that include the combination of α-helix and β-sheet.

References

1. Abe, N., Mamitsuka, H.: Predicting Protein Secondary Structure Using Stochastic Tree Grammars. Machine Learning 29, 275–301 (1997)
2. Akutsu, T., Miyano, S.: On the Approximation of Protein Threading. Theor. Comp. Sci. 210, 261–275 (1999)
3. Asai, K., Hayamizu, S., Handa, K.: Prediction of Protein Secondary Structure by the Hidden Markov Model. Bioinformatics 9, 141–146 (1993)
4. Berrera, M., Molinari, H., Fogolari, F.: Amino Acid Empirical Contact Energy Definitions for Fold Recognition in the Space of Contact Maps. BMC Bioinformatics 4 (2003)
5. Boullier, P.: Range Concatenation Grammars. In: Sixth Intl.Workshop on Parsing Technologies (IWPT 2000), pp.53–64 (2000)
6. Branden, C., Tooze, J.: Introduction to Protein Structure, 2nd edn. Garland Publishing (1999)
7. Cai, L., Malmberg, R.L., Wu, Y.: Stochastic Modeling of RNA Pseudoknotted Structures: A Grammatical Approach. Bioinformatics 19, i66–i73 (2003)
8. Chiang, D., Joshi, A.K., Searls, D.B.: Grammatical Representations of Macromolecular Structure. J. Comp. Biol. 13, 1077–1100 (2006)
9. Dosztányi, Z., Csizmók, V., Tompa, P., Simon, I.: The Pairwise Energy Content Estimated from Amino Acid Composition Discriminates between Folded and Intrinsically Unstructured Proteins. J. Mol. Biol. 347, 827–839 (2005)
10. Eddy, S.R., Durbin, R.: RNA Sequence Analysis Using Covariance Models. Nucl. Acids Res. 22, 2079–2088 (1994)

11. Hobohm, U., Scharf, M., Schneider, R., Sander, C.: Selection of a Representative Set of Structures from the Brookhaven Protein Data Bank. Protein Sci. 1, 409–417 (1992)
12. Hua, S., Sun, Z.: A Novel Method of Protein Secondary Structure Prediction with High Segment Overlap Measure: Support Vector Machine Approach. J. Mol. Biol. 308, 397–407 (2001)
13. Hubbard, T.J.P.: Use of β-Strand Interaction Pseudo-Potentials in Protein Structure Prediction and Modelling. In: The Twenty-Seventh Annual Hawaii Intl. Conf. on System Sciences, pp. 336–344 (1994)
14. Kabsch, W., Sander, C.: Dictionary of Protein Secondary Structure: Pattern Recognition of Hydrogen-Bonded and Geometrical Features. Biopolymers 22, 2577–2637 (1983)
15. Kato, Y., Seki, H., Kasami, T.: RNA Pseudoknotted Structure Prediction Using Stochastic Multiple Context-Free Grammar. IPSJ Trans. Bioinformatics 47, 12–21 (2006)
16. Krogh, A., Brown, M., Mian, I.S., Sjölander, K., Haussler, D.: Hidden Markov Models in Computational Biology: Applications to Protein Model. J. Mol. Biol. 235, 1501–1531 (1994)
17. Lathrop, R.H.: The Protein Threading Problem with Sequence Amino Acid Interaction Preferences is NP-Complete. Protein Eng. 7, 1059–1068 (1994)
18. Lin, K., Simossis, V.A., Taylor, W.R., Heringa, J.: A Simple and Fast Secondary Structure Prediction Method Using Hidden Neural Networks. Bioinformatics 21, 152–159 (2005)
19. Maier, R.: The Complexity of Some Problems on Subsequences and Supersequences. J. ACM 25, 322–336 (1978)
20. Muggleton, S., King, R., Sternberg, M.: Protein Secondary Structure Prediction Using Logic-Based Machine Learning. Protein Eng. 5, 647–657 (1992)
21. Rivas, E., Eddy, S.R.: The Language of RNA: A Formal Grammar that Includes Pseudoknots. Bioinformatics 16, 334–340 (2000)
22. Rost, B., Sander, C.: Prediction of Protein Secondary Structure at Better than 70% Accuracy. J. Mol. Biol. 232, 584–599 (1993)
23. Sakakibara, Y., Brown, M., Hughey, R., Mian, I.S., Sjölander, K., Underwood, R.C., Haussler, D.: Stochastic Context-Free Grammars for tRNA Modeling. Nucl. Acids Res. 22, 5112–5120 (1994)
24. Tanaka, S., Scheraga, H.A.: Medium- and Long-Range Interaction Parameters between Amino Acids for Predicting Three-Dimensional Structures of Proteins. Macromolecules 9, 945–950 (1976)
25. Uemura, Y., Hasegawa, A., Kobayashi, S., Yokomori, T.: Tree Adjoining Grammars for RNA Structure Prediction. Theor. Comp. Sci. 210, 277–303 (1999)
26. Xu, Y., Xu, D., Uberbacher, E.C.: An Efficient Computational Method for Globally Optimal Threading. J. Comp. Biol. 5, 597–614 (1998)
27. Zhang, C., Kim, S.H.: Environment-Dependent Residue Contact Energies for Proteins. PNAS 97, 2550–2555 (2000)

Using Multi-scale Glide Zoom Window Feature Extraction Approach to Predict Protein Homo-oligomer Types

QiPeng Li, Shao Wu Zhang, and Quan Pan

School of Automation/School of Mechatronics, Northwestern Polytechnical University, 127 YouYi West Rd., Xi'an 710072, Shaanxi, China
liqipeng@nwpu.edu.cn

Abstact. The concept of multi-scale glide zoom window was proposed and a novel approach of multi-scale glide zoom window feature extraction was used for predicting protein homo-oligomers. Based on the concept of multi-scale glide zoom window, we choose two scale glide zoom window: whole protein sequence glide zoom window and kin amino acid glide zoom window, and for every scale glide zoom window, three feature vectors of amino acids distance sum, amino acids mean distance and amino acids distribution, were extracted. A series of feature sets were constructed by combining these feature vectors with amino acids composition to form pseudo amino acid compositions (PseAAC). The support vector machine (SVM) was used as base classifier. The 75.37% total accuracy is arrived in jackknife test in the weighted factor conditions, which is 10.05% higher than that of conventional amino acid composition method in same condition. The results show that multi-scale glide zoom window method of extracting feature vectors from protein sequence is effective and feasible, and the feature vectors of multi-scale glide zoom window may contain more protein structure information.

Keywords: Multi-scale glide zoom window, feature extraction, pseudo amino acid compositions, homo-oligomer.

1 Introduction

In the protein universe, there are many different classes of oligomer, such as monomer, dimer, trimer, tetramer, and so forth. These quaternary structures are closely related to the functions of the proteins [1, 2]. Some special functions are realized only when protein molecules are formed in oligomers; e.g., GFAT, a molecular therapeutic target for type-2 diabetes, performs its special function when it is a dimer [3], some ion channels are formed by a tetramer [4], and some functionally very important membrane proteins are of pentamer [5,6,7]. It is generally accepted that the amino acid sequence of most, not all, proteins contains all the information needed to fold the protein into its correct three-dimension structure structure [8,9]. So, predicting oligomers types from given protein sequences is important.

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 78–86, 2008.

Garian [10], Chou and Cai [11], Zhang [12] predicted homodimer and non-homodimer using decision-tree models and a feature extraction method (simple binning function), pseudo-amino acid composition feature extraction method, amino acid index auto-correlation functions respectively. Zhang [13] also predicted protein homo-oligomer types by pseudo amino acid composition. They found that protein sequences contain quaternary structure information.

The concept of multi-scale glide zoom window based on the protein sequence was proposed in this paper. Three kinds of feature vector incorporating sequence order effect, that is, amino acids distance sum, amino acids mean distance and amino acids distribution , were extracted from whole protein sequence glide zoom window and kin amino acid glide zoom window of protein sequence. This new feature extraction method is combined felicitously with a support vector machine [14, 15] to predict homodimers, homotrimers, homotetramers and homohexamers.

2 Materials and Methods

2.1 Database

The dataset1283 consists of 1283 homo-oligomeric protein sequences, 759 of which are homodimers (2EM), 105 homotrimers (3EM), 327 homotetramers (4EM) and 92 homohexamers (6EM). This dataset was obtained from SWISS-PROT database [16] and limited to the prokaryotic, cytosolic subset of homo-oligomers in order to eliminate membrane proteins and other specialized proteins.

2.2 The Concept of Multi-scale Glide Zoom Window

Multi-scale glide zoom window of every nature amino acid can be described as multi-scale segment sequence (or, whole sequence) of one protein sequence, that is, the every scale glide zoom window of one nature amino acid can be decided by three factors: constructing rule of xth scale glide zoom window, kth protein sequence and ith amino acid. So, for one protein sequence, we can obtain many glide zoom windows and extract feature vectors from every glide zoom window. This novel multi-scale glide zoom window feature extraction method is very depends on constructing rule of every scale glide zoom window. In this paper, we extract feature vectors of one protein sequence from 2-scale glide zoom window. The first scale glide zoom windows of every nature amino acid are all the whole protein sequence, which provide panorama of a protein sequence. The second scale glide zoom window of every nature amino acid are kin amino acid glide zoom window, which begins from the position where every kin amino acid appears firstly and ends at the position where this kin amino acid appears lastly among the whole protein sequence, which focuses on corresponding local of every nature amino acid in a protein sequence. There are one first scale glide zoom window and twenty second scale glide zoom windows for every protein sequence. For example, for the protein sequence 'MITRMSELFLRTLRDDP', the first scale glide zoom windows of every nature amino acid are all the whole protein sequence itself 'MITRMSELFLRTLRDDP'. The second scale glide zoom window of nature amino acid M is 'MITRM', the second scale glide zoom window of nature amino acid T is 'TRMSELFLRT', the second scale glide

zoom window of nature amino acid D is 'DD', and so on. If one nature amino acid does not appear in the protein sequence, the second scale glide zoom window of this nature amino acid is empty. The position and the width of every second scale glide zoom window are variable. Apparently, the second scale glide zoom window contains some sequence order information. The width of first scale glide zoom window is equal to the length of the protein sequence.

2.3 The Multi-scale Glide Zoom Window Feature Extraction Methods

Suppose the dataset consists of N homo-oligomeric protein sequences. p^k represents the kth protein sequence. α_i represents the ith amino acid of the nature amino acid set AA, $AA = \{A, R, N, D, C, O, E, G, H, I, L, K, M, F, P, S, T, W, Y, V\}$. Here, We can use $z_i^{x,k}$ to represent the xth scale glide zoom window of α_i in p^k. $f_i^{x,k}$ and $l_i^{x,k}$ represent the first position and last position of $z_i^{x,k}$ in the kth protein sequence p^k, respectively. $L_i^{x,k}$ is defined as length of $z_i^{x,k}$. According to the definition of first scale glide zoom window in section 2.2, every first scale glide zoom window of α_i in p^k is the same whole sequence. Apparently, $z_i^{1,k}$ is p^k. $L_i^{1,k}$ is the length of p^k ,which we can denote as L^k. $f_i^{1,k}$ and $l_i^{1,k}$ are 1 and L^k respectively. According to the definition of second scale glide zoom window in section 2.2, $f_i^{2,k}$ and $l_i^{2,k}$ are first and last position where α_i appear among p^k , respectively. $z_i^{2,k}$ is segment sequence between $f_i^{2,k}$ and $l_i^{2,k}$. $L_i^{2,k}$ is equal to $l_i^{2,k} - f_i^{2,k}$. In order to describe the positions of every nature amino acid in p^k , We first defined a position indicator $o_{i,j}^k$.

$$o_{i,j}^k = \begin{cases} 1 \text{ if } \alpha_i \text{ locates in } jth \text{ position of } p^k \\ 0 \text{ if } \alpha_i \text{ does not locate in } jth \text{ position of } p^k \end{cases} \tag{1}$$

Then, we map protein sequence p^k to a position indicator matrix V^k.

$$V^k = \begin{bmatrix} v_1^k \\ \cdots \\ v_i^k \\ \cdots \\ v_{20}^k \end{bmatrix} = \begin{bmatrix} o_{1,1}^k, \cdots, o_{1,j}^k, \cdots, o_{1,L^k}^k \\ \cdots, \cdots, \cdots, \cdots, \cdots \\ o_{i,1}^k, \cdots, o_{i,j}^k, \cdots, o_{i,L^k}^k \\ \cdots, \cdots, \cdots, \cdots, \cdots \\ o_{20,1}^k, \cdots, o_{20,j}^k, \cdots, o_{20,L^k}^k \end{bmatrix}_{20 \times L^k}, \quad k = 1, \cdots, N \tag{2}$$

Here, position indicator vector v_i^k shows where α_i locates in the p^k.

In order to extract various feature vectors of $z_i^{x,k}$ with v_i^k, we defined a coordinate axis vector $w_i^{x,k}$.

$$w_i^{x,k} = [\xi_{i,1}^{x,k}, \xi_{i,2}^{x,k}, ..., \xi_{i,j}^{x,k}, ..., \xi_{i,L^k}^{x,k}]_{1\times L^k} \quad , x = 1,2; \ j = 1, ..., L^k \tag{3}$$

Here,

$$\xi_{i,j}^{1,k} = j \quad , j = 1, ..., L^k \tag{4}$$

$$\xi_{i,j}^{2,k} = \begin{cases} j - f_i^{2,k} + 1 & \text{if } f_i^{2,k} \le j \le l_i^{2,k} \\ 0 & \text{if } j < f_i^{2,k} \ or \ j > l_i^{2,k} \end{cases} \tag{5}$$

To integrate more sequence order information, according to the concept of multi-scale glide zoom window, three kinds of feature vector of every scale glide zoom window are extracted to predict homo-oligomers. The three kinds of feature vector of every scale glide zoom window are defined as follows:

1) Amino Acids Distance Sum Feature Vector

The amino acids distance sum feature vector of p^k is expressed as the following 20-D feature vector:

$$S^{x,k} = [\eta_1^{x,k}, ..., \eta_i^{x,k}, ..., \eta_{20}^{x,k}] \quad k = 1, \cdots, N \tag{6}$$

Here,

$$\eta_i^{x,k} = w_i^{x,k} \times (v_i^k)^T \quad k = 1, \cdots, N \tag{7}$$

Conveniently, S^1 and S^2 are respectively used to present the amino acids distance sum feature sets of first and second scale glide zoom windows.

2) Amino Acids Mean Distance Feature Vector

The amino acids mean distance feature vector of p^k is expressed as the following 20-D feature vector:

$$M^{x,k} = [\mu_1^{x,k}, ..., \mu_i^{x,k}, ..., \mu_{20}^{x,k}] \quad k = 1, \cdots, N \tag{8}$$

Here,

$$\mu_i^{x,k} = \begin{cases} \dfrac{1}{L_i^{x,k}} w_i^{x,k} \times (v_i^k)^T & , \ \text{if } L_i^{x,k} \ne 0 \\ 0 & , \ \text{if } L_i^{x,k} = 0 \end{cases} \tag{9}$$

Conveniently, M^1 and M^2 are respectively used to present the amino acids mean distance feature sets of first and second scale glide zoom windows.

3) Amino Acids Distribution Feature Vector

The amino Acids distribution feature vector of p^k is expressed as the following 20-D feature vector:

$$D^{x,k} = \left[\rho_1^{x,k}, \cdots, \rho_i^{x,k}, \cdots, \rho_{20}^{x,k} \right], \quad k = 1, \cdots, N \tag{10}$$

Here,

$$\rho_i^{x,k} = \begin{cases} \dfrac{1}{L_i^{x,k}} \displaystyle\sum_{j=f_i^{x,k}}^{l_i^{x,k}} \left(o_{i,j}^k \times j - \dfrac{1}{L_i^{x,k}} w_i^{x,k} \times (v_i^k)^T \right)^2, & \text{if } L_i^{x,k} \neq 0 \\ 0, & \text{if } L_i^{x,k} = 0 \end{cases} \tag{11}$$

Conveniently, D^1 and D^2 are respectively used to present the amino acids distribution feature sets of first and second scale glide zoom windows. It is easy to certified that D^1 is equal to D^2, so, we can marked D^1 and D^2 as D.

2.4 Assessment of the Prediction System

The prediction quality can be examined using the jackknife test. The cross-validation by jackknifing is thought the most objective and rigorous way in comparison with sub-sampling test or independent dataset test [17, 18]. During the process of jackknife analysis, the datasets are actually open, and a protein will in turn move from each to the other. The total prediction accuracy (Q), Sensitivity (Q(class(*k*))) and Matthew's Correlation Coefficient (MCC) [19] for each class of homo-oligomers calculated for assessment of the prediction system are given by:

$$Q = \sum_{k=1}^{M} p_k \Big/ N \times 100\% \tag{12}$$

$$Q\left(class(k)\right) = p_k / \left(p_k + u_k\right) \tag{13}$$

$$MCC(class(k)) = \frac{p_k n_k - u_k o_k}{\sqrt{\left(p_k + u_k\right)\left(p_k + o_k\right)\left(n_k + u_k\right)\left(n_k + o_k\right)}} \tag{14}$$

Here, M is the total number of classes, p_k is the number of correctly predicted sequences of *k* class protein homo-oligomers, u_k is the number of under-predicted sequences of *k* class protein homo-oligomers, n_k is the number of correctly predicted sequences not of *k* class protein homo-oligomers, o_k is the number of over-predicted sequences of *k* class protein homo-oligomers. According to The dataset1283 used in this paper, M=4, class(1), class(2),class(3) and class(4) are 2,3,4 and 6 respectively. 2, 3, 4 and 6 represent 2EM, 3EM, 4EM and 6EM respectively.

3 Results and Discussion

3.1 The Results of Different Pseudo Amino Acids Composition Feature Sets

C presents the feature set based on the amino acid composition approach [20]. Twenty-seven feature sets of pseudo amino acid composition (PseAAC) are constructed by feature sets D, M^1, M^2, S^1, S^2 of glide zoom window and C. The results of these twenty-seven PseAAC feature sets and feature set C with RBF SVM and one-versus-one strategy in jackknife test are shown in table 1.

From Table 1, we can see that the result of $CDM^1M^2S^2$ is the best in all the feature sets, and the total accuracy is 75.53%, which is 6.71% higher than that of C. The accuracies of feature sets which include M^1, M^2 or both of them are higher than that of other feature sets which do not include M^1, M^2 or both of them. These results suggest that, in every scale glide zoom window, the feature set of amino acids mean distance is more effective and robust than other feature sets. In addition, the accuracies of feature sets which include D, S^1, S^2 except M^1 and M^2 are near that of feature set C. The reasons are that there may be some redundancy and conflict information between these feature sets, or the unbalance of sample numbers among the four classes.

Table 1. Results of 28 Feature sets with RBF SVM and one-versus-one strategy in jackknife test

Feature sets	2EM		3EM		4EM		6EM		Q%
	Q(2) %	MCC(2)	Q(3) %	MCC(3)	Q(4) %	MCC(4)	Q(6) %	MCC(6)	
C	**91.57**	**0.3582**	**42.86**	**0.5726**	**38.53**	**0.3568**	**18.48**	**0.3088**	**68.82**
CD	**95.39**	**0.6630**	**32.38**	**0.5276**	**33.03**	**0.3611**	**1.09**	**0.0992**	**67.58**
CM^1	**92.23**	**0.5152**	**50.48**	**0.6621**	**57.49**	**0.5258**	**29.35**	**0.4412**	**75.45**
CM^2	**91.17**	**0.7497**	**53.33**	**0.6511**	**55.35**	**0.5053**	**30.43**	**0.4373**	**74.59**
CS^1	**95.12**	**0.3341**	**32.38**	**0.5188**	**33.95**	**0.3627**	**2.17**	**0.1403**	**67.73**
CS^2	**94.33**	**0.6813**	**36.19**	**0.5150**	**37.61**	**0.3753**	**3.26**	**0.1439**	**68.59**
CDM^1	**92.89**	**0.5051**	**50.48**	**0.6690**	**55.35**	**0.5155**	**26.09**	**0.4318**	**75.06**
CDM^2	**91.04**	**0.7495**	**53.33**	**0.6511**	**55.05**	**0.4989**	**30.43**	**0.4373**	**74.43**
CDS^1	**94.60**	**0.3325**	**32.38**	**0.5188**	**35.17**	**0.3696**	**3.26**	**0.1720**	**67.81**
CDS^2	**95.92**	**0.6612**	**28.57**	**0.4922**	**32.11**	**0.3569**	**1.09**	**0.0992**	**67.34**
CM^1M^2	**92.36**	**0.5013**	**53.33**	**0.6898**	**55.66**	**0.5178**	**25.00**	**0.3955**	**74.98**
CM^1S^1	**91.44**	**0.5105**	**53.33**	**0.6765**	**57.49**	**0.5183**	**30.43**	**0.4447**	**75.29**
CM^1S^2	**91.96**	**0.5113**	**53.33**	**0.6765**	**56.57**	**0.5201**	**29.35**	**0.4267**	**75.29**
CM^2S^1	**91.30**	**0.5025**	**53.33**	**0.6573**	**55.66**	**0.5065**	**32.61**	**0.4587**	**74.90**
CM^2S^2	**91.17**	**0.7514**	**53.33**	**0.6572**	**55.05**	**0.4973**	**31.52**	**0.4480**	**74.59**
CS^1S^2	**95.65**	**0.3347**	**30.48**	**0.5102**	**33.33**	**0.3641**	**1.01**	**0.0992**	**67.65**
CDM^1S^1	**92.23**	**0.5133**	**53.33**	**0.6765**	**56.27**	**0.5235**	**30.43**	**0.4447**	**75.45**
CDM^2S^2	**90.78**	**0.7481**	**53.33**	**0.6634**	**55.35**	**0.4995**	**31.52**	**0.4480**	**74.43**
CDS^1S^2	**94.07**	**0.3429**	**32.38**	**0.5190**	**37.61**	**0.3679**	**3.26**	**0.1720**	**68.12**
$CM^1M^2S^1$	**92.49**	**0.5085**	**53.33**	**0.6899**	**56.27**	**0.5233**	**26.09**	**0.4151**	**75.29**
$CM^1M^2S^2$	**92.36**	**0.5065**	**53.33**	**0.6899**	**56.27**	**0.5213**	**26.09**	**0.4151**	**75.21**
$CM^1S^1S^2$	**92.89**	**0.5137**	**52.38**	**0.6831**	**56.27**	**0.5235**	**26.09**	**0.4319**	**75.45**
$CM^2S^1S^2$	**91.04**	**0.4985**	**53.33**	**0.6573**	**55.96**	**0.5070**	**32.61**	**0.4657**	**74.82**
$CDM^1M^2S^1$	**92.75**	**0.5125**	**53.33**	**0.6900**	**56.27**	**0.5273**	**26.09**	**0.4152**	**75.45**
$CDM^1M^2S^2$	**92.89**	**0.5145**	**53.33**	**0.6900**	**56.27**	**0.5294**	**26.09**	**0.4152**	**75.53**
$CDM^2S^1S^2$	**91.57**	**0.4965**	**53.33**	**0.6635**	**54.43**	**0.5019**	**32.61**	**0.4657**	**74.75**
$CM^1M^2S^1S^2$	**92.23**	**0.5072**	**53.33**	**0.6831**	**56.57**	**0.5218**	**26.09**	**0.4151**	**75.21**
$CDM^1M^2S^1S^2$	**92.36**	**0.5065**	**53.33**	**0.6831**	**56.27**	**0.5250**	**26.09**	**0.4073**	**75.21**

3.2 The Influence of the Unbalance of Sample Numbers among the Four Classes

We used the weighted factor approach to investigate the influence of the sample unbalance among the four classes. According to the number of four types of protein homo-oligomer, the weighted factor values of 2EM, 3EM, 4EM and 6EM are calculated as follow: 759/759, 759/105, 759/327, 759/92. The results of twenty-eight feature sets using weighted factor approach are shown in table 2.

From table 2, we can see that, in the weighted factor conditions, the total accuracies of all feature sets except CS^1S^2 based on the two scale glide zoom window are higher than that of C. The result of $CDM^1M^2S^1$ is the best, and the total accuracy are 75.37%, which are 10.05 higher than that of feature set C. These results suggest that weighted factor approach can weaken influence of the unbalance of sample numbers among the four classes.

Table 2. Results of 28 feature sets with RBF SVM and one-versus-one strategy in jackknife test using weighted factor approach

Feature sets	2EM		3EM		4EM		6EM		Q%
	Q(2)%	MCC(2)	Q(3) %	MCC(3)	Q(4) %	MCC(4)	Q(6) %	MCC(6)	
C	**70.36**	**0.3577**	**49.52**	**0.4772**	**63.91**	**0.3859**	**46.74**	**0.3752**	**65.32**
CD	**76.02**	**0.4105**	**53.33**	**0.5213**	**64.83**	**0.4383**	**42.39**	**0.4092**	**68.90**
CM^1	**78.79**	**0.4881**	**59.05**	**0.5911**	**69.72**	**0.5127**	**51.09**	**0.4983**	**72.88**
CM^2	**78.00**	**0.4647**	**59.05**	**0.5532**	**67.58**	**0.5035**	**53.26**	**0.5188**	**72.02**
CS^1	**74.31**	**0.4163**	**57.14**	**0.5196**	**65.75**	**0.4571**	**48.91**	**0.4237**	**68.90**
CS^2	**76.81**	**0.4363**	**55.24**	**0.5371**	**66.36**	**0.4665**	**45.65**	**0.4305**	**70.15**
CDM^1	**78.92**	**0.4838**	**60.00**	**0.5981**	**68.50**	**0.5041**	**51.09**	**0.4982**	**72.72**
CDM^2	**78.79**	**0.4723**	**60.00**	**0.5677**	**66.97**	**0.5039**	**54.35**	**0.5356**	**72.49**
CDS^1	**75.89**	**0.4327**	**58.10**	**0.5375**	**65.44**	**0.4609**	**47.83**	**0.4312**	**69.76**
CDS^2	**75.76**	**0.4271**	**57.14**	**0.5300**	**64.83**	**0.4537**	**46.74**	**0.4127**	**69.37**
CM^1M^2	**82.35**	**0.5150**	**60.95**	**0.6450**	**68.50**	**0.5279**	**51.09**	**0.5463**	**74.82**
CM^1S^1	**78.52**	**0.4833**	**59.05**	**0.5991**	**69.42**	**0.5031**	**51.09**	**0.5020**	**72.64**
CM^1S^2	**80.24**	**0.4931**	**57.14**	**0.5811**	**69.72**	**0.5265**	**51.09**	**0.5275**	**73.58**
CM^2S^1	**78.52**	**0.4763**	**60.00**	**0.5713**	**68.20**	**0.5054**	**53.26**	**0.5355**	**72.56**
CM^2S^2	**78.39**	**0.4735**	**59.05**	**0.5604**	**67.89**	**0.5025**	**53.26**	**0.5270**	**72.33**
CS^1S^2	**65.88**	**0.3722**	**62.86**	**0.4681**	**64.53**	**0.4211**	**51.09**	**0.3296**	**64.22**
CDM^1S^1	**80.37**	**0.4797**	**56.19**	**0.5736**	**67.58**	**0.5117**	**53.26**	**0.5533**	**73.19**
CDM^2S^2	**80.24**	**0.4837**	**60.00**	**0.5866**	**66.36**	**0.5077**	**54.35**	**0.5443**	**73.19**
CDS^1S^2	**77.47**	**0.4424**	**58.10**	**0.5450**	**64.83**	**0.4686**	**47.83**	**0.4485**	**70.54**
$CM^1M^2S^1$	**82.48**	**0.5172**	**61.90**	**0.6520**	**68.20**	**0.5258**	**52.17**	**0.5646**	**74.98**
$CM^1M^2S^2$	**82.21**	**0.5085**	**61.90**	**0.6519**	**67.28**	**0.5164**	**52.17**	**0.5595**	**74.59**
$CM^1S^1S^2$	**79.18**	**0.4843**	**57.14**	**0.5848**	**69.42**	**0.5103**	**51.09**	**0.5102**	**72.88**
$CM^2S^1S^2$	**76.68**	**0.4546**	**62.86**	**0.5643**	**66.67**	**0.4823**	**53.26**	**0.5264**	**71.32**
$CDM^1M^2S^1$	**83.27**	**0.5246**	**61.90**	**0.6522**	**67.89**	**0.5328**	**52.17**	**0.5648**	**75.37**
$CDM^1M^2S^2$	**83.16**	**0.5255**	**61.90**	**0.6522**	**68.20**	**0.5322**	**51.09**	**0.5513**	**75.29**
$CDM^2S^1S^2$	**80.50**	**0.4830**	**60.95**	**0.5899**	**65.44**	**0.5019**	**54.35**	**0.5529**	**73.19**
$CM^1M^2S^1S^2$	**83.14**	**0.5176**	**61.90**	**0.6521**	**66.97**	**0.5236**	**52.17**	**0.5646**	**75.06**
$CDM^1M^2S^1S^2$	**83.53**	**0.5223**	**61.90**	**0.6568**	**66.97**	**0.5269**	**52.17**	**0.5647**	**75.29**

4 Conclusion

A novel concept of multi-scale glide zoom window was proposed in this paper. Based on the concept of multi-scale glide zoom window, a protein sequence can be investigated from two scale glide zoom windows (whole protein sequence glide zoom

window and kin amino acid glide zoom window). Twenty-seven feature sets were constructed by combining five kinds of feature sets of the two scale glide zoom windows with amino acids composition to form pseudo amino acid compositions (PseAAC). The results show that the twenty-six feature sets based on the two scale glide zoom windows are better than feature set C in the weighted factor conditions, and weighted factor approach can weaken influence of the unbalance of sample numbers among the four classes. In the three kinds of feature sets of the two scale glide zoom window, amino acids mean distance feature set is most effective and robust. It is demonstrated that the concept of multi-scale glide zoom window provide a new scope to investigate primary protein sequence, the feature sets extracted from multi-scale glide zoom window may contain more protein structure information.

Acknowledgements. This paper was supported in part by the National Natural Science Foundation of China (No. 60775012 and 60634030) and the Technological Innovation Foundation of Northwestern Polytechnical University (No. KC02).

References

1. Chou, K.C.: Review: Low-frequency Collective Motion in Biomacromolecules and Its Biological Functions. Biophys. Chem. 30, 3–48 (1988)
2. Chou, K.C.: Review: Structural Bioinformatics and Its Impact to Biomedical Science. Curr. Med. Chem. 11, 2105–2134 (2004e)
3. Chou, K.C.: Molecular Therapeutic Target for Type-2 Diabetes. J. Proteome. Res. 3, 1284–1288 (2004a)
4. Chou, K.C.: Insights from Modelling Three-dimensional Structures of the Human Potassium and Sodium Channels. J. Proteome. Res. 3, 856–861 (2004b)
5. Chou, K.C.: Insights from Modelling the 3D Structure of the Extracellular Domain of Alpha7 Nicotinic Acetylcholine Receptor. Biochem. Biophys. Res. Commun. 319, 433–438 (2004c)
6. Chou, K.C.: Modelling Extracellular Domains of GABA-A Receptors: Subtypes 1, 2, 3, and 5. Biochem. Biophys. Res. Commun. 316, 636–642 (2004d)
7. Oxenoid, K., Chou, J.J.: The Structure of Phospholamban Pentamer Reveals a Channel-Like Architecture in Membranes. Proc. Natl. Acad. Sci. USA 102, 10870–10875 (2005)
8. Anfinsen, C.B., Haber., E., Sela, M., White, F.H.: The Kinetics of the Formation of Native Ribonuclease During Oxidation of the Reduced Polypeptide Chain. Proc. Natl. Acad. Sci. USA 47, 1309–1314 (1961)
9. Anfisen, C.B.: Principles that Govern the Folding of Protein Chains. Science 181, 223–230 (1973)
10. Garian, R.: Prediction of Quaternary Structure from Primary Structure. Bioinformatics 17, 551–556 (2001)
11. Chou, K.C., Cai, Y.D.: Predicting Protein Quaternary Structure by Pseudo Amino Acid Composition. Proteins Struct. Func. Gene. 53, 282–289 (2003b)
12. Zhang, S.W., Quan, P., Zhang, H.C., Zhang, Y.L., Wang, H.Y.: Classification of Protein Quaternary Structure with Support Vector Machine. Bioinformatics 19, 2390–2396 (2003)
13. Zhang, S.W., Pan., Q., Zhang., H.-C., Shao., Z.-C., Shi, J.-Y.: Prediction of Protein Homo-oligomer Types by Pseudo Amino Acid Composition: Approached with an Improved Feature Extraction and Naive Bayes Feature Fusion. Amino Acids 30, 461–468 (2006)

14. Vapnik, V.: The Nature of Statistical Learning Theory. Springer, New York (1995)
15. Vapnik, V. (ed.): Statistical Learning Theory. Wiley, New York (1998)
16. Bairoch, A., Apweiler, R.: The SWISS-PROT Protein Data Bank and Its New Supplement TrEMBL. Nucleic Acids Res. 24, 21–25 (1996)
17. Chou, K.C., Zhang, C.T.: Prediction of Protein Structural Classes. Crit. Rev. Biochem. Mol. Biol. 30, 275–349 (1995) (review)
18. Zhou, G.P., Assa-Munt, N.: Some Insights Into Protein Structural Class Prediction. Proteins Struct. Funct. Genet. 44, 57–59 (2001)
19. Fasman, G.D. (ed.): Handbook of Biochemistry and Molecular Biology, 3rd edn. CRC Press, Boca Raton (1976)
20. Bahar, I., Atilgan, A.R., Jernigan, R.L., Erman, B.: Understanding the Recognition of Protein Structural Classes by Amino Acid Composition. Proteins 29, 172–185 (1997)

Extraction of Binding Sites in Proteins by Searching for Similar Local Molecular Surfaces

Satoshi Koizumi[1], Keisuke Imada[2], Tomonobu Ozaki[3], and Takenao Ohkawa[1]

[1] Graduate School of Engineering, Kobe University
kizm-sts@cs25.scitec.kobe-u.ac.jp
[2] Graduate School of Science and Technology, Kobe University
[3] Organization of Advanced Science and Technology, Kobe University

Abstract. There is much research on the automatic extraction of new binding sites in proteins by searching for common sites in proteins with identical functions. While many binding sites consist of concave structures, it is difficult to compare such concaves directly due to the various sizes of concaves. To cope with this difficulty and to realize detailed and precise comparisons between concaves, we propose a method of searching for and comparing concaves by gradually changing the size. By experiments with enzyme proteins, we confirmed that extraction accuracy for the binding sites is improved.

1 Introduction

The functional analysis of proteins is an important research area for elucidating the mechanism of living bodies. Recently, a variety of papers concerning the analysis of protein, e.g, constituent atoms, amino acid sequences and characteristic structures, have been published [1]. The sites on the molecular surface of a protein related to functions are called *functional sites*, and specifying them can provide clues for further analysis. Some proteins can function by binding to other proteins or compounds (ligands) at functional sites. Moreover, it is well-known that the surface shape and the physical properties of binding sites are involved in bindings to ligands because binding occurs on the molecular surface [2]. Therefore, analysis at the functional sites and the molecular surfaces is useful for specifying the protein function [3,4]. For example, one can identify an unknown binding site by searching for structures that resemble well-known binding sites as well as by extracting structurally common sites within proteins that have the same function. The local pattern that commonly appears in a group of proteins is generally known as a motif. Moreover, various kinds of protein motifs are based on target patterns. While a sequential pattern that repeatedly appears in the base sequence and amino acid sequence is called a *sequence motif*, a structual pattern that appears in the structural feature is called a *structural motif*. These motifs extracted from proteins having the same function often correspond to functional or binding sites. Moreover, a binding site, which usually forms a concavity called a *pocket*, is regarded as a structural motif candidate. Therefore, searching for similar pockets within proteins that have the same function helps

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 87–97, 2008.

specify binding sites. However, deciding the size of pockets for similarity evaluation beforehand is difficult because the size of pockets may vary from protein to protein.

In this paper, we approach the problem of flexible comparisons between pockets. We propose the following way to evaluate similarity between pockets. If we specify the size of the local sites of pockets, the local sites of pockets with specified size can be extracted, and we can compute similarity between local sites. While slightly modifying the size, we repeatedly perform the above computation. Similarity between pockets is the highest similarity between local sites. Comparisons between pockets that do not depend on pocket size are enabled by focusing on local sites in pockets. Pockets of proteins with identical functions are compared using a defined similarity measure to extract similar pockets as motifs.

The rest of this paper is organized as follows. In Section 2, we introduce motifs on molecule surfaces. In Section 3, the proposed framework is described in detail. In Section 4, the parameter settings for the experiments are described. After mentioning the experimental results in Section 5, we conclude this paper in Section 6.

2 Motifs on Molecule Surfaces

2.1 Protein Motifs

Locally common amino acid residues exist in amino acid sequences of proteins having the same function. They are called sequence motifs. Since functional and binding sites are often included in a sequence motif, sequence motifs are considered candidates of binding sites. Sequence motifs are crucial, but motifs based on structural data have recently become of major interest because of the following facts [5]:

- **Extraction of residues located far from each other in the sequence**
  Amino acid residues that are located far from each other in the sequence, despite located close to each other in space, interact to form an binding site (Fig. 1). Thus, it is difficult for these amino acid residues to be defined as a sequence motif.
- **Evolutionary conservation of structural features**
  An amino acid sequence is altered during evolution. On the other hand, the structural features of a protein tend to be conserved more than amino acid sequences. Structural motifs provide biologically and evolutionarily interesting insights and help predict protein functions.

In this paper, local patterns that commonly appear on molecular surfaces are defined as surface motifs (hereafter, motif).

2.2 Pockets as Motif Candidates

A protein family is a group of proteins with similar functions. Some functional sites, which commonly appear in each member of a protein family, have a similar

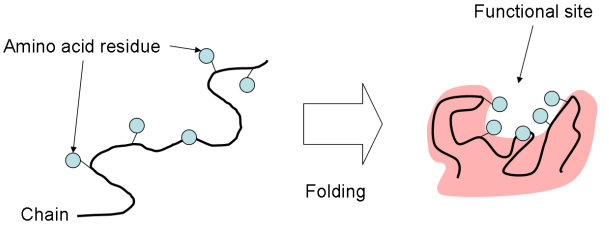


Fig. 1. Folding of distant amino acid

shape on the molecular surface and similar physical properties. Moreover, binding sites include a complex shape structure on molecule surfaces. For instance, when a serine protease acts as a catalyst, an other protein binds nonpolar pockets in the neighborhood of its functional site. Thus, the pocket is a candidate for a binding site [6]. We consider extracting motifs from surface data as extracting similar pockets among a protein family.

We use the surface data in eF-site[1] to extract the motifs. The surface data consist of polygons, and each polygon vertex has its position and physical properties (maximum curvature, minimum curvature, electrostatic potential, and hydrophobicity). These data are provided in an xml format. An example of surface data is shown in Fig. 2.

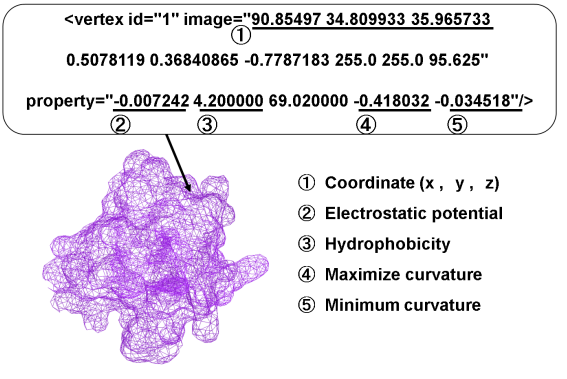


Fig. 2. Example of surface data

2.3 Extraction of Pockets

In this paper, we attempt to extract pockets using curvature. Gaussian curvature K and mean curvature H are defined as follows using maximum curvature κ_{max} and minimum curvature κ_{min} [8]:

$$K = \kappa_{max} \cdot \kappa_{min} \quad , \quad H = \frac{1}{2}(\kappa_{max} + \kappa_{min}).$$

[1] http://ef-site.hgc.jp/eF-site/[7]

Each vertex belongs to one of eight shapes base on the values of K and H. One is called concave if and only if $K > 0$ and $H > 0$. We extract the set of vertices that belongs to concave as a pocket using the region growing method [9]. That is, a pocket is extracted as a set of vertices. To remove pockets that are too small to be binding sites, we set a lower bound of the number of vertices that constitute a pocket. In addition, the cavities inside the protein are excluded because they do not appear on the surface.

3 Motif Extraction

3.1 Overview

In this section, we give an overview over our method for extracting the motifs of proteins, namely, for extracting similar pockets. A family that consists of n proteins is denoted as $\mathscr{F} = \{P_1, P_2, \ldots, P_n\}$. The set of pockets of each protein P_i is denoted as $mc(P_i) = \left\{p_1^i, p_2^i, \ldots\right\}$, and we consider the Cartesian product set of pocket set $S(\mathscr{F}) = mc(P_1) \times mc(P_2) \times \ldots \times mc(P_n)$ in $\mathscr{F}$ and call it a motif group. In this paper, we rank an element of a motif group using some similarity measure and extract pockets in superior elements as the motif in each protein. However, a multiple comparison of an element of the motif group is difficult because, if a family has n proteins, each of which has 30 pockets, $\mid S(\mathscr{F}) \mid = 30^n$, we conduct pairwise comparison. The motif extraction procedure is as follows.

1. First, given pocket p, $msp(p, P)$ denotes a pocket in protein P that is the most similar to p, called the most similar pocket. The formal definition of the similarity between pockets will be explained later in Section 3.2. Next, the most similar pockets are calculated for all pockets in P_i. The set of the pairs of a pocket in P_i and the most similar pocket in P_j, denoted as $pair(P_i, P_j)$, are formally defined as follows:

$$pair(P_i, P_j) = \{\langle p, q\rangle \mid p \in mc(P_i), q = msp(p, P_j)\}\,. \tag{1}$$

2. $M(\mathscr{F})$ is obtained by applying the above operation to all proteins in protein family $\mathscr{F}$:

$$M(\mathscr{F}) = \bigcup_{x,y \in \mathscr{F}, x \neq y} pair(x, y). \tag{2}$$

3. Finally, an element of motif group $s \in S(\mathscr{F})$ is ranked using the following score:

$$score(s) = \mid \{x, y \in s \mid \langle x, y\rangle \in M(\mathscr{F})\} \mid . \tag{3}$$

Equation (3) is based on the idea that the pocket equivalent to the motif has a lot of frequency that is most similar pockets for pairwise comparison. The elements of $S(\mathscr{F})$ are ranked using Equation (3), and the pockets in the superior one are extracted as the motif in each protein. An example of the process of the

extraction of motifs from $\mathscr{F} = \{P_1, P_2, P_3\}$ is illustrated in Fig. 3. Proteins P_1, P_2, and P_3 only have four, three, and three pockets respectively. First, the most similar pocket is calculated for each pocket of proteins P_1, P_2, and P_3 (Fig. 3 ①). For instance, if the most similar pocket of p_1^1 is p_2^2, then (p_1^1, p_2^2) becomes an element of $pair(P_1, P_2)$. $M(\mathscr{F})$ is the union of $pair(P_1, P_2)$, $pair(P_1, P_3)$, $pair(P_2, P_1)$, $pair(P_2, P_3)$, $pair(P_3, P_1)$, $pair(P_3, P_1)$, and $\mid M(\mathscr{F}) \mid = 20$ in Fig. 3. Next, the element of the motif group is ranked using Equation (3). In the case of s_4, $score(s_4)$ is the number of arbitrary pairs of elements in $s4$ that are also in $M(\mathscr{F})$ (Fig. 3②). As a result of ranking, s_5 has the high score, and pockets in s_5 are extracted as motifs on proteins P_1, P_2, P_3 (Fig. 3③).

In the above method, one crucial thing to consider is the similarity measure between pockets, which is needed to get the most similar pocket $msp(p, P)$. It is difficult to compare pockets directly because sizes differ in each protein. In the next section, we introduce a similarity measure between pockets.

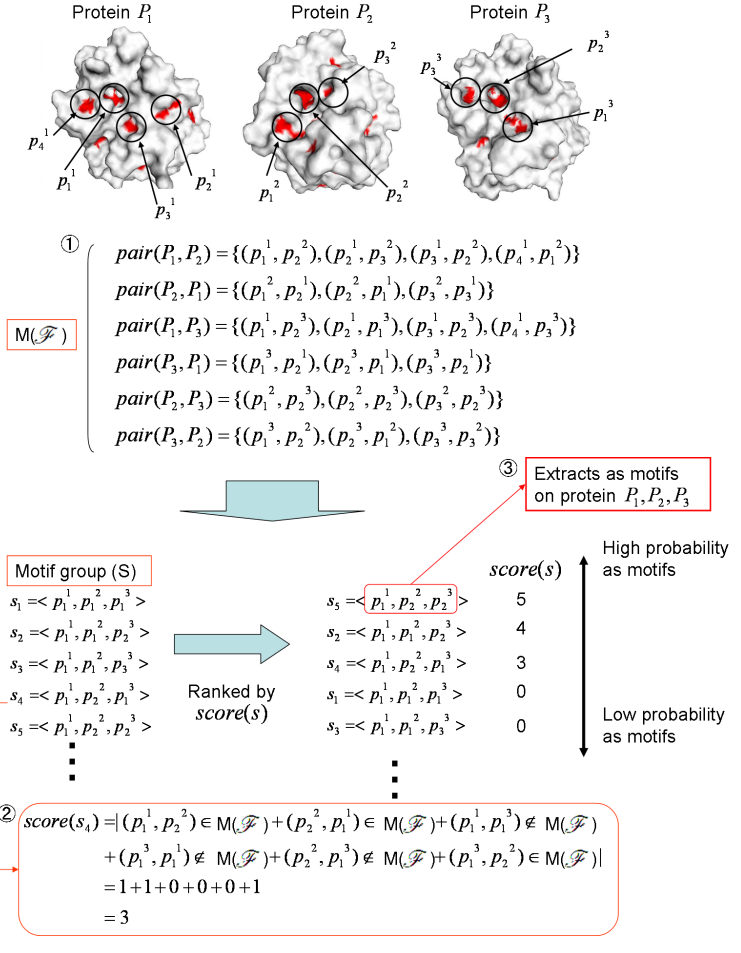


Fig. 3. Overview of motif extraction

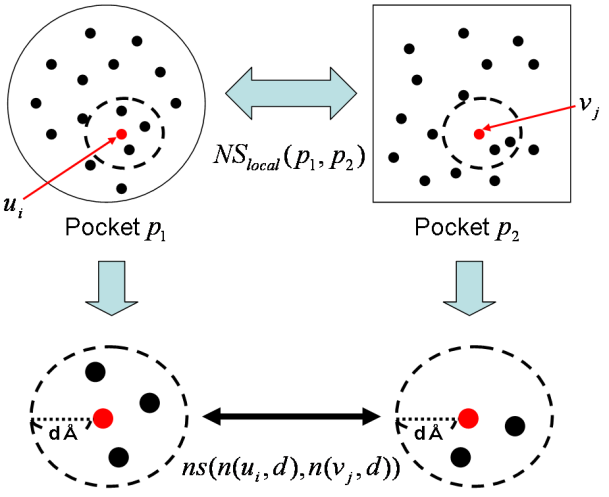


Fig. 4. Dissimilarity between pockets

3.2 Dissimilarity between Pockets

As mentioned above, the surface data consist of polygons, and each polygon vertex has its position and physical properties. In this section, we present a similarity measure between pockets that is independent of the size of pockets by comparing neighboring vertices that compose pockets. The procedure is described as follows (Fig. 4).

1. Each pocket is represented as a set of vertices. Consider two pockets, $p_1 = \{u_1, u_2, \ldots\}$ and $p_2 = \{v_1, v_2, \ldots\}$, where u_i and v_j are the vertices of p_1 and p_2. Moreover, let $n(u_i, d)$ be a set of neighboring vertices located within $d\mathring{A}$ from u_i.
2. The distance between two sets of neighboring vertices $n(u_i, d)$ and $n(v_j, d)$ is denoted as $ns(n(u_i, d), n(v_j, d))$. To evaluate the distance of neighboring vertices, we use physical properties and give three definitions of the distance between two sets of neighboring vertices: $ns1$, $ns2$, and $ns3$. They are defined by combining the average and the variance, which are representative measures for set comparisons:

$$ns1(n(u_i,d),n(v_j,d)) = \sum_{i \in c,h} |ave_i(n(u_i,d)) - ave_i(n(v_j,d))| \quad (4)$$

$$ns2(n(u_i,d),n(v_j,d)) = \sum_{i \in c,h} |ave_i(n(u_i,d)) \cdot var_i(n(u_i,d)) - ave_i(n(v_j,d)) \cdot var_i(n(v_j,d))| \quad (5)$$

$$ns3(n(u_i,d),n(v_j,d)) = \sum_{i \in c,h} |ave_i(n(u_i,d)) - ave_i(n(v_j,d))| + |var_i(n(u_i,d)) - var_i(n(v_j,d))|, \quad (6)$$

where $ave_c(X)$ and $ave_h(X)$ denote the average value of the electrostatic potential and hydrophyobicity of vertex set X. Similarly, $var_c(X)$ and $var_h(X)$ denote those variances. Note that electrostatic potential and hydrophobicity are normalized from 0 to 1.

3. The dissimilarity between pockets p_1 and p_2 is defined as the minimum distance of neighboring vertices in p_1 and p_2:

$$NS_{local}(p_1, p_2) = min_{u_i \in p_1, v_j \in p_2}(ns(n(u_i, d), n(v_j, d))). \quad (7)$$

From the above definition, the most similar pocket is defined about pocket p and protein P using the dissimilarity between pockets:

$$msp(p, P) \Leftrightarrow q \in mc(P),\ s.t.\ \forall x \in mc(P), NS_{local}(p, q) \leq NS_{local}(p, x). \quad (8)$$

The first experiment assessed which equation is more suitable to define distance. This experiment used ten proteins (1owe-A, 1owd-A, 1gjc-B, 1sqt-A, 1sqo-A, 1sqa-A, 1owi-A, 1u6q-A, 1gj7-AB, and 1owk-A) that belong to a urokinase-type plasminogen activator, where 1owe is PDB-ID and A is a chain name. The Structure Classification of Protein (SCOP [10]) is referred to for obtaining information about the protein family. The ten proteins are divided into one training protein and nine test proteins. The pocket that corresponds to the binding site in the training protein is compared to pockets in the test proteins. The above operation is iterated ten times by altering the training protein. The most similar pocket in each test protein is obtained using Equation (8). Note that to calculate the most similar pocket in (8), we need Equation (7), which must be instantiated by Equations (4), (5), or (6). If the most similar pocket in each test protein is actually a binding site, we consider that the method has successfully obtained correct pockets in the test protein. To judge whether the pocket is actually a binding site, we use the information on the nonpolar pockets located in neighborhood of functional sites as a binding site. PROSITE[2] is used as the information of functional sites. The accuracy, which is the ratio of successfully obtaining binding sites, of the three definitions (4), (5), and (6) is 56%, 67%, and 87%, respectively. From these results, Equation (6) is employed as a distance measure between neighboring vertices.

The second experiment confirmed the effectiveness of using local parts of pockets. We compared the proposed dissimilarity measure to that based on all vertices in pockets. The dissimilarity between pockets using all vertices in pockets is formally defined as follows:

$$NS_{global}(p_1, p_2) = ns(p_1, p_2). \quad (9)$$

Ten experiments were conducted by altering a training protein, as in the first experiment, and we aggregated the results for each training protein. The result is shown in Fig. 5. In this experiment, the neighboring range was set to $4\mathring{A}$. The horizontal axis is a protein ID (PDB-ID), and the vertical axis is the accuracy of detecting a correct pocket. "Global" means the results obtained using all the vertices in the pockets, and "Local" means the results obtained using the neighboring vertices. We see that "Local" consistently outperforms "Global".

[2] http://www.expasy.org/prosite/[11]

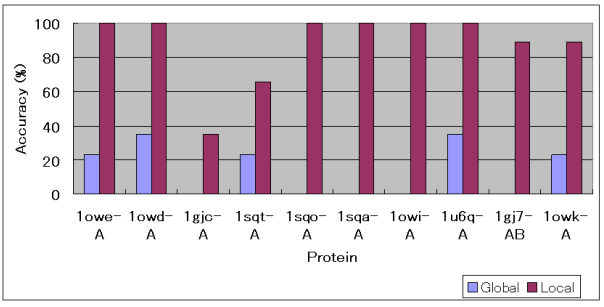


Fig. 5. Efectiveness of using local part of pockets

In this experiment, the neighboring range is 4Å, but it is not obvious which neighboring range is really effective. If the neighboring range is too small, it may cause sites noise that is accidentally similar. On the other hand, if the neighboring range is too wide, it targets all the vertices of the pockets. In the next section, we introduce a method that dynamically determines the neighboring range.

4 Automatic Setting of Neighboring Range

4.1 Alternation of Neighboring Range

Since it is not obvious how far the neighboring range is effective for comparing pockets, we show the influence of the neighboring range in pocket comparisons. We used a family containing ten proteins (1gbt-A, 1fn8-A, 1f0t-A, 1eb2-A, 1fy5-A, 1fn6-A, 1fni-A, 1bra-A, 1co7-E, and 1fy8-E) that belong to a trypsin. Ten experiments were conducted by altering training proteins, as in the first experiment in Section 3.2, and we altered the neighboring range from 0.25 to 6.0Å. The accuracy of detecting a correct pocket is shown in Fig. 6, where the optimal

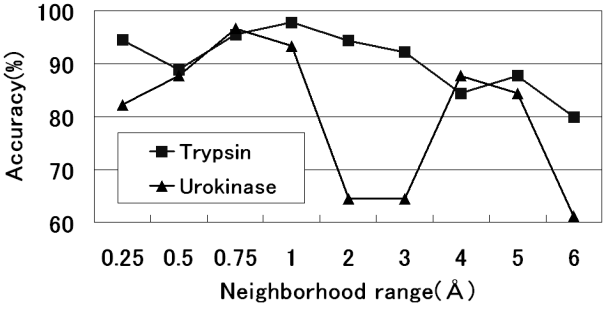


Fig. 6. Validation of neighboring range

neighboring range differs with each protein. The accuracy greatly differs with the neighboring range, which suggests that the optimal neighboring range value should be explored automatically. The neighboring range is expanded stepwisely, and the most similar pockets are calculated using the optimal neighboring range.

4.2 Similar Neighboring Range Prior Method

The similar neighboring range prior method (SNP method) gives priority to the most similar neighboring range between pockets. An overview of the SNP method is shown in Fig. 7. This method is based on the idea that the important neighboring range about binding site is restricted. If the dissimilarity between $p_1 = \{u_1, u_2, \ldots\}$ and $p_2 = \{v_1, v_2, \ldots\}$ is calculated, the dissimilarity between pockets is redefined as Equation (10). Note that dissimilarity between pockets is reflected in neighboring range $d\mathring{A}$, because similarity in a narrow range tends to include noise. Less the dissimilarity between pockets means more similar between pockets:

$$NS(p_1, p_2)_{local} = min_d\left(\frac{min(ns(n(u_i, d), n(v_j, d)))}{d}\right). \tag{10}$$

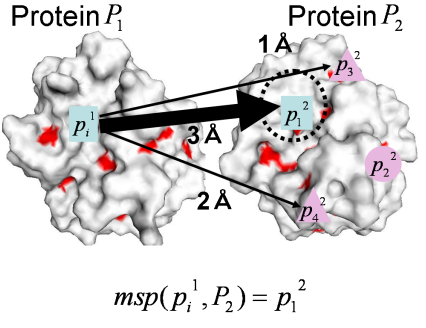


Fig. 7. SNP method

5 Experiments and Results

To verify the effectiveness of the proposed method, we conducted experiments for extracting motifs. We also evaluated whether the extracted motif is a binding site. Nonpolar pockets located in neighborhood of functional sites were used as correct data. The information of the function sites was obtained from PROSITE. The proposed method needs family information because it extracts common pockets within the same family as motifs. In this experiment, we used family information from SCOP and proteins classified as serine proteases. The family information used is shown in Table 1. In the automatic settings of the neighboring range, the initial range was $0.25\mathring{A}$, and the upper bound of the neighboring

Table 1. Family and member proteins (PDB-ID)

Family	Protein
Chymotrypsin	2gmt, 1cho-E, 1gcd, 1gl0-E, 1acb-E
Protease B	1sgq-E, 1sgp-E, 1ds2-E, 1ct4-E, 1ct2-E
Trypsin	1gbt, 1fn8-A, 1f0t-A, 1eb2-A, 1fy5-A
Alpha-Lytic protease	1ssx-A, 1qq4-A, 1p12-E, 1qrw-A, 1qrx-A
Urokinase-type plasminogen activator	1owe-A, 1owd-A, 1gjc-B, 1sqt-A, 1sqo-A
Coagulation factor VIIa	1dva-H, 1o5d-H, 1klj-H, 1dan-H, 1kli-H

Table 2. Result of extracted motifs

	RANK (SNP)	SCORE (SNP)
Chymotrypsin	2	17
Protease B	1	20
Trypsin	1	19
Alpha-Lytic protease	7	14
Urokinase-type plasminogen activator	1	16
Coagulation factor VIIa	1	17

range was $6\mathring{A}$. All experiments were done on an Intel Xeon 2.80 GHz PC with 2 GB of main memory running Debian Linux (32 bits). The experiment runtime to extract motifs from six families was about half a day. The results of the extraction of motifs are shown in Table 2, where SCORE means the *score* value of the elements of the motif group. RANK means the level at which the elements of the motif group are ranked by their $SCORE$s. A motif group has thousands of elements, but pockets in the elements of the top-ranked motif group are binding sites.

One method closely related to our work is LFM-Pro [12], which is a framework for identifying family specific local sites. LFM-Pro resembles our method in terms of extracting family specific structural features. We focus on the pockets in a protein, and features are extracted as surface motifs, which are a portion of the molecular surface. On the other hand, in LFM-Pro, geometrically significant local structural centers are first identified, and then the geometrical and biochemical environment around these centers are evaluated at the atom-level to distinguish a target family. Quantitative comparison with LFM-Pro is future work.

6 Conclusion

In this paper, we proposed a method of extracting binding sites from protein molecular surfaces using a similarity measure between pockets by comparing neighboring vertices. We successfully found binding sites in enzyme proteins by applying the proposed method.

The proposed method assumes that information about the protein family can be clarified in advance. If we cannot get complete information about the protein family, we will explore a new method in which protein-protein interaction is employed complementarily as a substitute for family information. In addition, applying the proposed method to the protein classification problem is a crucial remaining work.

References

1. Suyama, M., Ohhara, O.: Domcut: prediction of inter-domain linker regions in amino acid sequences. Bioinformatics 19(5), 673–674 (2003)
2. Goto, S., Nishioka, T., Kanehisa, M.: Ligand: chemical database for enzyme reactions. Bioinformatics 14, 591–599 (1998)
3. Shrestha, N.L., Kawaguchi, Y., Nakagawa, T., Ohkawa, T.: A method of filtering protein surface motifs based on similarity among local surfaces (2004)
4. Liang, J., Edelsbrunner, H., Woodward, C.: Anatomy of protein pockets and cavities: Measurement of binding site geometry and implications for ligand design. Protein Science 7, 1884–1897 (1998)
5. Shrestha, N.L., Kawaguchi, Y., Ohkawa, T.: Sumomo: A protein surface motif mining module. International Journal of Computational Intelligence and Applications 4(4), 431–449 (2004)
6. Stryer, L.: Biochemistry, Tokyou Kagaku Doujin
7. Kinoshita, K., Nakamura, H.: ef-site and pdbjviewer: database and viewer for protein functional sites. Bioinformatics 20(8), 1329–1330 (2004)
8. Goldman, B.B., Wipke, W.T.: Qsd quadratic shape descriptors. 2. molecular docking using quadratic shape descriptors (qsdock). PROTEIN: Structures, Function, and Genetics 38, 79–94 (2000)
9. Takagi, M., Shimoda, H.: Handbook of image analysis. Tokyo University Publication (1991)
10. Murzina, A.G., Brennera, S.E., Hubbarda, T., Chothia, C.: Scop: a structural classification of proteins database for the investigation of sequences and structures. J. Mol. Biol. 247, 536–540 (2004)
11. Sigrist, C.J.A., Cerutti, L., Hulo, N., Gattiker, A., Falquet, L., Pagnia, M., Bairoch, A., Bucher, P.: Prosite: a documented database using patterns and profiles as motif descriptors. Brief Bioinform. 3, 265–274 (2002)
12. Sacan, A., Ozturk, O., Ferhatosmanoglu, H., Wang, Y.: Lfm-pro: a tool for detecting significant local structural sites in proteins. Bioinformatics 23(6), 709–716 (2007)

A Clustering Based Hybrid System for Mass Spectrometry Data Analysis

Pengyi Yang[1] and Zili Zhang[1,2]

[1] Intelligent Software and Software Engineering Laboratory,
Faculty of Computer and Information Science,
Southwest University, Chongqing 400715, China
[2] School of Engineering and Information Technology, Deakin University,
Geelong, Victoria 3217, Australia
zzhang@deakin.edu.au

Abstract. Recently, much attention has been given to the mass spectrometry (MS) technology based disease classification, diagnosis, and protein-based biomarker identification. Similar to microarray based investigation, proteomic data generated by such kind of high-throughput experiments are often with high feature-to-sample ratio. Moreover, biological information and pattern are compounded with data noise, redundancy and outliers. Thus, the development of algorithms and procedures for the analysis and interpretation of such kind of data is of paramount importance. In this paper, we propose a hybrid system for analyzing such high dimensional data. The proposed method uses the k-mean clustering algorithm based feature extraction and selection procedure to bridge the filter selection and wrapper selection methods. The potential informative mass/charge (m/z) markers selected by filters are subject to the k-mean clustering algorithm for correlation and redundancy reduction, and a multi-objective Genetic Algorithm selector is then employed to identify discriminative m/z markers generated by k-mean clustering algorithm. Experimental results obtained by using the proposed method indicate that it is suitable for m/z biomarker selection and MS based sample classification.

1 Introduction

With the development of high-throughput proteomic technologies such as mass spectrometry (MS), we are now able to detect and discriminate disease patterns in complex mixtures of proteins derived from biological fluids such as serum, urine or nipple aspirate fluid [1,2]. The technologies commonly employed in such kind of differential studies are time-of-flight (TOF) spectroscopy with matrix-assisted or surface-enhanced laser desorption/ionization (SELDI) or SELDI-TOF [3,4]. Similar to microarray studies, SELDI-TOF datasets consist of tens of thousands of mass/charge (m/z) ratios per specimen [5,6]. Each m/z value of the spectrum approximately reflects the abundance of peptides of certain mass [7]. Despite of its great promise, the analysis of the data generated by such studies presented several major challenges. The challenges originate from the nature that

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 98–109, 2008.

SELDI-TOF datasets are often with large number of features and limited size of samples which are known as the curse-of-dimensionality and curse-of-dataset-sparsity problems [8]. To make the problem worse, SELDI-TOF data are often extremely noisy and redundant. Thus, how to select a subset of m/z biomarkers that not only can yield low sample misclassification rate but also have true biological importance are of great value.

Generally, feature selection algorithms can be categorized into three groups, namely, filter, wrapper and embedded.

With filter approaches, the feature subsets are selected with certain kind of evaluation criterion such as Mutual Information [9], t-statistic [10], χ^2-statistic [11] and Information Gain [12]. Although, filter selection methods are relatively computational efficient, they totally ignore the effects of the selected feature subset on the performance of the inductive algorithm [13]. More importantly, features selected with filter approaches are often highly correlated [14]. Therefore, redundancy and data noise are introduced, leading to the decrease of the classification accuracy while increasing the computational burden. Wrapper method get its name because the inductive algorithm is used or "wrapped" as the feature evaluation tool in the selection process. Classical wrapper methods often utilize forward selection and backward elimination to search feature sub-space, while advanced types of wrappers introduce the use of Evolution Strategy (ES) [15] and Genetic Algorithm (GA) [5,16,17]. Although wrapper methods often produce higher sample classification accuracy than filter methods, they are extremely computational intensive compared with filters. Overfitting is another problem of applying wrapper methods to high feature-to-sample ratio dataset analysis. The third group of selection methods are embedded approaches, which use the inductive algorithm itself as the feature selector and classifier. Examples are ID3 [18] and C4.5 [19]. The drawback of such kind of feature selection methods is that they are often greedy search based algorithms [20], using only the top ranked feature to perform sample classification in each step while an alternative split may perform better.

Since each type of feature selection method has its advantage and weakness, hybrid systems are often preferred for robustness and efficiency in feature selection application [6,21,22,23,24,25]. In [14], Jaeger et al. suggested that in microarray data analysis genes with high correlations are potentially belong to the same biological pathway. Therefore, if certain pathway has the main influence, the gene selection results may be dominated by such pathway, while other informative pathways will be totally ignored. This is especially phenomenal when one performs aggressive feature reduction with filter based methods which often consider each feature separately. To include information from other disease related pathways, several feature extraction methods have been proposed [22,24,25]. In [25], a k-mean clustering procedure is conducted to cluster the genes with similar expression pattern into groups. Then the mean expression level of a group of genes is calculated and used as the "prototype gene" for the later learning and classification process. However, a disadvantage of this method is that the "prototype gene" is a transformed feature vector which does not bear true biological

meaning. In [22], 50-100 genes from a microarray dataset are firstly pre-filtered by filter algorithms such as ReliefF, Information Gain and χ^2-statistic, and then hierarchically clustered. A representative gene which is most similar to the mean expression of its belonging cluster is then selected for later sample classification purpose. While this method does hold promise in identifying biologically important biomarkers, the size of the pre-filtered genes (50-100) potentially confined its power to include as much useful pathway information as possible. As for [24], the gene ranking and gene clustering processes are conducted independently. The final gene sets are then selected by using gene ranking and clustering information collaboratively. One drawback of this process is that the number of selected genes is still too large for any biological validation.

Similar to gene expression studies, when analyzing SELDI-TOF datasets, it's reasonable to assume that high correlation of m/z markers are the indication that they may belong to the same protein or proteins in the same pathway. The rationale of this argument is based on the central dogma of biology that proteins are the functional products of various mRNAs which are produced by their corresponding genes. Therefore, if the resulting classifier is created by several m/z markers with high correlation, the classifier will gain not much extra information than using just one representative m/z marker in this correlated group. In this study, we propose a k-mean clustering based biomarker extraction and selection method to bridge filter based and wrapper based feature selection algorithms. The advantages of this hybrid system are as follows:

- Filter based algorithm is employed to speed up the feature selection process by pre-filtering the potential disease related m/z markers. Therefore the total computation time is shortened than using wrapper based algorithm directly.
- The potential disease related m/z markers selected by filters are subject to the k-mean clustering algorithm for correlation, redundancy and data noise reduction. This procedure generates an information enriched and redundancy reduced dataset, which is crucial in creating accurate classification model.
- With above dimensional reduction, data cleansing and information extraction processes, the wrapper algorithm can be easily applied to identify a minimum m/z marker set, while also create accurate classification model.

We applied the proposed feature selection strategies to the analysis of two SELDI-TOF datasets and the experimental results are encouraging.

This paper is organized as follows: An overview of the proposed system is given in Section 2. Section 3 details the experiment designs while Section 4 provides the experimental results. Section 5 concludes the paper.

2 System Overview

The proposed system can be sequentially divided into following five steps:

- Firstly, a filter based feature selection method is conducted to pre-filter the potential biomarkers, by selecting the top 2000 m/z biomarkers with relatively high differential power.

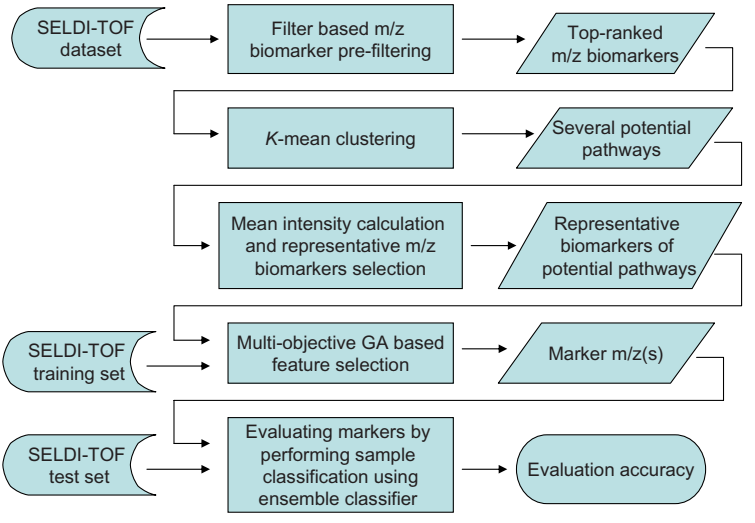


Fig. 1. The work flow of the proposed system for m/z marker selection and evaluation

- After the pre-filtering process, k-mean clustering is conducted on the resulting feature set. Ideally, each cluster corresponds roughly to a biological pathway.
- The mean intensity pattern of each cluster is calculated (also known as feature extraction) and an m/z marker which has the most similar intensity pattern to the mean intensity pattern is then selected as the representative m/z marker of this cluster.
- A multi-objective GA based wrapper selector is employed to further minimize feature redundancy by identifying informative pathway representatives and discard the uninformative ones.
- Lastly, an ensemble classifier integrated by majority voting is utilized to evaluate the selected m/z markers by performing sample classification.

Figure 1 visualizes the entire system work flow.

3 Methods

In this section we give a short description of the SELDI-TOF datasets used in the experiment and detail the design of each step.

3.1 Dataset

The SELDI-TOF MS datasets generated from prostate cancer analysis [3] and from ovarian cancer analysis [4] are applied to evaluate the proposed system.

The first dataset named "Prostate dataset", consists of 322 serum samples which are categorized into four classes. The first class contains 190 serum samples

which have been diagnosed as benign prostate hyperplasia with serum prostate-specific antigen (PSA) level greater than or equal to 4 ng/mL. The second class has 63 serum diagnosed as no evidence of disease with serum PSA level less than 1 ng/mL. The third class contains 26 serum samples diagnosed as prostate cancer with serum PSA level between 4 and 10 ng/mL. The last 43 serum samples were categorized as the fourth class with serum PSA level greater than 10 ng/mL.

The second dataset is a binary dataset, which contains only two classes referred as "Cancer" and "Normal". We named this dataset "Ovarian dataset". It includes 253 samples which can be divided into 91 normal samples and 162 ovarian cancer samples. Finally, the total m/z number of the dataset is 15154. Both datasets were split into training set for feature selection and test set for evaluation in our experiment. Table 1 summarizes the datasets and the partitions.

Table 1. SELDI-TOF MS datasets used in the experiment

Prostate dataset		**training**	**test**	**Ovarian dataset**		**training**	**test**
benign:	190	95	95	normal:	91	46	45
no evidence:	63	32	31				
cancer(4-10):	26	13	13	cancer:	162	81	81
cancer(10-):	43	22	21				

3.2 Pre-filtering

Most SELDI-TOF datasets contain several tens of thousands of m/z features, but only a small portion of these markers are trait associated [8,26]. By preforming a filter based pre-selection, we can eliminate the unrelated markers which may skew the final selection results. At the same time, the computation burden is also greatly decreased. However, the main concern is that the reduction should be carried out without sacrificing any useful information. In this study, we used two types of filter algorithms, namely, χ^2-statistic and Information Gain for the pre-filtering purpose. A safe number of m/z markers used in our experiment is 2000, which is large enough to capture most differential markers from various pathways while also suitable for k-mean clustering algorithm to work with.

3.3 *k*-Mean Clustering

k-mean clustering is an iterative algorithm. It groups the similar elements into a cluster while also increases the dissimilarity among different clusters by using a given definition of similarity and cluster mean. One major challenge of applying k-mean clustering algorithm is that the number of the clusters (k) must be determined before conducting the clustering process [22,24]. Yet, previous study [24] illustrated that the change of the k value (from 100 to 220) had quite limited impact on the classification results with different size of feature sets.

In this work, we carried out the k-mean clustering on the pre-filtered 2000 m/z markers and group them into 50 clusters. By doing so, markers with high

correlations are put into the same blocks for later feature extraction and representative marker selection. However, since k-mean clustering is stochastic, in our experiment we found that a different initial partition can result different clustering outcomes. To include as many potential pathways as possible while also avoiding the clustering results been affected by certain initialization, we repeated the k-mean clustering on the pre-filtering set 5 times with different initialization, producing five 50-cluster sets (a total of 250 clusters) for later process.

3.4 Cluster Feature Extraction and Representative Selection

Followed by k-mean clustering, we extract the mean intensity pattern of each cluster by averaging each m/z intensity value within the same cluster. After obtaining the mean intensity pattern of each cluster, a representative m/z for each cluster is then selected by comparing the similarity of mean intensity pattern of the cluster and the individual m/z markers and choosing the m/z with the minimum difference. The difference is defined as follow:

$$difference = \sum_{i=1}^{n} (\mathbf{x}_i - \overline{\mathbf{x}}_i)^2 \tag{1}$$

where n is the total number of samples, while $\overline{\mathbf{x}}_i$ is the mean intensity values of m/z markers of the ith sample within a cluster. With above extraction and selection process, our method selects one representative marker per cluster. One may ask that whether one representative m/z marker of a cluster is sufficient. In [24], Cai et al. evaluated using more than one representative per cluster to form the resulting feature subset, their experimental results demonstrated that one representative per cluster actually outperforms other choices (from 2 to 5).

After performing above procedures on all five k-mean clustered datasets with different initial partition, the selected representatives were then combined to form the clustering processed set for later wrapper based selection.

3.5 Multi-objective GA Based Feature Selection

It is important to notice that not all biological pathway information in the dataset are related to the disease or the biological trait of interest. Thus, those unrelated pathway representatives are redundant features in classification. Including these redundant features will increase the computational expenses while also compounds the identification of disease related biomarkers. Therefore, a multi-objective GA based feature selection step is employed to further minimize the m/z marker size by only selecting those highly discriminative representatives and their combinations.

The detail of the multi-objective GA based ensemble algorithm is described in [23]. Basically, this hybrid algorithm utilizes a multi-objective GA as the feature space searching engine while an ensemble classifier is used as the feature subsets evaluator to evaluate feature combination produced by multi-objective GA. Here the ensemble classifier is the combination of five individual classifiers

(decision tree, logistic regression, support vector machine, naive bayes and k-nearest neighbor) integrated with majority voting strategy.

The fitness function of the multi-objective GA is defined as the average sample classification accuracy and the consensus sample classification accuracy:

$$fitness_1(s) = \frac{\sum_{j=1}^{n} accuracy_j(s)}{n} \tag{2}$$

$$fitness_2(s) = consensus(s) \tag{3}$$

$$fitness(s) = \frac{fitness_1(s) + fitness_2(s)}{2} \tag{4}$$

where $accuracy_j(s)$ specify the classification accuracy of the jth classifier upon the sth feature set, while $consensus(s)$ specify the classification accuracy using majority voting with the five classifier committee upon the sth feature set.

Table 2 provides the details of the GA parameters used in the experiment, and the training portion of the datasets were used to perform the m/z marker selection.

Table 2. Genetic Algorithm Parameter Settings

Parameter	Value
Genetic Algorithm	Multi-Objective
Population Size	100
Selector	Binary Tournament Selection
Crossover	Single Point (0.7)
Mutation	Multi-Point (0.05 & 0.25)
Termination Condition	50th generation

3.6 Subset Evaluation

After the m/z marker selection process, the selected m/z markers are then evaluated by the ensemble classifier itself with the test portion of the datasets. Three repeated runs of 10-fold stratified cross-validation with random partition are applied to the test datasets, and the sample classification accuracy is calculated by averaging the results. It is worth noting that the feature selection and evaluation processes are accomplished using multiple classifiers. Therefore, they are less subject to certain inductive algorithm and have better generalization.

For the comparison purpose, we provide a baseline by using filter selected m/z markers as the inputs of the ensemble classifier directly. Also, we compare the evaluation accuracy of the m/z markers selected by applying multi-objective GA based algorithm directly to the 2000 pre-filtered candidate markers with the proposed method (which applying multi-objective GA based algorithm after the k-mean clustering process). With the consideration of the stochastic nature of

GA, each GA based method is conducted with 5 independent runs. We report the mean results of the 5 runs and give the standard deviation in the form of $mean \pm \sigma$.

4 Results

The first question should be asked is how many m/z markers we should select as the final feature set for sample classification. To answer this question we utilized the proposed methods (using both χ^2-statistic and Information Gain) to test the marker size varying from 5 to 40 with a step of 5, using Prostate dataset. Figure 2 depicts the test results. It's evident that a size of 20-25 m/z marker set is sufficient. Therefore, in the following comparison experiments, we evaluate the m/z combinations with size varying from 5 to 25 with a step of 5 for both SELDI-TOF datasets.

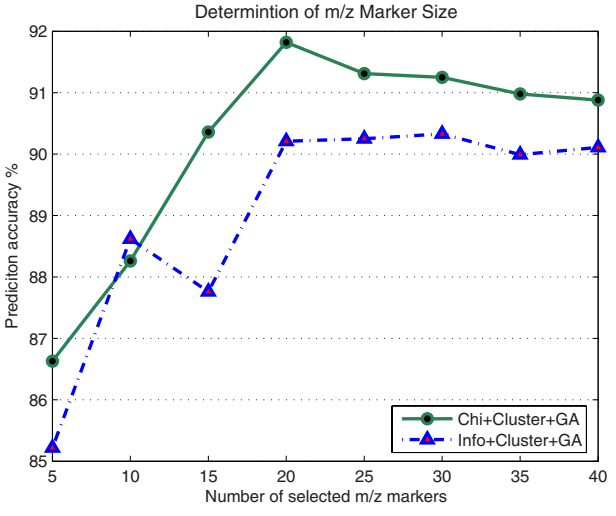


Fig. 2. To determine the size of the m/z markers for sample classification, we test the marker size varying from 5 to 40 with a step of 5, using Prostate dataset

As aforementioned, each of the two feature filter methods was used to rank the m/z markers, respectively. Then we compared the evaluation accuracy of the following three different processes:

1. Using filter ranked top m/z marker combinations (5, 10, 15, 20, and 25) for subset evaluation and sample classification.
2. Using the top 2000 m/z markers ranked by a filter as a pre-filtered marker pool, and applying multi-objective GA based algorithm to select m/z marker combinations (5, 10, 15, 20, and 25) for subset evaluation and sample classification.

3. Applying the proposed process. Using the top 2000 m/z markers ranked by a filter as a pre-filtered marker pool, and employing k-mean clustering and representative selection process to reduce correlation, redundancy and noise. Then utilizing multi-objective GA based algorithm to select m/z marker combinations (5, 10, 15, 20, and 25) for subset evaluation and sample classification.

The subset evaluation process was carried out as described in Section 3.6. Table 3 provides detailed evaluation accuracy of each method with m/z combination size of 5, 10, 15, 20, and 25. As can be seen, the evaluation accuracy of using solely filters of χ^2-statistic and Information Gain from 5 to 25 features with both MS datasets does not differ significantly. For prostate dataset, the average of 80.08 for χ^2-statistic and the average of 82.90 for Information Gain are obtained. As for ovarian dataset, the average results are 95.20 for χ^2-statistic and 95.17 for Information Gain. This is consistent with the assumption that the filter selected top markers is strongly correlated. When used to construct classifier, such a redundant feature set does not provides much extra information than using just a subset of it. Based on the experiment results, it is also readily noticed that GA based methods achieved higher classification accuracy than using filter ranked features directly. This evidence suggests that beside several top ranked features more information for sample classification do contained in the rest of the feature pool and GA based selection scheme be able to identify these "important" features. When comparing the results of applying multi-objective GA method directly with the 2000 pre-filtered m/z features and the results of applying multi-objective GA method with k-mean clustering processed datasets, we found that the classification accuracy of the later is generally about 2-3 percent higher with few exceptions. These results indicate that the k-mean clustering based feature correlation and redundancy reduction process can further improve the final feature selection and sample classification outcomes.

Table 3. Evaluation accuracy of each method using test datasets

Prostate Dataset						
m/z Size	χ^2	χ^2+GA	χ^2+Cluster+GA	Info	Info+GA	Info+Cluster+GA
5	80.88	83.05 ± 3.7	86.63 ± 2.2	83.69	83.48 ± 3.3	85.22 ± 3.1
10	79.28	87.79 ± 1.6	88.26 ± 1.4	82.54	86.09 ± 2.7	88.62 ± 1.7
15	81.06	88.63 ± 1.3	90.36 ± 1.7	81.88	86.46 ± 3.4	87.76 ± 2.9
20	79.85	90.58 ± 1.5	91.82 ± 1.8	83.13	87.97 ± 1.5	90.21 ± 1.5
25	79.34	89.46 ± 1.0	91.31 ± 1.9	83.27	88.31 ± 2.0	90.25 ± 1.2
Ovarian Dataset						
m/z Size	χ^2	χ^2+GA	χ^2+Cluster+GA	Info	Info+GA	Info+Cluster+GA
5	94.39	96.88 ± 1.4	97.66 ± 1.1	94.54	97.13 ± 1.3	97.96 ± 1.1
10	95.02	97.08 ± 0.9	98.58 ± 0.8	95.49	97.27 ± 1.4	98.88 ± 0.9
15	95.94	97.22 ± 0.8	98.24 ± 0.8	94.86	98.63 ± 0.6	98.47 ± 0.3
20	95.79	96.48 ± 1.0	98.82 ± 0.9	95.46	97.26 ± 0.8	98.48 ± 0.5
25	94.86	96.39 ± 1.3	98.12 ± 1.3	95.48	98.42 ± 0.8	98.32 ± 0.9

Table 4. Top 5 frequently selected m/z biomarkers of the proposed system, using the Prostate and Ovarian datasets, respectively. Each m/z marker is ranked by selection frequency, and the overlapped ones are shown in bold.

	χ^2-Statistic		Information Gain	
Rank No.	m/z *id*	selection freq.	m/z *id*	selection freq.
1	**0.054651894**	0.96	**125.2173**	0.92
2	**125.2173**	0.76	**0.054651894**	0.80
3	497.9286	0.64	**478.95419**	0.72
4	**271.33373**	0.60	**271.33373**	0.72
5	**478.54579**	0.56	362.11416	0.68
1	**MZ436.63379**	0.88	**MZ245.53704**	0.94
2	**MZ245.53704**	0.82	**MZ436.63379**	0.78
3	MZ4003.6449	0.74	**MZ6803.0344**	0.72
4	MZ28.900817	0.68	MZ7898.4503	0.56
5	**MZ6803.0344**	0.62	MZ557.06335	0.56

Table 5. The classification results of prostate dataset (test set) using top 5 m/z markers selected by the proposed method with χ^2-statistic and Information Gain, respectively. Correctly classified samples are in bold.

		χ^2-Statistic				Information Gain			
Class	Samples	B	NE	C4-10	C10-	B	NE	C4-10	C10-
benign (B)	95	**93**	0	2	0	**94**	0	1	0
no evidence (NE)	31	2	**28**	0	1	1	**27**	0	3
cancer(4-10) (C4-10)	13	2	0	**9**	2	3	0	**8**	2
cancer(10-) (C10-)	21	1	0	3	**17**	1	0	3	**17**

Table 4 lists the top 5 most frequently selected m/z markers using the proposed method with χ^2-statistic and Information Gain, respectively. There are several overlapped biomarkers (marked with bold type) in the two independent results despite the use of two different pre-filtering algorithms, indicating the potential disease association of them. For prostate dataset, using these top 5 m/z markers selected with χ^2-statistic filtering, the evaluation accuracy with the test set is 91.88, while using the top 5 m/z markers selected with Information Gain, the evaluation accuracy with the test set is 91.25. As for ovarian dataset, the classification accuracy using the top 5 m/z is 98.40 with χ^2-statistic filtered dataset and 97.97 with Information Gain filtered dataset. Table 5 provides the confusion matrix of the prostate data classification results.

5 Discussion and Conclusion

In this paper, we proposed a k-mean clustering based feature extraction and selection approach for the analysis of mass spectrometry dataset. The proposed method sequentially combines pre-filtering, k-mean clustering based correlation

reduction and GA based wrapper selection processes. The clustering process serves as the bridge between filter based pre-selection and final wrapper based feature selection. It decreases the dimensionality of the pre-filtered dataset while also reduces the correlation of the m/z markers, outputting a nearly noise-free and information enriched dataset.

The experimental results suggest that the clustering based correlation reduction process can improve the sample classification accuracy and the system's power in disease related biomarker selection. It also demonstrates the potential use of this hybrid system in disease related biological pathway identification.

References

1. Morris, J.S., Coombes, K.R., Koomen, J., Baggerly, K.A., Kobayashi, R.: Feature extraction and quantification for mass spectrometry in biomedical applications using the mean spectrum. Bioinformatics 21(9), 1764–1775 (2005)
2. Petricoin, E.F., Liotta, L.A.: SELDI-TOF-based serum proteomic pattern diagnostics for early detection of cencer. Curr. Opin. Biotechnol. 15, 24–30 (2004)
3. Petricoin, E.F., Ornstein, D.K., Paweletz, C.P., Ardekani, A.M., Hackett, P.S., Hitt, B.A., Velassco, A., Trucco, C., Wiegand, L., Wood, K., Simone, C.B., Levine, P.J., Linehan, W.M., Emmert-Buck, M.R., Steinberg, S.M., Kohn, E.C., Liotta, L.A.: Serum Proteomic Patterns for Detection of Prostate Cancer. Journal of the National Cancer Institute 94(20), 1576–1578 (2002)
4. Petricoin, E.F., Ardekani, A.M., Hitt, B.A., Levine, P.J., Fusaro, V.A., SteinBerg, S.M., Mills, G.B., Simone, C., Fishman, D.A., Kohn, E.C., Liotta, L.A.: Use of proteomic patterns in serum to identify ovarian cancer. The Lancet 359, 572–577 (2002)
5. Li, L., Umbach, D.M., Terry, P., Taylor, J.A.: Application of the GA/KNN method to SELDI proteomics data. Bioinformatics 20(10), 1638–1640 (2004)
6. Yu, J.S., Ongarello, S., Fiedler, R., Chen, X.W., Toffolo, G., Cobelli, C., Trajanoski, Z.: Ovarian cancer identification based on dimensionality reduction for high-throughput mass spectrometry data. Bioinformatics 21(10), 2200–2209 (2005)
7. Boguski, M.S., McIntosh, M.W.: Biomedical informatics for proteomics. Nature 422, 233–236 (2003)
8. Somorjai, R.L., Dolenko, B., Baumgartner, R.: Class prediction and discovery using gene microarray and protenomics mass spectroscopy data: curses, caveats, cautions. Bioinformatics 19(12), 1484–1491 (2003)
9. Ding, C., Peng, H.: Minimum Redundancy Feature Selection From Microarray Gene Expression Data. Journal of Bioinformatics and Computational Biology 3(2), 185–205 (2005)
10. Golub, T.R., Tamayo, D.K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J.P., Coller, H., Loh, M.L., Downing, J.R., Caligiuri, M.A., Boomfield, C.D., Lander, E.S.: Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression Monitoring. Science 286, 531–537 (1999)
11. Liu, H., Li, J., Wang, L.: A Comparative Study on Feature Selection and Classification Methods Using Gene Expression Profiles and Proteomic Patterns. Genome Informatics 13, 51–60 (2002)
12. Su, Y., Murali, T., Pavlovic, V., Schaffer, M., Kasif, S.: RankGene: Identification of Diagnostic Genes Based on Expression Data. Bioinformatics 19(12), 1578–1579 (2003)

13. Kohavi, R., John, G.: Wrapper for feature subset selection. Artificial Intelligence 97(1-2), 273–324 (1997)
14. Jaeger, J., Sengupta, R., Ruzzo, W.L.: Improved Gene Selection for Clssification of Microarrays. Pac. Symp. Biocomput., 53–64 (2003)
15. Jirapech-Umpai, T., Aitken, S.: Feature Selection and Classification for Microarray Data Analysis: Evolutionary Methods for Identifying Predictive Genes. BMC Bioinformatics 6, 146 (2005)
16. Yang, P.Y., Zhang, Z.L.: Hybrid Methods to Select Informative Gene Sets in Microarray Data Classification. In: Orgun, M.A., Thornton, J. (eds.) AI 2007. LNCS (LNAI), vol. 4830, pp. 811–815. Springer, Heidelberg (2007)
17. Yang, P.Y., Zhang, Z.L.: A Hybrid Approach to Selecting Susceptible Single Nucleotide Polymorphisms for Complex Disease Analysis. In: Proceedings of BMEI 2008, pp. 214–218. IEEE, Los Alamitos (2008)
18. Quinlan, J.R.: Learning efficient classification procedures and their applicaiton to chess and games. In: Machine Learning: An Artificial Intelligence Approach. Morgan Kaufmann, San Mateo (1983)
19. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann, San Mateo (1993)
20. Blum, A.L., Langley, P.: Selection of relevent Features and Examples in Machine Learning. Artificial Intelligence 97(1-2), 245–271 (1997)
21. Geurts, P., Fillet, M., de Seny, D., Meuwis, M.A., Malaise, M., Merville, M.P., Wehenkel, L.: Proteomic mass spectra classifcation using decision tree based ensemble methods. Bioinformatics 21, 3138–3145 (2005)
22. Wang, Y., Makedon, F., Ford, J., Pearlman, J.: HykGene: A Hybrid Approach for Selecting Marker Genes for Phenotype Classification using Microarray Gene Expression Data. Bioinformatics 21(8), 1530–1537 (2005)
23. Zhang, Z.L., Yang, P.Y.: An Ensemble of Classifier with Genetic Algorithm Based Feature Selection. (accepted by IEEE Intelligent Informatics Bulletin)
24. Cai, Z., Goebel, R., Salavatipour, M.R., Lin, G.: Selecting Dissimilar Genes for Multi-Class Classification, an Application in Cancer Subtyping. BMC Bioinformatics 8, 206 (2007)
25. Hanczar, B., Courtine, M., Benis, A., Hennegar, C., Clement, K., Zucker, J.-D.: Improving classification of microarray data using prototype-based feature selection. SIGKDD Explorations 5, 23–30 (2003)
26. Saeys, Y., Inza, I., Larrañaga, P.: A review of feature selection techniques in bioinformatics. Bioinformatics 23(19), 2507–2517 (2007)

A Modified Markov Clustering Approach for Protein Sequence Clustering

Lehel Medvés[1], László Szilágyi[1,2], and Sándor M. Szilágyi[1]

[1] Sapientia - Hungarian Science University of Transylvania,
Faculty of Technical and Human Science, Târgu-Mureş, Romania
medveslehel@yahoo.com, {lalo,szs}@ms.sapientia.ro
[2] Budapest University of Technology and Economics, Department of Control
Engineering and Information Technology, Budapest, Hungary

Abstract. In this paper we propose a modified Markov clustering algorithm for efficient clustering of large protein sequence databases, based on previously evaluated sequence similarity criteria. The proposed alteration consists in an exponentially decreasing inflation rate, which aims at helping the quick creation of the hard structure of clusters by using a strong inflation in the beginning, and at producing fine partitions with a weaker inflation thereafter. The algorithm, which was tested and validated using the whole SCOP95 database, or randomly selected 10-50% sections, generally converges within 12-14 iteration cycles and provides clusters of high quality. Furthermore, a novel generalized formula is given for the inflation operation, and an efficient matrix symmetrization technique is presented, in order to improve the partition quality with relatively low amount of extra computations. A large graph layout technique is also employed for the efficient visualization of the obtained clusters.

Keywords: Markov clustering, protein sequence clustering, sparse matrix, large graph layout, SCOP95 database.

1 Introduction

One of the main goals of functional genomics is to establish protein families in large databases. Successful classification of protein families can have significant contributions to the delineation of functional diversity of homologous proteins, and can provide valuable evolutionary insights as well [9].

By definition, protein families represent groups of molecules showing relevant sequence similarity [4]. Members of such protein families may serve similar or identical biological purposes [12]. Identifying these families is generally performed by clustering algorithms, which are supported by similarities and/or dissimilarities, previously computed between all pairs of protein sequences.

Performing an accurate clustering results in such protein families, whose members are related by a common evolutionary history [10]. If this condition holds, well established properties of some proteins in the family may be reliably transferred to other members whose functions are not well known [11]. Several protein clustering methods are currently available in the literature [7,14]. One of

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 110–120, 2008.

the greatest obstacle for them represents the multi-domain structures of many protein families [5].

TRIBE-MCL is an efficient protein sequence clustering method proposed by Enright et al. [9], based on Markov chain theory [6,7]. The authors assigned a graph structure to the protein database such a way that each protein has a corresponding node, while edge weights in the graph represent a priori computed similarity values, obtained via BLAST search methods [2]. Clusters were then obtained by alternately applying two operations to the similarity matrix: inflation and expansion. The former represents a task, which re-evaluates the values within columns of the matrix by raising higher probabilities and suppressing the small ones, while the latter aims at favoring longer walks along the graph, which is obtained via matrix squaring.

In this paper we propose a modification of the TRIBE-MCL algorithm, in order to enhance its accuracy and improve its time complexity. This is achieved by introducing a time-varying inflation rate and thus forcing the algorithm to apply a stronger inflation in the first iteration when the hard structure of the clusters is established, and reduce inflation strength for fine tuning the cluster shape in later iterations.

The remainder of this paper is structured as follows: Section 2 takes into account the functional details of the TRIBE-MCL algorithm and presents the proposed modifications. Section 3 presents our own considerations upon large graph layout techniques. Section 4 evaluates and discusses the efficiency and accuracy of the proposed method. Section 5 presents the conclusions and gives some hints for further research.

2 The Proposed Markov Clustering Approach

2.1 The TRIBE-MCL Algorithm

TRIBE-MCL is an iterative algorithm, which operates on a directional graph. The nodes of the graph represent the protein sequences we wish to cluster, while edges show the similarity between pairs of protein sequences. The edge lengths are stored in a so called similarity matrix. Theoretically, the initial edge lengths can be computed using any sequence alignment method. However, the convergence speed will depend on the initial similarities.

In most of the cases, this initial similarity matrix is not symmetrical. This may be treated as a problem or not. If one only wishes to cluster the sequences, that is, to find certain groups of proteins, which show high similarity within the cluster and low similarity between different clusters, then using symmetrical similarity matrix is recommendable. Asymmetrical similarity values, however, may reveal the direction of evolution among the proteins situated within a given cluster. Consequently, making the similarity matrix symmetrical in every iteration is an extra step, whose benefits and costs will be tested in the followings.

The TRIBE-MCL algorithm treats the similarity matrix as a stochastic matrix [8]. In such a matrix, probability or possibility values are stored: in general,

S_{ij} represents the possibility, that protein i becomes protein j during an evolutionary step. A decision has to be made at the beginning, whether S is treated as a column stochastic matrix or a row stochastic matrix. The difference is significant: if S is a column stochastic matrix, the columns are normalized in every iteration and thus they become probability values of past evolutionary steps: for example S_{ij} shows what is the probability that protein j was i before the latest evolutionary step. On the other hand, rows of a column stochastic matrix are not normalized, so values in row j show the possible outcomes of a next evolutionary step, and their likelihood values, which are not probabilities as their sum is not 1. In this paper we chose to treat the similarity matrix as a column stochastic matrix.

The TRIBE-MCL algorithm consists of two main operations, namely the inflation and expansion, which are repeated alternately until a convergence is reached, that is, clusters become stable. Inflation has the main goal to favor more likely direct walks along the graph in the detriment of less likely walks, while expansion reveals possible longer walks along the graph.

2.2 The Inflation Operation

The inflation operation has the main goal to modify the similarity values within the columns of the similarity matrix such a way, that differences gain some emphasis. In other words, inflation favors more probable walks over less probable walks along the protein graph [9]. Literature recommends using the following inflation operation:

$$S_{ik}^{(n+1)} = \frac{\left(S_{ik}^{(n)}\right)^r}{\sum_{j=1}^{N}\left(S_{jk}^{(n)}\right)^r} \quad , \tag{1}$$

where r represents the inflation rate, which controls the strength of inflation. The larger the inflation rate, the more favored will be the high similarities.

Besides the inflation itself, an intentional side effect is the normalization of the columns: whatever the similarity values were before inflation (except for a zero column, which is unlikely to occur), the column will sum up at one after the operation.

If we examine the evolution of the number of clusters of different sizes (one such representation can be seen in Fig. 2(left)), we can remark, that TRIBE-MCL has two stages of its runtime: it needs some iterations until the changes influence the number of clusters significantly. The end of this first stage is shown by a significant maximum in the numbers or small clusters (not singletons). The number of iterations in this first stage strongly depends on the inflation rate: in case of $r = 1.1$, this first stage may need 7-8 iterations, while in case of $r = 1.5$, 3-4 iterations suffice. During the second stage, the number of small clusters decreases and finally stabilizes after 8-12 iterations.

In our opinion, the inflation rate has to be chosen such a way, that it is large enough in the first stage, so that the formation of cluster begins quickly and thus

the first stage doesn't last too long. On the other hand, in the second stage the inflation rate has to be small enough, to obtain high-quality clusters. In order to deal with both these requirements, in this paper we propose the usage of a variable inflation rate, given by the equation:

$$r^{(n)} = 1 + r_0 \times \exp\left(-\frac{n}{\tau}\right) \quad , \tag{2}$$

where $1 + r_0$ represents the initial inflation rate, n is the ordinal number of the current iteration, and τ is a time constant. In this paper we use: $r_0 = 2$ and $\tau = 10$, which gives the inflation rate the variation shown in Fig. 3(right).

This inflation operation could be generalized in the following manner: let us define a continuous function $I : [0, +\infty) \to [0, +\infty)$, $I(0) = 0$, $I'(x) > 0$, $I''(x) > 0\ \forall x > 0$. It can be proved, that the generalized inflation, defined as:

$$S_{ik}^{(n+1)} = \frac{I(S_{ik}^{(n)})}{\sum_{j=1}^{N} I(S_{jk}^{(n)})} \quad , \tag{3}$$

where I was established according to the above mentioned conditions, favors high similarities over low ones. Nevertheless, this generalized formula give us a higher freedom to choose the inflation operation. Obviously, setting $I(x) = x^r$ returns us to (1).

2.3 The Expansion Operation

The expansion operation is associated with random walks of higher lengths along the graph, which may include several steps [9]. It is computed with the normal matrix squaring operation. It produces new probabilities with all pairs of nodes, where one node is the point of departure and the other is the destination. Obviously, we will get high probabilities for pairs of nodes situated within the same cluster, and low ones for nodes from different clusters.

Expansion needs two instances of the similarity matrix. Consequently this is the operation, which determines the amount of directly processable protein sequences, due to the memory limitations of the PC.

2.4 Matrix Symmetry

Even if similarity measures are initially symmetrized, this property gets lost after the first inflation, due to the nature of the operator, as it treats the similarities as a column stochastic matrix. If there is any reason (dictated by the biological scenario) for which symmetrical similarity matrix is required, the following extra processing step should be inserted in every iteration of the TRIBE-MCL.

1. For any $i, j = 1 \ldots n$, $i \neq j$, if $|S_{ij} - S_{ji}| > \varepsilon$, set $S_{ij}^{(\text{new})} = S_{ji}^{(\text{new})} = \sqrt{S_{ij} \times S_{ji}}$, where ε is a previously set small constant.
2. Normalize the columns of the stochastic matrix.
3. Repeat steps 1-2, until symmetry is reached with ε tolerance.

2.5 Implementation Issues

The amount of protein sequences involved in clustering is theoretically limited by the following relation:

$$2 \times N^2 \times d \leq M \ , \tag{4}$$

where d represents the memory required by one probability value (acceptable resolution requires 16 bits), and M is the available amount of memory. Considering 1GB storage space, this means a theoretical limit of $N < 16384$ proteins. Most protein databases contain even more data. When the necessary storage for two similarity matrices required by the expansion is not available, we propose using a sparse matrix representation. If we suppose three decimal representation of transition probabilities, a maximum number of 1000 values can be nonzero in each column, but practically their count will be less with at least an order of magnitude. So the sparse matrix representation, even if needs three times more space for a single probability value, can reduce the necessary memory, and can significantly increase the amount of simultaneously processable proteins.

2.6 Algorithm

The proposed algorithm can be summarized as follows:

1. Compute initial similarity matrix.
2. Inflate the similarity data according to Eq. (3).
3. Expand the similarity data via matrix squaring.
4. Symmetrize the similarity matrix if desired.
5. Repeat steps 2-4 until transition probabilities stabilize. Generally 10-15 iterations suffice.

3 Graph Visualization

By applying the proposed Markov clustering method to subsets of the SCOP95 database, we obtain a few large clusters among the small ones. In order to visualize the structure of such large clusters, we propose a modified version of the large graph layout (LGL) algorithm given in [1].

The subgraph obtained from the Markov clustering (from now on: subgraph), representing a large cluster, provides the input data for the proposed layout generating algorithm.

The proposed algorithm places the nodes within the setting gradually, and allows them to move according to some attractive and repulsive forces that interact among them (see Fig. 1). The magnitude of the attractive forces between two given nodes only depends on their similarity value, their relative position only influences the direction of the force. There is also a repulsive force between any two nodes, whose strength only depends on their physical distance. This force has the main goal to keep nodes distant from each other. Short distance implies extremely strong repulsive force, which loses its strength if distances grow, and at a given distance limit the repulsive force is extinguished.

These forces are not meant in physical terms, and the equation (8) that describes the movement of nodes doesn't correspond to physical laws either.

The computation of the graph is performed according to the following rules:

1. As a first step, a minimum spanning tree (MST) of the subgraph is generated, using as weights the dissimilarity values obtained as a negative power of the computed similarities. This MST will establish the order in which the nodes will be placed into the layout.
2. One of the nodes in the MST needs to be declared central node. This node could be arbitrarily chosen, however, we always choose the node showing largest similarities to its neighbors.
3. The central node is placed into the origin, and it's frozen in that position.
4. We look for the neighbors of the central node in the MST, and place them on a hypersphere having its center in the origin and its radius of unit length. We let these nodes move without leaving the surface of the hypersphere, according to the attractive and repulsive forces that interact among them. Finally these nodes are frozen in their stabilized positions.
5. We look for the neighbors of the nodes found in the previous step in the MST, and place them on a double-radius hypersphere according to the following formula:

$$\boldsymbol{v}_{\text{new node}} = \left(\frac{\boldsymbol{v}_{\Omega}}{||\boldsymbol{v}_{\Omega}||} + \frac{\boldsymbol{v}_{\text{parent}} - \boldsymbol{v}_{\text{grandparent}}}{||\boldsymbol{v}_{\text{parent}} - \boldsymbol{v}_{\text{grandparent}}||} \right) + \boldsymbol{v}_{\text{parent}} + \nu \ , \qquad (5)$$

where the notation $\boldsymbol{v}$ refers to position vector, Ω represents the set of nodes already present in the layout, and ν is a random noise vector, which assures that newly introduced nodes will not be placed into identical positions. The movement of nodes are governed by attractive and repulsive forces among them. The sum of forces that influence node n_k is given by

$$\boldsymbol{F}_k = \sum_{i \in \Omega_k} \left[\boldsymbol{F}_{i,k}^{(\mathrm{a})} + \boldsymbol{F}_{i,k}^{(\mathrm{r})} \right] \ . \qquad (6)$$

where $\boldsymbol{F}_{i,k}^{(\mathrm{a})}$ stands for the attractive force emerging from the similarity between proteins represented by nodes n_i and n_k, while $\boldsymbol{F}_{i,k}^{(\mathrm{r})}$ is the repulsive force born from the mutual positions of nodes n_i and n_k, and $\Omega_k = \Omega - \{k\}$. The forces influencing node n_k are computed as follows:

$$\boldsymbol{F}_k = \sum_{i \in \Omega_k} \left[\frac{(\boldsymbol{v}_i - \boldsymbol{v}_k)}{||\boldsymbol{v}_i - \boldsymbol{v}_k||} \times S(n_i, n_k) + (\boldsymbol{v}_i - \boldsymbol{v}_k) \times g\left(||\boldsymbol{v}_i - \boldsymbol{v}_k||\right) \right] \ , \qquad (7)$$

where $S(n_i, n_k) = S_{ik}$ is the similarity value provided by the Markov clustering, and $g(\cdot)$ represents the function that describes the behavior of repulsive forces: it is considered as an exponentially decreasing function that reaches the zero value at a given distance. The movement of node n_k is described by the equation

$$\boldsymbol{v}_k \leftarrow \boldsymbol{v}_k + \Delta t \times \boldsymbol{F}_k \ , \qquad (8)$$

where Δt is the considered time step. As a final computation step, the position vector v_k is brought back to its hypersphere of radius r_k:

$$\boldsymbol{v}_k \leftarrow \boldsymbol{v}_k \times \frac{r_k}{||\boldsymbol{v}_k||} \quad . \tag{9}$$

Nodes are frozen in the stable position they reach in several movement steps. A single movement step is depicted in Fig. 1.

6. Repeat the previous step until all nodes of the subgraph will be included into the layout.

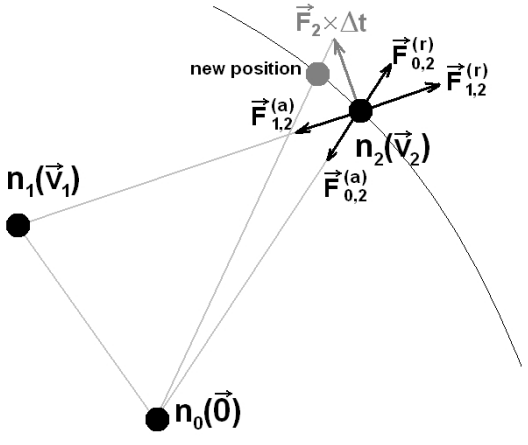


Fig. 1. Balance of forces of one node in case of a 3-node setting

4 Results and Discussion

The proposed modified TRIBE-MCL method was evaluated using the SCOP95 database [3,13], which contains protein sequences that show at most 95% similarity with each other.

The number of protein sequences in the database is quite large to handle using a PC. That's why, in some cases, we decided to randomly choose only a part (multiples of 10%) of the database to test the efficiency and accuracy of the clustering.

Tests revealed the efficiency of the proposed algorithm: the first stage, during which the hard structure is established, usually requires 3 or 4 iteration cycles. The second stage needs further 8-10 cycles to reach convergence.

Figures 2-4 show the results of the algorithm performed on 40% of the SCOP95 database. Figure 2 (left) shows the varying number of clusters of different sizes over 20 iterations.

The boundary between the running stages of the algorithm, indicated by the maxima, are clearly visible at iteration number 3. Figure 2 (right) indicates the total number of non-singleton cluster after each cycle. This latter image indicates

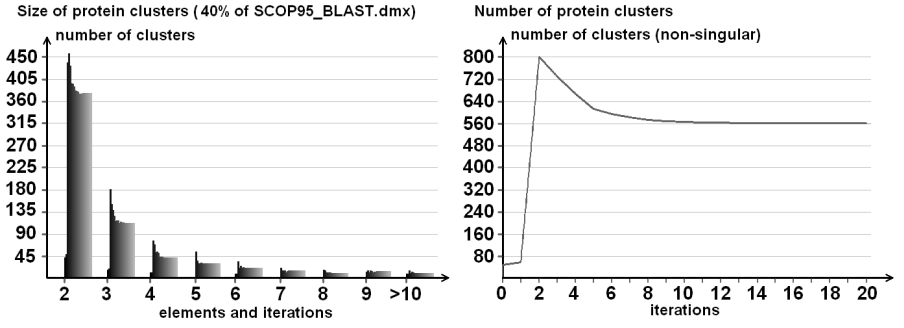


Fig. 2. (left) The evolution of number of different sized clusters, during 20 iteration cycles. The first stage needed four iterations, while the full convergence required an 8-cycle second stage; (right) Graphical representation of the number of non-singleton clusters vs. iteration index.

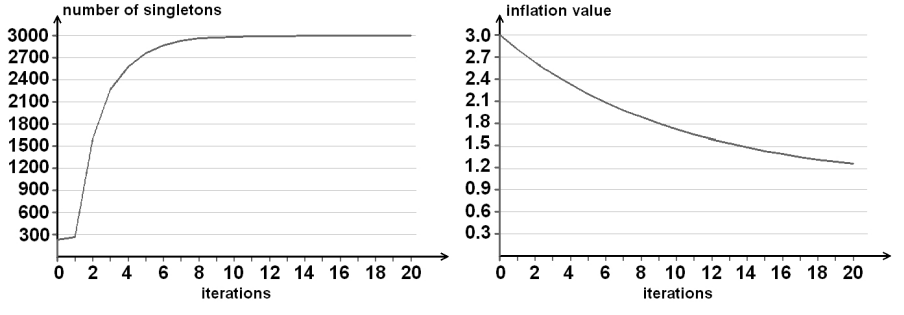


Fig. 3. (left) The first rising, then converging number of singletons; (right) The variable inflation rate, represented vs. iteration count

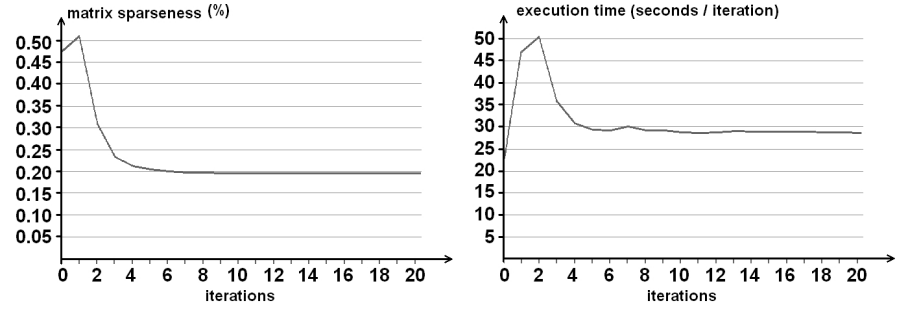


Fig. 4. The varying inner parameters of the algorithm: the sparseness of the similarity matrix (left) and the duration in time of each iteration, when the input data was randomly chosen 40% of SCOP95 database

Fig. 5. The best sequence alignment found in the largest cluster of randomly chosen 40% of the SCOP95 proteins

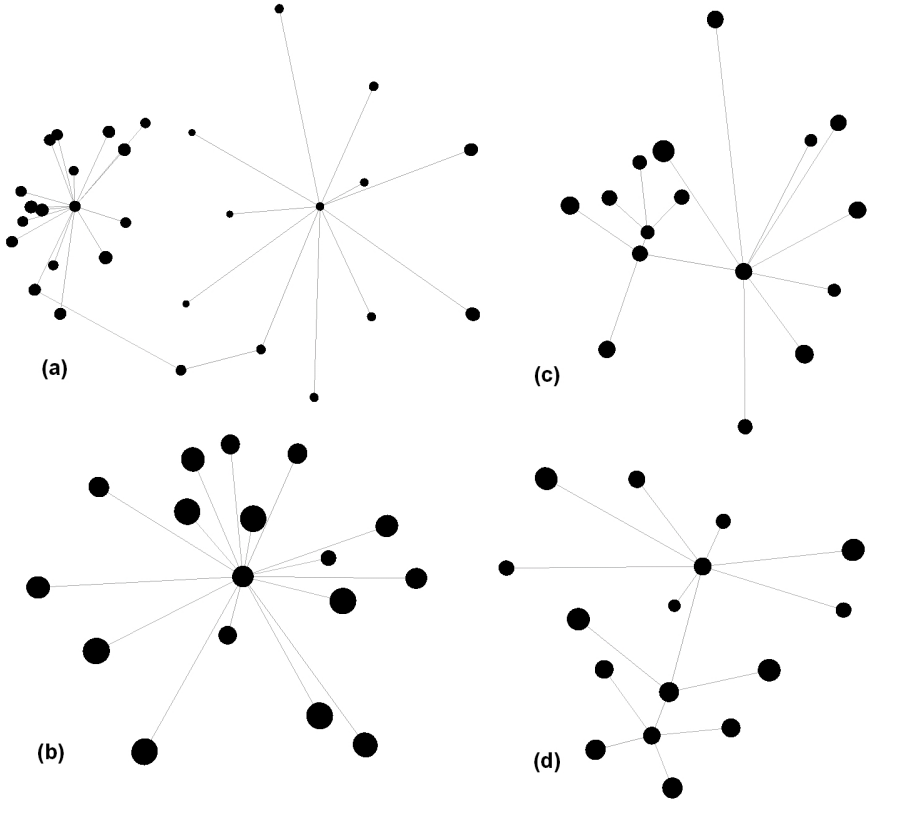


Fig. 6. 3D representation of a given cluster during the iterations of the modified TRIBE-MCL algorithm: (a) after 2 iterations, (b) after 5 iterations, (c) after 8 iterations, (d) after 10 iterations

iteration 14 to be the one, when convergence is established. Figure 3 (left) shows the variation of the number of detected isolated protein sequences in the input data. Figure 3 (right) is a graphical representation of the variable inflation rate, using the parameters proposed in the previous chapter.

Figure 4 shows the runtime parameters: on the left side we can see the sparseness of the similarity matrix, when we used four decimals resolution for similarity values. Sparseness values below 0.5% means that only one out of 200 probabilities is non-zero during the computations, so the number of simultaneously processable proteins can increase 8-10 times, if we turn to sparse matrix representation within the bounds of the the limited storage space. Figure 4 (right) indicates the time necessary to process each iteration with a PC having Athlon64 3200+ processor and 1GB RAM.

Figure 5 shows the alignment of those two protein sequences, which were found the most similar ones inside the largest obtained cluster.

Figure 6 presents the aspect of a given cluster at different stages of the Markov clustering. After two iterations, the graph still consists of a weak union of two clusters. After five iterations, the set of proteins that would finally belong to the cluster is mostly established (only one node will leave the cluster after this point). But the structure of the cluster still undergoes several slight changes, as shown in the representations of later stages.

Finally, Fig. 7 shows a 2D graph representation of the largest cluster found, produced with the proposed large graph layout algorithm.

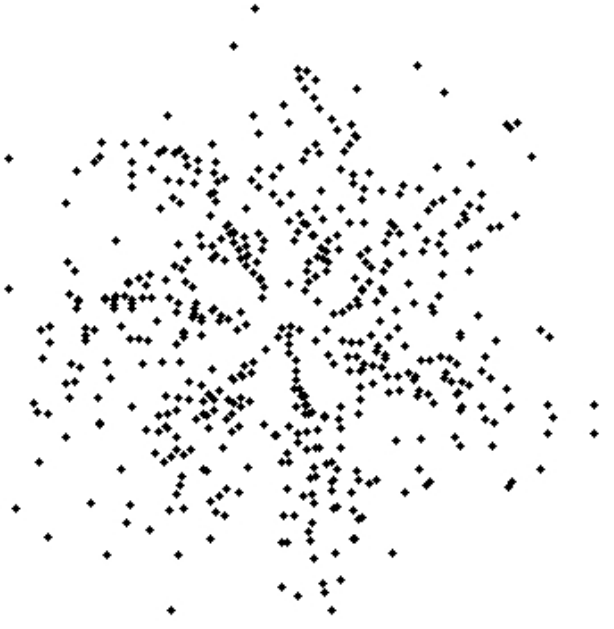


Fig. 7. A 2D graph representation of the largest cluster, consisting of 542 proteins

5 Conclusions

In this paper we proposed a modification in the TRIBE-MCL algorithm, in the terms of a variable inflation rate. An exponentially decreasing value of the inflation rate was recommended in order to deal with the nature of the problem: a high inflation rate at the beginning serves the quick establishment of hard structure of clusters, while a lower inflation rate in the followings serves the final partition quality. The proposed algorithm was found efficient in time and accurate in forming protein families. We also proposed a similarity matrix symmetrization scheme, for the case when clustering intends to ignore the evolutionary direction. Moreover, we also presented a general formulation to the inflation operation, giving the theoretical conditions of the inflation function that can replace the simple power function. Future works will aim at implementing and testing this latter proposal, and at enhancing the LGL algorithm to provide finer representation of the obtained clusters.

References

1. Adai, A.T., Date, S.V., Wieland, S., Marcotte, E.M.: LGL: Creating a map of protein function with an algorithm for visualizing very large biological networks. J. Mol. Biol. 340, 179–190 (2004)
2. Altschul, S.F., Madden, T.L., Schaffen, A.A., Zhang, J., Zhang, Z., Miller, W., Lipman, D.J.: Gapped BLAST and PSI-BLAST: a new generation of protein database search program. Nucleic Acids Res. 25, 3389–3402 (1997)
3. 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. Nucl. Acids Res. 32, D226–D229 (2004)
4. Dayhoff, M.O.: The origin and evolution of protein superfamilies. Fed. Proc. 35, 2132–2138 (1976)
5. Doolittle, R.F.: The multiplicity of domains in proteins. Ann. Rev. Biochem. 64, 287–314 (1995)
6. Eddy, S.R.: Hidden Markov models. Curr. Opin. Struct. Biol. 6, 361–365 (1996)
7. Eddy, S.R.: Profile hidden Markov models. Bioinf. 14, 755–763 (1998)
8. Enright, A.J., Ouzounis, C.A.: BioLayout: an automatic graph layout algorithm for similarity visualization. Bioinf. 17, 853–854 (2001)
9. 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)
10. Fitch, W.M.: Aspects of molecular evolution. Ann. Rev. Genet. 7, 343–380 (1973)
11. Heger, A., Holm, L.: Towards a covering set of protein family profiles. Prog. Biophys. Mol. Biol. 73, 321–337 (2000)
12. Hegyi, H., Gerstein, M.: The relationship between protein structure and function: a comprehensive survey with application to the yeast genome. J. Mol. Biol. 288, 147–164 (1999)
13. Lo Conte, L., Ailey, B., Hubbard, T.J., Brenner, S.E., Murzin, A.G., Chothia, C.: SCOP: a structural classification of protein database. Nucleic Acids Res. 28, 257–259 (2000)
14. Tsoka, S., Ouzounis, C.A.: Recent developments and future directions in computational genomics. FEBS Lett. 480, 42–48 (2000)

Feature Selection and Classification for Small Gene Sets

Gregor Stiglic[1,2], Juan J. Rodriguez[3], and Peter Kokol[1,2]

[1] University of Maribor, Faculty of Health Sciences, Zitna ulica 15, 2000 Maribor, Slovenia
[2] University of Maribor, Faculty of Electrical Engineering and Computer Science, Smetanova 17, 2000 Maribor, Slovenia
{gregor.stiglic,kokol}@uni-mb.si
[3] University of Burgos, c/ Francisco de Vitoria s/n, 09006 Burgos, Spain
jjrodriguez@ubu.es

Abstract. Random Forests, Support Vector Machines and k-Nearest Neighbors are successful and proven classification techniques that are widely used for different kinds of classification problems. One of them is classification of genomic and proteomic data that is known as a problem with extremely high dimensionality and therefore demands suited classification techniques. In this domain they are usually combined with gene selection techniques to provide optimal classification accuracy rates. Another reason for reducing the dimensionality of such datasets is their interpretability. It is much easier to interpret a small set of ranked genes than 20 or 30 thousands of unordered genes. In this paper we present a classification ensemble of decision trees called Rotation Forest and evaluate its classification performance on small subsets of ranked genes for 14 genomic and proteomic classification problems. An important feature of Rotation Forest is demonstrated – i.e. robustness and high classification accuracy using small sets of genes.

Keywords: Gene expression analysis, machine learning, feature selection, ensemble of classifiers.

1 Introduction

There are many new classification methods and variants of existing techniques for classification problems. One of them is Random Forests classifier that was presented in [1] by Breiman and Cutler. It has proven to be fast, robust and very accurate technique that can be compared with the best classifiers (e.g. Support Vector Machines [2] or some of the most efficient ensemble based classification techniques) [3]. Most of these techniques are also used in genomic and proteomic classification problems where classifiers need to be specialized for high dimensional problems. The other option is integration of feature pre-selection into classification process where initial feature set is reduced before the classification is done. Most of the early experiments using microarray gene expression datasets used simple statistical methods of gene ranking to reduce the initial set of attributes. Recently more advanced feature selection methods from the machine learning field are applied to pre-selection step in genomic and proteomic classification problems. Although a small number of genes is

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 121–131, 2008.

preferred, we try to avoid extremely small subsets of genes, like Wang et al. [4], where subsets with only two or three genes were used for classification.

This paper attempts to evaluate two widely used feature selection techniques to determine the most appropriate number of features that should be retained in preselection step to achieve the best classification performance. Additionally, this paper introduces one of the most recent classification techniques called Rotation Forest [5] to genomic and proteomic classification using small sets of genes.

Section 2 of this paper presents a novel ensemble based classification model called Rotation Forests. The rest of the paper is organized as follows: in section 3 we review the feature selection and classification methods used in this paper, in section 4 we present results of our experiments comparing classification accuracy of Rotation Forests to three classification methods. Section 5 concludes the paper and gives some future research directions on usage of Rotation Forests in genomic and proteomic classification problems.

2 Rotation Forest

Rotation Forest is a novel classification technique that was initially presented by Rodriguez et al. [4] and applied to several machine learning problems. In order to obtain successful ensembles, the member classifiers have to be accurate and diverse. Because of sampling process in Bagging and Random Forests it is necessary to obtain diverse classifiers, but using a subset of the examples to train the classifiers can degrade the accuracy of the member classifiers. Hence, a natural question is if it is possible to obtain diverse classifiers without discarding any information in the dataset.

Most ensemble methods can be used with any classification method, but decision trees are one of the most commonly used. There are ensemble methods designed specifically for decision trees, such as Random and Rotation Forests. The latter is based on the sensibility of decision trees to axis rotations; the classifiers obtained with different rotations of a dataset can be very different. This sensibility is usually considered as a disadvantage, but it can be very beneficial when the trees are used as members of an ensemble. The trees obtained from a rotated dataset can still be accurate, because they use all the information available in the dataset, but simultaneously they can be very diverse.

As in Bagging and Random Forests, each member of the ensemble is trained with a different dataset. These datasets are obtained from a random transformation of the original training data. In Rotation Forests, the transformation of the dataset consists of the following steps:

- Features are randomly grouped in k groups.
- For each group of features:
  - A new dataset consisting of all examples using sets of features from step one is created.
  - All examples of randomly selected classes are removed from this new dataset.
  - A subset of randomly chosen examples is eliminated from the new dataset (by default 25% of samples are removed)

    - PCA (Principal Component Analysis) is applied to the remaining samples in a dataset.
    - PCA components are considered as a new set of features. None of the components is discarded.
- All training samples are transformed using new variables selected by PCA for each group.
- A classifier is built from transformed training set.
- Another classifier is build by returning to the first step in case final number of classifiers in ensemble is not reached.

This transformation produces a rotation of the axis. The transformed dataset has as many examples as the original dataset and all the information that was in the original dataset remains in the transformed dataset, because none of the components is discarded and all the training examples are used for training all the ensemble methods.

The number of features in each group (or the number of groups) is a parameter of the method. The optimal value for this parameter depends on the dataset and it could be selected with an internal cross validation. Nevertheless, in this work the default value was used, and groups were formed using 3 features. The selection of the optimal value of this parameter would increase notably the time necessary for the training of the classifiers and would give an advantage of Rotation Forests with respect to other ensemble methods that do not optimize the value of any parameters.

The elimination of classes and examples of the dataset is done because PCA is a deterministic method, and it would not be difficult (especially for big ensembles) that some members of the ensemble had the same (or very similar) grouping of variables. Hence, an additional source of diversity was needed. This elimination is only done for the dataset used to do PCA; all the examples are used for training the classifiers in the ensemble.

3 Feature Selection and Classification Techniques

The main idea of feature selection is to choose a subset of variables that can significantly improve the time complexity and accuracy of a classification model. This is even more important in microarray based classification problems where initial set of features consists of thousands of gene expression values. With such a large amount of features it is of special interest to search for a dependency between optimal number of selected features and accuracy of classification model. There are two groups of feature selection techniques – filter and wrapper based methods [5]. Filter based methods rely on information content of features. Different metrics like distance metrics, information measures, correlation and consistency metrics can be used to get useful subsets when filter based feature selection is used. In wrapper approach subsets of features are selected based on how well those features classify training samples. The selection is done using the induction algorithm as a black box. Usually a search for a quality subset is done using the induction algorithm itself as a part of the evaluation function.

Symons and Nieselt [6] showed that in most microarray gene expression classification problems, filter based approaches outperform wrapper based approaches. In our experiments the following filter based approaches were used:

- ReliefF
- Support Vector Machine Recursive Feature Elimination (SVM-RFE)

Additional to two feature selection methods a set of four classification techniques was used in experiments presented in this paper:

- Random Forests
- Rotation Forests
- Support Vector Machines (SVM)
- k-Nearest Neighbors (k-NN)

A machine learning software framework named Weka [7] was used for all experiments described in this paper. Each of the above mentioned methods, except self-developed Rotation Forest algorithm is already implemented in Weka.

All above mentioned methods except Rotation Forest, that were explained earlier, are briefly described in the remainder of this section.

3.1 ReliefF

ReliefF feature selection algorithm is based on original Relief algorithm [8] that could only be used for classification problems with two class values. Basic idea of Relief algorithm is ranking of features based on their ability to distinguish between instances that are near to each other. Original algorithm was extended by Kononenko [9] so that it can deal with multi-class problems and missing values. Later it was further improved by Robnik-Sikonja and Kononenko [10] so that it is suitable for noisy data and can also be used for regression problems. Default settings for Weka implementation of ReliefF that also supports feature selection for regression problems were used in our study.

3.2 Support Vector Machines - Recursive Feature Elimination (SVM-RFE)

SVM in combination with Recursive Feature Elimination (SVM-RFE) were introduced to gene selection in bioinformatics by Guyon et al. [11]. SVM-RFE feature selection method is based on linear SVM used as the learning algorithm in recursive selection of nested subsets of features. In the final step of each cycle, all feature variables are ranked and a pre-selected number of the worst ranked features are eliminated. By default a single feature is eliminated in each round, it is also possible to remove more than one feature per round. In our experiment a setting where 50% of the remaining features are removed in each step was used.

3.3 Random Forests

Breiman upgraded the idea of Bagging by combining it with the random feature selection for Decision Trees. This way he created Random Forests, where each member of the ensemble is trained on a bootstrap replicate as in bagging. Decision trees are then grown by selecting the feature to split on at each node from randomly selected

number of features. Number of chosen features is set to $log_2(k+1)$ as in [12], where k is the total number of features.

Random Forests is an ensemble building method that works well even with noisy content in the training dataset and is considered as one of the most competitive methods that can be compared to boosting [13].

3.4 Support Vector Machines (SVM)

SVM are increasingly popular classifiers in many areas, including bioinformatics [2]. The most basic variant of SVM use linear kernel and try to find an optimal hyperplane that separates samples of different classes. When classes can be linearly separated, the hyperplane is located so that there is maximal distance between the hyperplane and the nearest sample of any of the classes. In cases when samples cannot be linearly separated, there is no optimal separating hyperplane; in such cases, we try to maximize the margin but allow some classification errors. For all experiments in this study an advanced version of SVM called Sequential Minimal Optimization (SMO) proposed by Platt [14, 15] is used. It offers very quick and reliable learning of the decision models based on SVM.

3.5 k-Nearest Neighbors (k-NN)

Nearest Neighbors classifier is a typical representative of case based classifiers where all samples are stored for later use in the classification process [16]. It aims to classify samples according to similarities or distance between them. A class value is defined using class values of k nearest samples. Similarity to neighboring samples is calculated using distance between samples that is usually measured using Euclidean distance metric.

Another important parameter that has to be set is number of neighbors that will be used for calculation of class value. The most common settings for this parameter are 1, 3 or 5. In our experiments we always use 5 neighbors for class value estimation whose vote for final class is weighted according to their distance from the neighbor.

k-NN based classifiers are most useful in cases with continuous attribute values that also include genomic and proteomic datasets. It is also welcome if datasets contain low number of samples (e.g. gene expression datasets), because of high computational cost of k-NN classification process when number of samples rises.

4 Experiment Settings and Results

In our experiments two feature selection methods from section 3 were tested on 14 publicly available genomic and proteomic datasets presented in Table 1. No modification of original data in form of normalization or discretization was needed. All datasets are available at Kent Ridge Biomedical Data Set Repository [17] where additional information including references to original work for each of the datasets can be found. All tests were done using 10-fold cross-validation measuring the classification accuracy that can be calculated as a quotient between number of correctly classified and number of all samples in a testing set. To avoid feature selection bias, as discussed in Ambroise and McLachlan [18], a separate feature selection process was done for each training and test set during 10-fold cross validation.

Table 1. Details for genomic and proteomic datasets from Kent Ridge repository

Dataset	Original Work	Genes	Patients	Classes
ALL	Yeoh et al.	12558	327	7
ALLAML	Golub et al.	7129	72	2
Breast	Van't Veer et al.	24481	97	2
CNS	Mukherjee et al.	7129	60	2
Colon	Alon et al.	2000	62	2
DLBCL	Alizadeh et al.	4026	47	2
DLBCL-NIH	Rosenwald et al.	7399	240	2
DLBCL-Tumor	Shipp et al.	6817	77	2
Lung	Gordon et al.	12533	181	2
Lung-Harvard	Bhattacharjee et al.	12600	203	5
Lung-Michigan	Beer et al.	7129	96	2
MLL	Armstrong et al.	12582	72	3
Ovarian	Petricoin et al.	15154	253	2
Prostate	Singh et al.	12600	102	2

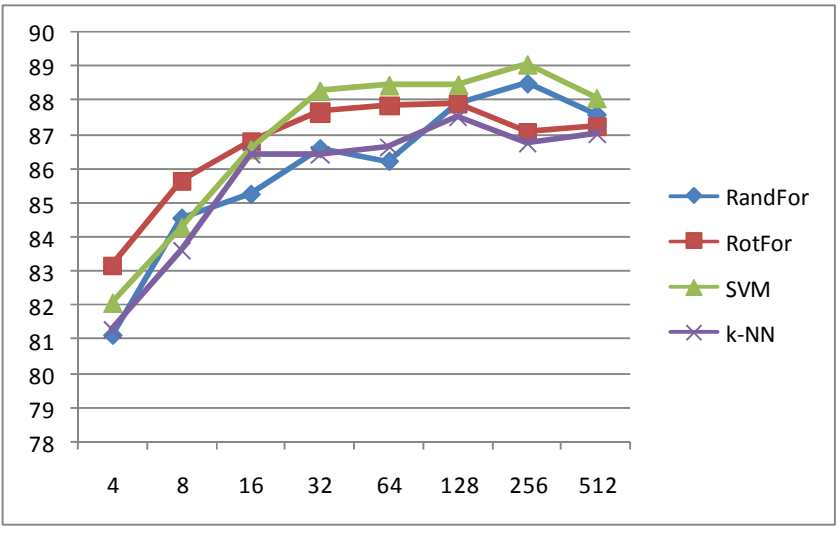


Fig. 1. Average accuracy on all datasets using four classification methods on reduced datasets with different number of genes (ReliefF feature selection)

Each ensemble (Random Forests and Rotation Forest) consisted of 100 decision trees, while number of features used for classification ranged from 4 to 512 and was defined as 2^i, where $i = 2,\ldots,9$.

In the first experiment a set of classification accuracy measurements was done based on reduced gene sets. ReliefF was used for feature selection using default settings of Weka environment. Averaged classification accuracy for specific feature

selection settings using k-top most ranked genes, where k ranges from 4 to 512, is presented in Figure 1. It can be observed that with number of selected features under 16, Rotation Forest outperforms all other methods, while SVM take over for higher numbers of selected genes. The highest classification accuracy was obtained using 256 most significant genes according to ReliefF, using SVM classifier (89.06%).

To obtain a better picture of dominance between compared methods and to avoid unreliable averaging of results, we did a comparison using statistical test. Non-parametric Friedman's statistical test [19] was used to compute average ranks of compared methods. Figure 2 presents Friedman's average ranks for all compared classifiers and different feature selection settings using ReliefF. It can be seen that Rotation Forest strongly dominates all other methods in the first three points, while SVM strongly dominate Rotation Forest in the last two settings. Average ranks shown in Figure 2 were calculated for results from all 14 datasets using SPSS statistical tools. Average rank, in our case of four compared methods, can have a value from 1 to 4. If a method hypothetically wins all comparison tests based on average accuracy it would be assigned an average rank of 4, while method losing all pairwise comparisons would score an average rank of 1.

The same settings as in the first experiment were used for the second experiment where SVM-RFE was used for feature selection tasks. Figure 3 presents results of average accuracy levels across all 14 datasets. It can be observed that Rotation Forest classifier is in front all the way up to the point of the highest classification accuracy at 128 selected genes (89,51% accuracy). Similar to the previous experiment the performance of Rotation Forest deteriorates with high numbers of selected features, where SVM perform better again.

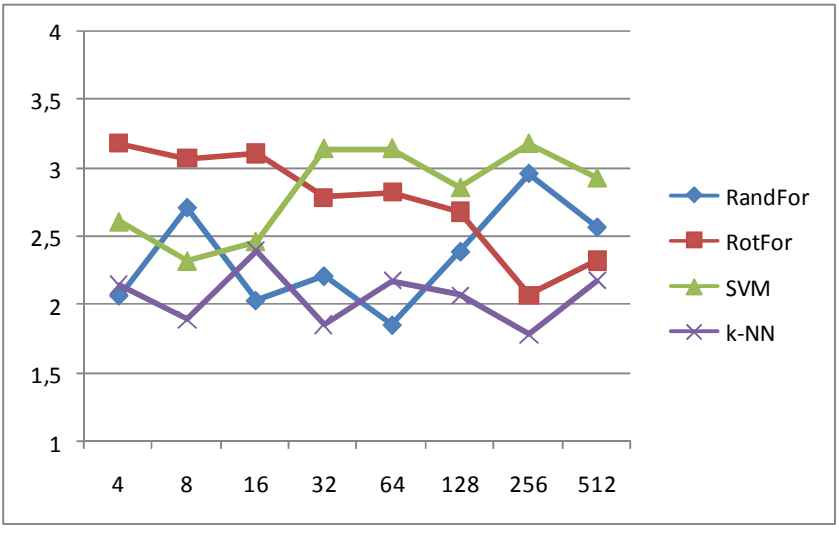


Fig. 2. Average rank for all four classification methods on reduced datasets with different number of genes using ReliefF feature selection

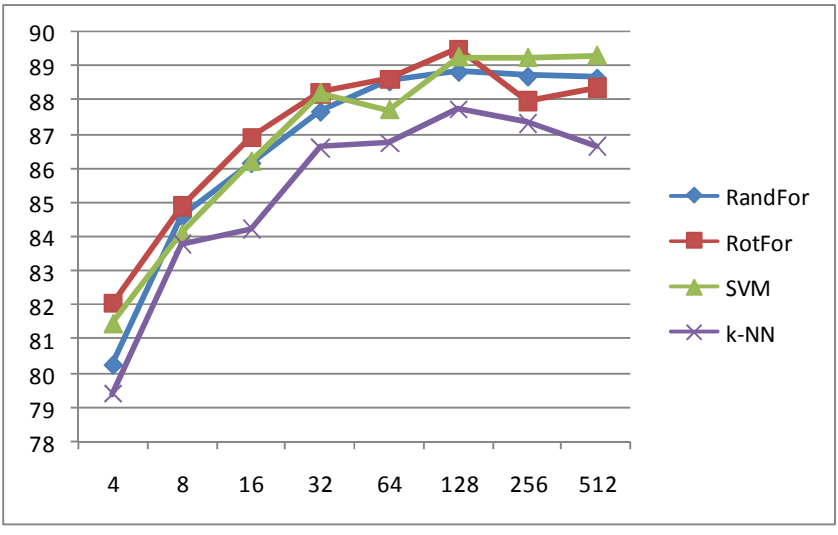


Fig. 3. Average accuracy using SVM-RFE based feature selection

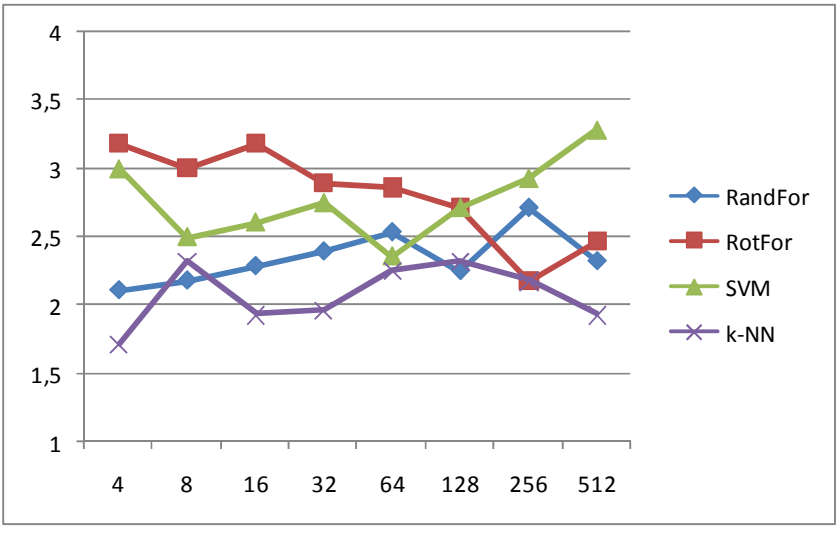


Fig. 4. Average ranks using SVM-RFE based feature selection

Friedman test shows significant differences among compared methods again. When Friedman test hypothesis is rejected, it is usually followed by a pairwise comparison of classification methods. This can be done by Wilcoxon signed-rank test [20] that represents non-parametric alternative to the paired Student t-test for two related measurements.

Wilcoxon test was done on all pairwise combinations in the first ReliefF based and the second SVM-RFE based experiments. In case of ReliefF results a significant dominance of Rotation Forest and SVM methods compared to Random Forests and k-NN is shown. However there is no significant difference between Rotation Forest and SVM ($p = 0.828$). There is also no significant difference between results of Random Forests and k-NN ($p = 0.167$). Results were almost the same for SVM-RFE feature selection with the only exception – Rotation Forest did not manage to significantly outperform Random Forests ($p = 0.066$) although it was performing better.

Figure 4 confirms results from Figure 3 where it can be seen that Rotation Forest dominates all other classification methods up to the and including a point where 128 most significant genes were selected.

Given the highest accuracy of 89.51% one would assume that a combination of Rotation Forest and SVM-RFE based feature selection using 128 most significant genes is the best combination. But in many cases biologists are interested in smaller sets of genes that can be more descriptive and give more information than large sets of genes that are difficult to interpret.

Figure 5 shows a combination of both best methods (Rotation Forest and SVM) using average accuracy levels for both feature selection techniques simultaneously. One should notice that although SVM-RFE achieves better average accuracy overall, it is evident that ReliefF should be preferred when a small number of selected genes should be obtained.

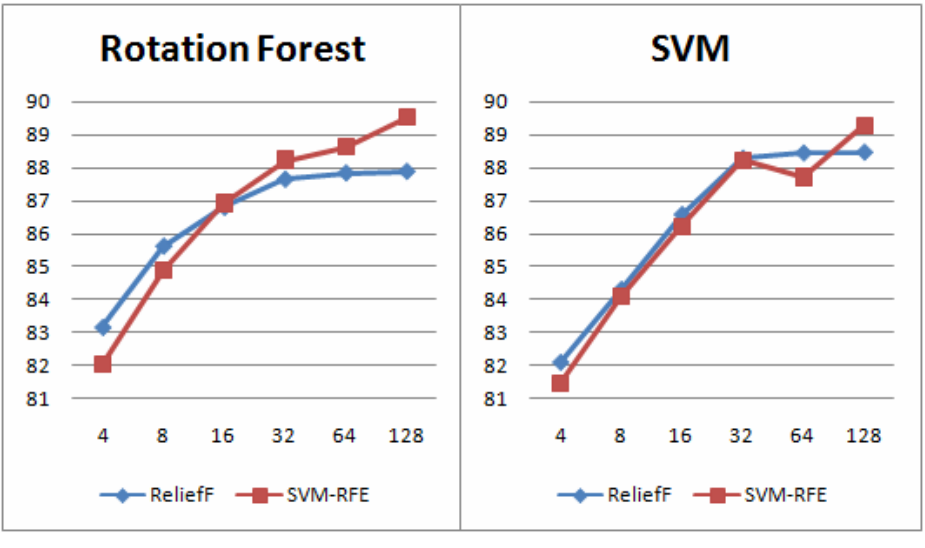


Fig. 5. Simultaneous comparison of ReliefF and SVM-RFE feature selection techniques

5 Conclusions

This paper presents a novel classification method for genomic and proteomic data classification. The results indicate that Rotation Forests can be considered as one of

the most useful classification techniques on small gene sets. Another important issue that was researched in this paper is a problem of finding the optimal number of genes to get the most out of the classifier. It was shown that there is no optimal solution to this problem. One can get significantly different results when comparing classification accuracy when an extremely low number of genes is used to classification accuracy in higher dimensional problems using different classifiers. It is however practically impossible to define a fixed number of features that should be selected for optimal classification performance. On the other hand it was obvious that there are some classification techniques that should be used when a low number of genes is preferred (Rotation Forest) and some methods that demand higher number of genes (SVM) for optimal classification accuracy. It was shown that ReliefF should be used for extremely small sets of selected features, while SVM-RFE performs better in higher dimensions. It should also be noticed that SVM-RFE cannot be used for regression problems, where ReliefF will be the only available solution out of the two presented feature selection methods.

One of the issues for the future is evaluation of Rotation Forests on even more datasets. Unfortunately it is not possible to directly use Rotation Forests for feature selection, but there are other ways of using the power of Rotation Forests. One of such is their ability to very accurately estimate the similarity of cases and could therefore be used for implementation of clustering algorithms. As we know clustering is also one of the most widely used methods in unsupervised analysis that is being used in bioinformatics and therefore opens a lot of new areas where Rotation Forests could be used.

References

1. Breiman, L.: Random forests. Machine Learning 45, 5–32 (2001)
2. Vapnik, V.: Statistical learning theory. John Wiley and Sons, New York (1998)
3. Caruana, R., Niculescu-Mizil, A.: An empirical comparison of supervised learning algorithms. In: Proceedings of the 23rd international Conference on Machine Learning (ICML 2006), vol. 148, pp. 161–168 (2006)
4. Wang, L., Chu, F., Xie, W.: Accurate Cancer Classification Using Expressions of Very Few Genes. IEEE/ACM Transactions on Computational Biology and Bioinformatics 4(1), 40–53 (2007)
5. Rodríguez, J.J., Kuncheva, L.I., Alonso, C.J.: Rotation forest: A new classifier ensemble method. IEEE Transactions on Pattern Analysis and Machine Intelligence 28(10), 1619–1630 (2006)
6. John, G.H., Kohavi, R., Pfleger, K.: Irrelevant Features and the Subset Selection Problem. In: Proceedings of the Eleventh International Conference on Machine Learning, pp. 121–129 (1994)
7. Symons, S., Nieselt, K.: Data Mining Microarray Data – Comprehensive Benchmarking of Feature Selection and Classification Methods. Pre-print, http://www.zbit.uni-tuebingen.de/pas/preprints/GCB2006/SymonsNieselt.pdf
8. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools with Java implementations. Morgan Kaufmann, San Francisco (2005)
9. Kira, K., Rendell, L.A.: A practical approach to feature selection. In: Proceedings of International Conference on Machine Learning (ICML1992), pp. 249–256 (1992)

10. Kononenko, I.: Estimating attributes: analysis and extension of relief. In: Proceedings of European Conference on Machine Learning (ICML1994), pp. 171–182 (1994)
11. Robnik-Sikonja, M., Kononenko, I.: An adaptation of Relief for attribute estimation in regression. In: Machine Learning: Proceedings of the Fourteenth International Conference (ICML 1997), pp. 296–304 (1997)
12. Guyon, I., Weston, J., Barnhill, S., Vapnik, V.: Gene selection for cancer classification using support vector machines. Machine Learning 46, 389–422 (2002)
13. Díaz-Uriarte, R., Alvarez de Andrés, S.: Gene selection and classification of microarray data using random forest. BMC Bioinformatics 7, 3 (2006)
14. Dietterich, T.G.: Ensemble Learning. In: Arbib, M.A. (ed.) The Handbook of Brain Theory and Neural Networks, pp. 405–408. The MIT Press, Cambridge (2002)
15. Platt, J.: Machines using Sequential Minimal Optimization. In: Schoelkopf, B., Burges, C., Smola, A. (eds.) Advances in Kernel Methods - Support Vector Learning (1998)
16. Keerthi, S.S., Shevade, S.K., Bhattacharyya, C., Murthy, K.R.K.: Improvements to Platt's SMO Algorithm for SVM Classifier Design. Neural Computation 13(3), 637–649 (2001)
17. Mitchell, T.: Machine Learning. McGraw Hill, New York (1997)
18. Kent Ridge Biomedical Data Set Repository: sdmc.i2r.a-star.edu.sg/rp/
19. Ambroise, C., McLachlan, G.J.: Selection bias in gene extraction on the basis of microarray gene-expression data. Proc. Natl. Acad. Sci. USA 2002 99, 6562–6566 (2002)
20. Friedman, M.: A comparison of alternative tests of significance for the problem of m rankings. The Annals of Mathematical Statistics 11(1), 86–92 (1940)
21. Wilcoxon, F.: Individual comparisons by ranking methods. Biometrics Bulletin 1, 80–83 (1945)

Pseudoknot Identification through Learning TAG_{RNA}

Sahar Al Seesi, Sanguthevar Rajasekaran, and Reda Ammar

Computer Science and Engineering Department, University of Connecticut
{sahar,rajasek,reda}@engr.uconn.edu

Abstract. Studying the structure of RNA sequences is an important problem that helps in understanding the functional properties of RNA. Pseudoknot is one type of RNA structures that cannot be modeled with Context Free Grammars (CFG) because it exhibits crossing dependencies. Pseudoknot structures have functional importance since they appear, for example, in viral genome RNAs and ribozyme active sites. Tree Adjoining Grammars (TAG) is one example of a grammatical model that is more expressive than CFG and has the capability of dealing with crossing dependencies. In this paper, we describe a new inference algorithm for $TAG_{RNA,}$ a sub-model of TAG. We also introduce an RNA structure identification framework, TAG_{RNA}Inf, within which the TAG_{RNA} inference algorithm constitutes the core of the training phase. We present the results of using the proposed framework for identifying RNA sequences with pseudoknot structures. Our results outperform those reported in [14] for the same problem that employs a different grammatical formalism.

1 Introduction

In recent times there has been an observed acceleration in the RNA structure determination and analysis [11] owing to its paramount importance. This is partly due to the discovery of many new functional RNAs, such as miRNAs and tmRNAs [3] [16] [29]. Another factor that has led to the speeding up of RNA structural research is the rise of the RNA World Hypothesis [9] which suggests that the current DNA and protein world has evolved from an RNA based world. This Hypothesis is supported by the fact that RNA can carry genetic information like DNA and it is capable of catalyzing reactions like proteins (rRNA). Genetic information of some existent viruses is carried in RNA form [15]. Since the function of bimolecular sequences depends on its structure, analyzing RNA structures is essential to create new drugs and understand genetic diseases [6] [20]. Computational methods can provide less expensive solutions to structure analysis than other methods such as nuclear magnetic resonance and x-ray crystallography.

In the early 90's, David Searls studied the linguistics of biological sequences [23]. He suggested the use of formal grammars as a tool to model and analyze DNA, RNA, and proteins. The use of grammars has attracted the attention of many researchers [7] [26] because it can model long range interactions. In addition, grammatical models are concise and easy to understand representation of structures of sequence families. Thus, it is considered to be a natural analytical approach to fully understanding the structure and properties of these sequences. Results for secondary structure prediction

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 132–143, 2008.

and multiple sequence alignment agree with and sometimes suggest improvements over traditional methods [21].

Pseudoknot is one type of RNA structures that cannot be modeled with Context Free Grammars (CFG) because it exhibits crossing dependencies. Pseudoknot structures have functional importance since they appear, for example, in viral genome RNAs [15], ribozyme active sites [25], and tmRNA [28]. Among the available research in analyzing pseudoknot structures are the works of Akutsu [2], Dirks and Pierce [8], and Reeder and Giegerich [18]. These algorithms are not based on formal grammars. In the area of modeling molecular sequences grammatically, more than one model, capable of representing pseudoknots, have been presented. Cai *et. al.* [7] proposed Parallel Communicating Grammar Systems (PCGS) with an $O(n^6)$ time parsing algorithm. Another model which also requires $O(n^6)$ parsing time has been proposed by Rivas and Eddy[19]. Uemura et. al. [26] suggested the use of a submodel of TAG, TAG_{RNA}. Our solution is based on the TAG_{RNA} model.

Recently, there has been a special focus on the use of grammatical inference in bioinformatics. Sakakibara has published [22] in which he discusses the general merits of using grammatical inference in bioinformatics. Brazma *et. al.* [5] have proposed an approach to discover simple grammars for families of biological sequences. The grammatical formalisms they use are subclasses of regular patterns. On the use of grammatical inference to analyze RNA structures with Pseudoknots, Laxminarayana *et. al.* [13] presented an inference algorithm for Terminal Distinguishable Even Linear Grammars (TDELG), and they have shown how to use this algorithm in an Infer-Test model for the detection of a pseudoknot structure in an RNA sequence. The experimental results they presented [14] show 54% sensitivity when using 50% of the RNA sample for training. The sensitivity rises to 85% only when 90% of the sample is used for training. Specificity was not reported. This is the same problem as the one we address, and our results outperform those numbers, as it will be shown. Takakura *et. al.* have published [24] in which they give a linear time algorithm for generating probabilistic TAG_{RNA} from alignment data. They use the inferred grammar to find new members of nc-RNA families, which is a different problem from the one we address in this paper.

The use of grammatical inference to automate the grammar building step is essential in facilitating the use of grammatical formalism by biologists. Otherwise, the biologist will always be dependent on a grammar expert. In this work, we present a complete RNA structure identification framework, TAG_{RNA}Inf, capable of handling pseudoknot structures. By structure identification we mean, given an RNA sequence, we answer the question of whther it exhibits a certain structure or not. In our approach, the structure is represented by a TAG which is inferred from a training set. We describe a new polynomial time inference algorithm for TAG_{RNA} which constitutes the core of the training phase within the identification framework. We evaluate our solution experimentally through calculating the sensitivity and specificity of identification.

2 TAG and TAG_{RNA}

Tree Adjoining Grammars (TAGs) were originally introduced, by Joshi *et. al.* [12], for use in the field of natural language processing. Uemura *et. al.* [26] defined a subclass of TAGs, $TAG_{RNA,}$ suitable to model RNA pseudoknot structures. They developed

an $O(n^5)$ time parsing algorithm for TAG_{RNA}. Before describing TAG_{RNA} we will first give a brief introduction to the original TAG model.

A Tree Adjoining Grammar (TAG) is defined to be a 5-tuple $(T \cup \{\varepsilon\}, N, I, A, S)$, where T is a set of terminal symbols, N is a set of non-terminal symbols, ε is the empty string symbol, and S is the starting symbol. I and A are defined as follows:

- I (initial trees): A finite set of finite trees with the internal nodes' labels belonging to $N \cup \{S\}$, the leaves' labels belonging to $T \cup \{\varepsilon\}$, and the root is labeled with S.
- A (auxiliary trees): A finite set of finite trees with the internal nodes' labels belonging to $N \cup \{S\}$, and the leaves' labels belonging to $T \cup \{\varepsilon\}$ except one leaf node which has the same label as the root. This special leaf node is called a *foot node*.

Trees belonging to $I \cup A$ are called elementary trees. A tree derived by composing two other trees is called a derived tree. Trees can be composed together using the adjoining operation. The adjoining operation composes an auxiliary tree α with a foot node labeled X with any other tree β that has some internal node with the same label X. The operation works as follows: we start with the tree β and we extract the sub-tree rooted at the internal node labeled with X (let that sub-tree be γ), and replace it with the α. Then at the foot node of α, we reinsert γ. The adjoining operation is illustrated in Fig. 1. Let $T = \{ t : \exists i \in I$ s.t. t can be derived from $i\}$, then L(TAG) consists of the yield of all the trees in T.

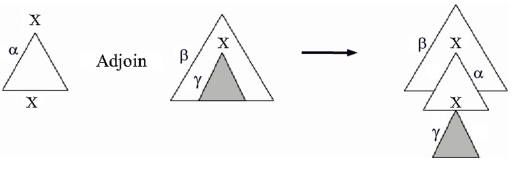


Fig. 1. The Adjoining Operation

In [26], Extended Simple Linear TAG (ESLTAG) is defined to be a subclass of TAG with adjoining constraints [27]. In ESLTAG, the adjoining operation can occur only at internal nodes tagged with the symbol *, and the number of these nodes is restricted. TAG_{RNA} is a sub-class of ESLTAG where only five types of elementary trees are allowed (Fig. 2)[1]. Each type of tree is responsible for a specific kind of branching or structural form that an RNA sequence can have.

3 The Structure Identification Framework

We introduce a complete RNA structure identification framework, TAG_{RNA}Inf, which is capable of handling pseudoknot structures. Within this framework, we present a new inference algorithm for TAG_{RNA} which constitutes the core of the training phase.

[1] Tree types of TAG_{RNA} will be explained further in section 3.1.2.

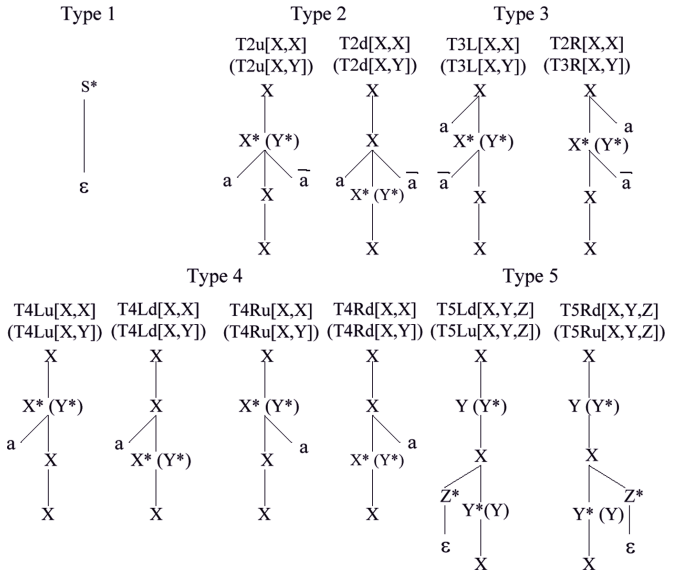


Fig. 2. TAG_{RNA}

Fig. 3 depicts the proposed framework. In the training phase, the inference algorithm is fed with a positive training set with structural information. The algorithm will generate a grammar for the provided sample. Then, the same sample along with a negative sample and the grammar generated by the inference algorithm will go through a TAG parser. For each input sequence the TAG parser will output a score. These

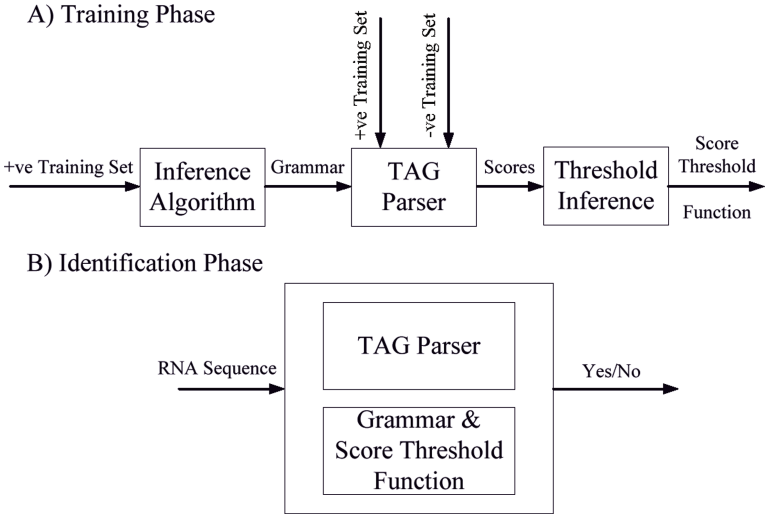


Fig. 3. TAG_{RNA}Inf : RNA Structure Identification Framework

scores will be the input to a threshold function inference module. The inferred threshold function will be used in the identification phase.

Several scoring functions can be used. For example, it can be either the number of base pairs or the minimum free energy (mfe) of the RNA sequence structure. Also, a probabilistic function can be used to generate the scores. Currently, we use the number of base pairs as the scoring function. We intend to investigate other alternatives. The inferred grammar and the scoring threshold function will be used by a TAG parser in the identification phase. Given an RNA sequence, the identification module will be able to check if this sequence has a certain structure such as a pseudoknot.

3.1 The Inference Algorithm

The grammar inference adopted here is a three step process. The input is a set of sequence data that includes the structure of each sequence, and the output is a grammar that models the input sample. If the input sample includes at least one sequence representing each RNA structure in the population being modeled, the output grammar will be a correct model for the RNA population from which the sample was drawn. For a population S, a grammar G is considered to be a correct model of S iff $S \subseteq L(G)$. For the purpose of evaluating the inferred grammar within the proposed framework, however, we calculate the sensitivity and specificity of identification.

The three steps of the inference process are: the pattern generation, the single pattern grammar generation, and the final grammar composition.

3.1.1 Pattern Generation

Definition: Let $(\chi, \overline{\chi^r})$ and $(\gamma, \overline{\gamma^r})$ be two substring pairs in a pattern p, we call the two pairs $(\chi, \overline{\chi^r})$ and $(\gamma, \overline{\gamma^r})$ a crossing dependency if $i < k < j < l$ where i, j, k, and l are the positions of χ, $\overline{\chi^r}$, γ, and $\overline{\gamma^r}$, respectively, in p.

The inputs to this phase are: the sequence size, the number of stems in the input sequence (n), the starting and ending indices of each stem in the sequence represented as a 4-tuple $(l_{i1},l_{i2},m_{i1},m_{i2})$, where if χ_i , $\overline{\chi_i^r}$ are the two strands of a stem in the sequence, l_{i1} and l_{i2} are the starting and ending positions of x_i, respectively, and m_{i1} and m_{i2} are the starting and ending positions of $\overline{\chi_i^r}$, respectively.

The pattern generation is based on sorting the pairs (l_{i1}, l_{i2}) and (m_{i1},m_{i2}) for all values of $i \leq n$ resulting in a sorted list P of $2n$ pairs (p_{i1}, p_{i2}). We maintain a link from each pair of numbers to its corresponding substring symbol χ_i's or $\overline{\chi_i^r}$'s. Thus, once the number pairs are sorted, the χ's are consequently sorted. Because any two intervals (p_{i1},p_{i2}), (p_{j1},p_{j2}) are non-overlapping we can perform the sort on the first value in the pairs, and because we are dealing with integers we can use radix sort. This will require linear time in the number of stems n. The generated pattern consists of the sorted χ's and $\overline{\chi^r}$'s with w's inserted, to represent loops in the RNA structure, wherever there is a gap between the numbers p_{i2} and $p_{(i+1)1.}$ The number of w's in a pattern must be less than or equal $2n + 1$. The insertion can be done by copying the sorted x_i's and $\overline{\chi_i^r}$'s sequentially in an array of size $4n + 1$. During the sequential copying process, we check for gaps and insert w's as necessary. This also requires linear time in n.

After the pattern is generated and before any grammar inference can be performed, we must insert the empty string symbol, ε, in the pattern. The empty string appears in TYPE 1 and TYPE 5 trees of TAG_{RNA} (see Fig. 2). Currently, we support patterns that have exactly one ε symbol. Considering all the crossing dependencies $((x, \overline{x^r}), (y, \overline{y^r},))$, ε is inserted at $i + 1$ where i is the index of the rightmost $\overline{x^r}$ in the pattern.

An Example

The pseudoknot structure at the gag-pol translational readthrough site of spleen necrosis virus [4] has the following pattern

$$w_1 x_1 w_2 x_2 w_3 x_3 \overline{x_2^r} w_4 \overline{x_1^r} w_5 \overline{x_3^r}$$

This pattern has two crossing dependencies, $((x_1, \overline{x_1^r}), (x_3, \overline{x_3^r}))$ and $((x_2, \overline{x_2^r}), (x_3, \overline{x_3^r}))$. Because $\overline{x_1^r}$ comes to the right of $\overline{x_2^r}$, the ε is inserted after $\overline{x_1^r}$.

The ε location identification is facilitated by generating a list, **links**, in which for each pair of dependent substrings $(x_i, \overline{x_i^r})$, **links**$[i] = (j, k)$ where j and k are the positions of x_i and $\overline{x_i^r}$ in the pattern, respectively. The list **links** is simply constructed by scanning the pattern once and filling the corresponding entries for each x_i and $\overline{x_i^r}$ in **links** as they are scanned in the pattern. Thus, the time required for generating **links** is $O(n)$, where n is the number of stems in the pattern. A simple search on links is performed to determine the position of ε which satisfies the above condition. This also requires linear time in the length of the pattern and consequently linear in the number of stems n. Thus, the total time required for this phase of the algorithm is $O(n)$.

3.1.2 Generating Grammar for a Single Pattern

The general idea of the grammar generation for a pattern is to choose the correct types of trees, from the TAG_{RNA} model, that can model dependencies between pairs of substring symbols in the pattern, or simply model independent substrings. The choice is dependent on the relative positions of the substrings being modeled and the position of ε. If we look at the types of trees in TAG_{RNA}, illustrated in Fig. 2, we notice the following. First, there is only one type of initial tree which is of TYPE 1. Thus the generated grammar will always have one of those trees. TYPE 2 trees can be used to model dependent pairs of substrings $(x, \overline{x^r})$ that appear on opposite sides of ε. TYPE 3 trees can be used to model dependent pairs of substrings $(x, \overline{x^r})$ that appear on the same side of ε. Finally, TYPE 4 trees can be used to model independent substrings (loops in the RNA structure) that are represented by w symbols in the generated pattern. As we mentioned above, we currently support patterns that have exactly one ε symbol. TYPE 5 trees can be used to model more complex structures with branching. At the moment, we do not make use of TYPE 5 Trees.

To generate the grammar for one pattern, the pattern is parsed one symbol at a time. For each independent substring symbol w or dependent pair of symbols $(x, \overline{x^r})$, two auxiliary trees are generated. The first tree has the same non-terminal label for the root, foot node and the adjoining node. This tree can be used recursively to generate terminals $\{c, g, u, a\}$ in the RNA sequence corresponding to the currently parsed pattern

substring symbol(s). The second tree is the same as the first one except that it has a different non-terminal label for the adjoining node. This tree allows transitioning to another substring or pair of substrings in the pattern. Generating a grammar for a single pattern requires linear time in the length of the pattern and, consequently, the number of stems is O(n). Algorithmic details and complexity analysis can be found in [1].

3.1.3 Final Grammar Composition

When the whole sample is processed, we will have a set of grammars, each representing the pattern of a single input example. To generate one grammar, which is representative of the whole input sample, we need to combine these grammars. The TAG union operator, defined in [27] can be used for this purpose. The union of two TAGs consists of the union of the elementary trees of both grammars.

If the input sample includes at least one sequence representing each RNA structure in the population being modeled, then the output grammar is a correct model for that population. For an input sample of size m RNA examples, the total time required by the algorithm is O(mn) where n is the maximum number of stems in an RNA example. Thus the algorithm is linear in the size of the input.

In order to reduce the size of the final grammar, the grammar composition step can be adjusted to check for input examples that have the same pattern. To accomplish that, any generated pattern must be saved. When a new example is encountered, a pattern is generated for it. Then, the set of saved patterns is searched. If the same pattern was generated before, we move to the next input example. If not, a grammar is inferred for the new pattern. The search process requires O(mn) time for one pattern. Thus, this modification increases the complexity to O(m^2n). Even though this is more than linear time, this algorithm is practical.

In practice, however, we prefer to keep the generated grammars separate. In later stages of the training phase and in the identification phase, the TAG parser will parse the input sequence against each of the generated grammars separately which is equivalent to parsing it against the union grammar. This will not increase the parsing complexity. On the contrary, it will help in optimizing it through eliminating the least effective grammars, as explained in section 3.3.

An Example

The input in this example is the following set of 4-tuples representing stems' positions for the delta ribozyme structure of the hepatitis delta virus (Italy variant) as it appears in the Pseudobase website [4].

(1,7,33,39), (16,19,81,84), (20,22,30,32), (43,49,68,74), (54,57,62,65)

First the corresponding pattern is generated:

$$x_1 w_1 x_2 x_3 w_2 \overline{x_3^r}\,\overline{x_1^r}\, \varepsilon w_3 x_4 w_4 x_5 w_5 \overline{x_5^r} w_6 \overline{x_4^r} w_7 \overline{x_2^r} w_8$$

Table 1 shows the output trees generated for each substring or pair of substrings in the above pattern. The substrings appear in the order in which they are processed by the algorithm.

Table 1. Output Trees for delta ribozyme structure of the hepatitas delta virus

Substring/Substring Pair	Generated Auxiliary Trees
$(x_1, \overline{x_1^r})$	T3L[S,S] & T3L[S,A]
w_1	T4Ld[A,A] & T4Ld[A,B]
w_8	T4Rd[B,B] & T4Rd[B,C]
$(x_2, \overline{x_2^r})$	T2d[C,C] & T2d[C,D]
$(x_3, \overline{x_3^r})$	T3L[D,D] & T3L[D,E]
w_2	T4Ld[E,E] & T4Ld[E,F]
w_3	T4Ru[F,F] & T4Ld[F,G]
w_7	T4Rd[G,G] & T4Rd[G,H]
$(x_4, \overline{x_4^r})$	T3R[H,H] & T3R[H,I]
w_4	T4Ru[I,I] & T4Ru[I,J]
w_6	T4Rd[J,J] & T4Rd[J,K]
$(x_5, \overline{x_5^r})$	T3R[K,K] & T3R[K,L]
w_5	T4Ru[L,L] & T4Ru[L,M]

3.2 The TAG Parser and the Scoring Function

We use a TAG parser in the training phase and the identification phase. In the training phase, the parser is used to generate a set of scores for the positive and negative training sequences. The generated scores are then input to a threshold function inference module. The scoring function used is a simple one that counts the number of base pairs for the sequence structure under a certain grammar. If there is more than one possible structure, due to the nondeterministic nature of the grammar, the parser will output the maximum score. As mentioned in section 3.2.3, a separate grammar for each pattern resulting from the positive training will be generated. The score for a certain sequence under the union of a set of grammars will, again, be the maximum of the scores generated from all grammars in the set.

The parser we used is an implementation of Rajasekaran's [17] and Vijay-Shankar and Joshi's [27] algorithms with some minor modifications. In our implementation of the TAG parser, in addition to n^4 matrix, A, maintained by the parser, we associate a list of 4-tuples with every node in the grammar. For a node α, a tuple $(i,j,k,l) \in List(\alpha)$ iff $\alpha \in A(i,j,k,l)$. This idea, borrowed from [17], does not improve the worst time complexity of the parser which is $O(n^6)$; however, it improves the average run time in practice due to sparsity of the matrix A. Another modification is the fact that the parser generates a score for each sequence instead of a yes/no output.

3.3 The Threshold Function Inference Module

This module infers a score threshold function $Th(l) = p$. A sequence s of size l is considered to have the RNA structure represented by a grammar G iff the TAG parser accepts s under G, with score $p_s \geq p$. Th(l) is a step function defined as follows:

$$Th(l) = p, \ i \leq l < j \tag{1}$$

Since both sensitivity and specificity are important criteria we infer a function Th(l) that maximizes the sum of sensitivity and specificity. This is achieved through calculating a function S for all possible paths of Th from l = 0 to l = n, where S is the maximum gain in specificity - loss in sensitivity resulting from each step the function Th makes and n is the maximum sequence size. Then Th is constructed by tracing back the path resulting in maximum S. Calculating maximum S can be done in $O(n^3m^2)$ time and $O(n^2m^2)$ memory using dynamic programming, where m is maximum reported score for the input sample.

Assume, with out loss of generality, that the number of sequences in the positive sample and the negative sample are equal. Let S(i,j,p,q) be maximum gain in specificity – loss in sensitivity possible for a threshold function segment that starts at Th(i) = p, and ends at Th(j) = q. Then, the dynamic programming recurrence formulae are given below

$$S(i,i,p,q) = S(i,i,q,q) = (\text{ the number of negative samples of length i with score} < q - \text{the number of positive samples of length i with score} < q) / \text{the sample size.} \quad (2)$$

and

$$\begin{aligned} S(i,j,p,q) &= S(i,i,p,p) + S(j,j,q,q) && , j = i+1 \\ &= \text{Max}_{\,p \le l \le m \le q} \left(S(i+1,j-1,l,m) + S(i,i,p,p) + S(j,j,q,q) \right) && , j \ge i+2 \end{aligned} \quad (3)$$

3.4 Selecting the Best Grammar Combination

As mentioned earlier, the scores resulting from each grammar for the patterns generated by the training sequences are reported separately. Instead of inferring the threshold function from the maximum score calculated over all the generated grammars, we try all possible combinations out of these grammars and pick the combination that generates the maximum sensitivity + specificity for the training set. This approach has two advantages. First, it eliminates the least informative and/or nearly redundant grammars. Meanwhile it enhances the time performance for the identification phase by reducing the number of grammars, or in other words, the size of the overall grammar.

This idea can further be used to restrict the number of grammars used to preset a maximum; thus choosing the best combination out of three or four grammars, for example. Even though trying out all possible combinations requires exponential time in the number of grammars, the number of grammars is usually small, resulting in the feasibility of this solution.

4 Experimental Results

To evaluate the effectiveness of the inferred grammars within $TAG_{RNA}Inf$, we calculate the sensitivity and specificity of identification.

$$Sensitivity = \frac{TP}{TP + FN} \text{ and } Specificity = \frac{TN}{TN + FP} \quad (4)$$

where TP, TN, FP, and FN are the number of true positives, the number of true negatives, the number of false positives and the number of false negatives respectively.

We used the grammar inference algorithm to infer a grammar for H-type pseudoknot from a positive training set with structural information. Then we used positive and negative training sets to infer the threshold function. The inferred grammar and score threshold function were applied to a test set of RNA sequences and the sensitivity and specificity were calculated.

For this experiment, we used these data sources:

- The positive data population of H-type pseudoknot sequences was collected from Pseudobase [4], the tmRNA database [28], and pseudoknot familes in the Rfam database [10]. We arbitrarily selected sequences from tmRNA and extracted PK1, PK2, and PK4 from them.
- The negative data population was driven from the Rfam database [10]. We selected non-pseudoknot families taking into consideration that the lengths of these sequences would be in the same range as the positive population.

The size of each population was 500 sequences. We randomly divided each of the data populations to three equal subsets: Training set, test set 1 and test set 2. Table 2 lists the sensitivity and specificity for each subset and for the whole population. Table 3 lists the sensitivity and specificity of TAG_{RNA}INF, TAG_{RNA} [26] and PknotsRG (mfe) [18] when applied to Test set 1. For TAG_{RNA}, and PknotsRG (mfe), we count TP to be the number of sequences belonging to the positive population with predicted structures exhibiting a pseudoknot. On the other hand, TN is the number of sequences belonging to the negative population with predicted structures not exhibiting a pseudoknot.

Results in table 2 indicate that our approach is solid and can result in very accurate predictions. The same problem has been addressed in [14] using a different grammatical formalism. However, the sensitivity we achieve is superior to that reported in [14]. For instance when the size of the training set is 50% of the available sample, they can achieve a sensitivity of only 54%. To achieve a sensitivity of 85%, they have to employ a training set of size 90% of the sample. They do not report specificity results. Results in table 3 indicate that our approach achieve a good balance between sensitivity and specificity.

Table 2. Experimental Results for TAG_{RNA}INF

Data Subset	Sensitivity	Specificity
Training set	87.4%	84.4%
Test set 1	78.4%	80.8%
Test set 2	79.6%	88%
Whole Population	81.8%	84.4%

Table 3. Comparative Results for Test set 1

	Sensitivity	Specificity
TAG_{RNA}INF	78.4%	80.8%
TAG_{RNA}	100%	71.3%
PknotsRG (mfe)	41.6%	81.4%

5 Conclusion

In this paper we have presented a grammatical inference algorithm for $TAG_{RNA,}$. We used the inference algorithm as a module within a complete RNA structure identification framework, TAG_{RNA}Inf, capable of identifying pseudoknot structures. The TAG parser used within TAG_{RNA}Inf utilizes a scoring function along with the inferred grammar. The scoring function currently used is the number of base pairs of the RNA structure detected by the parser. For a training set and a test set of equal size, our experimental results outperforms those reported in [14] for the same problem. They use a different grammatical model.

References

1. Al Seesi, S.: Pseudoknot Identification through Learning TAG_{RNA}, BECAT-CSE Technical Report, University of Connecticut (April 2008)
2. Akutsu, T.: Dynamic Programming Algorithms for RNA Secondary Structure Prediction with Pseudoknots. Discrete Applied Mathematics 104, 45–62 (2000)
3. Ambros, V., Bartel, B., Bartel, D.P., Burge, C.B., Carrington, J.C., Chen, X., Dreyfuss, G., Eddy, S.R., Griffiths-Jones, S., Marshall, M., Matzke, M., Ruvkun, G., Tuschl, T.: A Uniform System for microRNA Annotation. RNA 9(3), 277–279 (2003)
4. van Batenburg, F.H.D., Gultyaev, A.P., Pleij, C.W.A., Ng, J., Oliehoek, J.: Pseudobase: a Database with RNA Pseudoknots. Nucl. Acids Res. 28(1), 201–204 (2000)
5. Brazma, A., Jonassen, I., Vilo, J., Ukkonen, E.: Pattern Discovery in Biosequences. In: Honavar, V., Slutzki, G. (eds.) ICGI 1998. LNCS (LNAI), vol. 1433, pp. 255–270. Springer, Heidelberg (1998)
6. Buratti, E., Dhir, A., Lewandowska, M.A., Baralle, F.E.: RNA Structure is a Key Regulatory Element in Pathological ATM and CFTR Pseudoexon Inclusion Events. Nucl. Acids Res. 35(13), 4369–4383 (2007)
7. Cai, L., Malmberg, R., Wu, Y.: Stochastic Modeling of RNA Pseudoknotted Structures: a Grammatical Approach. Bioinformatics 19(supp. 1), 66–73 (2003)
8. Dirks, R.M., Pierce, N.A.: A Partition Function Algorithm for Nucleic Acid Secondary Structure Including Pseudoknots. J. Comput. Chem. 24(13), 1664–1677 (2003)
9. Gilbert, W.: The RNA World. Nature 319, 618 (1986)
10. Griffiths-Jones, S., Moxon, S., Marshall, M., Khanna, A., Eddy, S.R., Bateman, A.: Rfam: Annotating Non-coding RNAs in Complete Genomes. Nucl. Acids Res. 33, D121–D124 (2005)
11. Holbrook, S.R.: RNA Structure: the Long and the Short of it. Current Opinion in Structural Biology 15, 302–308 (2005)
12. Joshi, A.K., Levy, L., Takahashi, M.: Tree Adjunct Grammars. Journal of Computer and System Sciences 10, 136–163 (1975)
13. Laxminarayana, J.A., Nagaraja, G., Balaji, P.V.: Identification of Pseudoknots in RNA Secondary Structures: A Grammatical Inference Approach. In: Mukherjee, D.P., Pal, S. (eds.) Proceedings of 5th International Conference on Advances in Pattern Recognition (2003)
14. Laxminarayana, J.A., Nagaraja, G., Balaji, P.V.: Inference of a Subclass of Even Linear Languages and its Application to Pseudoknot Identification. In: Department of Computer Science and Engineering, Indian Institute of Technology, Bombay, India (manuscript, 2003)

15. Paillart, J.C., Skripkin, E., Ehresmann, B., Ehresmann, C., Marquet, R.: In vitro Evidence for a Long Range Pseudoknot in the 5'-Untranslated and Matrix Coding regions of HIV-1 Genomic RNA. J. Biol. Chem. 277, 5995–6004 (2002)
16. Pedersen, J.S., Bejerano, G., Siepel, A., Rosenbloom, K., Lindblad-Toh, K., Lander, E.S., Kent, J., Miller, W., Haussler, D.: Identification and Classification of Conserved RNA Secondary Structures in the Human Genome. Public Library of Science. Computational Biology 2(4), 33 (2006)
17. Rajasekaran, S.: Tree-Adjoining Language Parsing in $o(n^6)$ Time. SIAM Journal on Computing 25(4), 862–873 (1996)
18. Reeder, J., Giegerich, R.: Design, Implementation and Evaluation of a Practical Pseudoknot Folding Algorithm Based on Thermodynamics. BMC Bioinformatics 5, 104 (2004)
19. Rivas, E., Eddy, S.: The Language of RNA: a Formal Grammar that Includes Pseudoknots. Bioinformatics 16(4), 334–340 (2000)
20. Robertson, M.P., Igel, H., Baertsch, R., Haussler, D., Ares Jr., M., Scott, W.G.: The Structure of a Rigorously Conserved RNA Element within the SARS Virus Genome. Public Library of Science: Biology 3(1), 5 (2004)
21. Sakakibara, Y., Brown, M., Hughey, R., Mian, I.S., Sjolander, K., Underwood, R.C., Haussler, D.: Stochastic Context-Free Grammars for tRNA Modeling. Nucl. Acids Res. 22, 5112–5120 (1994)
22. Sakakibara, Y.: Grammatical Inference in Bioinformatics. IEEE Transactions on Pattern Analysis and Machine Intelligence 27, 1051–1062 (2005)
23. Searls, D.: The Linguistics of DNA. Am. Scient. 80, 579–591 (1992)
24. Takakura, T., Asakawa, H., Seki, S., Kobayashi, S.: Efficient Tree Grammar Modeling of RNA Secondary Structures from Alignment Data. In: Proceedings of posters of RECOMB 2005, pp. 339–340 (2005)
25. Tanaka, Y., Hori, T., Tagaya, M., Sakamoto, T., Kurihara, Y., Katahira, M., Uesugi, S.: Imino Proton NMR Analysis of HDV Ribozymes: Nested Double Pseudoknot Structure and Mg2+ Ion-Binding Site Close to the Catalytic Core in Solution. Nucl. Acids Res. 30, 766–774 (2002)
26. Uemura, Y., Hasegawa, A., Kobayashi, S., Yokomori, T.: Tree Adjoining Grammars for RNA Structure Prediction. Theoretical Computer Science 210(2), 277–303 (1999)
27. Vijay-Shanker, K., Joshi, A.K.: Some Computational Properties of Tree Adjoining Grammars. In: 23 rd Meeting of the Association for Computational Linguistics, pp. 82–93 (1985)
28. Williams, K.P., Bartel, D.P.: The tmRNA Website. Nucl. Acids Res. 26(1), 163–165 (1998)
29. Williams, K.P.: The tmRNA Website: Invasion by an Intron. Nucl. Acids Res. 30(1), 179–182 (2002)

Support Vector Based T-Score for Gene Ranking

Piyushkumar A. Mundra[1] and Jagath C. Rajapakse[1,2,3]

[1] Bioinformatics Research Center, School of Computer Engineering,
Nanyang Technological University, Singapore
[2] Singapore-MIT Alliance, Singapore
[3] Department of Biological Engineering,
Massachusetts Institute of Technology, USA
asjagath@ntu.edu.sg

Abstract. T-score between classes and gene expressions is widely used for gene ranking in microarray gene expression data analysis. We propose to use only support vector points for computation of t-scores for gene ranking. The proposed method uses backward elimination of features, similar to Support Vector Machine Recursive Feature Elimination (SVM-RFE) formulation, but achieves better results than SVM-RFE and t-score based feature selection on three benchmark cancer datasets.

1 Introduction

Simultaneous measurement of thousands of genes has become possible due to recent advances in DNA microarray technology. Unfortunately, due to high cost of experiments, sample sizes are still very small compared to the number of genes measured. Because of this bottleneck, curse of dimensionality and computational instabilities occur in microarray data analysis, which make it difficult to efficiently extract useful information. To overcome such problems, selection of relevant genes has became extremely important in microarray data analysis.

Various gene selection approaches have been recently proposed by different research groups [1,2,3,4,5,6,7,8,9,10,11]. Gene selection methologies can be broadly classified into two methods: filter methods and wrapper methods [2]. Filter methods evaluate gene subsets by looking at intrinsic characteristics of data with respect to class labels [1]. T-score, P-score, mutual information, euclidean distance, and correlation coefficients are some of the widely used filter criterions [2]. In wrapper approach, the goodness of gene subset is evaluated by estimating the accuracy and the selection is embedded in the specific learning method. Wrapper methods are better in principle but more complex and computationaly expensive. Various algorithms have been developed for gene ranking based on SVM [9,10,12]. Support vector machine - recursive feature elimination (SVM-RFE) is one of the widely used wrapper method [12]. SVM-RFE is a multivariate gene ranking method which uses SVM classifier for ranking. SVM-RFE has also been applied to peak selection of mass spectrometry data for cancer classification [13]. Recently, we proposed a linear combination of SVM-RFE with minimum redundancy maximum relevancy based filter criteria to minimize between gene redundancy without affecting classification performance [11].

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 144–153, 2008.

In filter approach, the standard practice is to consider all the sample points into gene ranking. But, the success of SVM in classification with its excellent generalization capability has proved that only boundary points are important for classification with an optimal margin. We propose a novel method for gene ranking by incorporating t-score in SVM-RFE based ranking to analyze support vector points. In this paper, we investigate the effect of t-score based ranking on classification performance while considering only support vector points. Proposed t-score gene ranking method is formulated in a backward elimination manner as removal of genes from dataset changes support vector points. As seen later, the proposed method showed better performance compared to t-score based or SVM-RFE method on benchmark datasets.

This manuscript is organized as follows: In section 2, we describe the SVM-RFE method and a detailed description of proposed method. Numerical experimental procedures and results are discussed in section 3. Finally, section 4 includes the discussion and conclusion.

2 Method

Let $D = \{x_{ij} : i = 1, 2, ..., n; j = 1, 2, ...m\}$ denotes the microarray gene expression dataset where x_{ij} is the expression measurement of ith gene in jth sample, n represents the total number of measured genes and m denotes the total number of samples. Let $x_j = (x_{1j}, x_{2j}, \ldots, x_{nj})$ be the gene expressions measured in the jth sample. In this paper, we address two class classification of tissue samples in to cancer or benign samples. Let the target class label of jth sample be $y_j \in \{+1, -1\}$ taking values +1 and -1 for being benign and cancerous tissues, respectively.

2.1 Support Vector Machine Recursive Feature Elimination(SVM-RFE)

The objective function for the Support Vector Machines maximize the margin of separation between two classes [14]. The soft-margin SVM is obtained by,

$$\max_\alpha W(\alpha) = \sum_{k=1}^{m} \alpha_k - \frac{1}{2} \sum_{k=1}^{m} \sum_{l=1}^{m} \alpha_k \alpha_l y_k y_l K(x_k, x_l) \tag{1}$$

$$\text{subject to } 0 \leq \alpha_k \leq \zeta, \text{for all } k = 1, \ldots, m \tag{2}$$

$$\text{and } \sum_{k=1}^{m} \alpha_k y_k = 0 \tag{3}$$

where $\{(x_k, y_k) : k = 1, 2, \ldots, m\}$ denotes the training examples. Here, ζ is SVM sensitivity parameter, $K(.,.)$ the Kernel function, and α_k is a parameter obtained by training SVM. SVM formulation only depends on the support vectors to define boundaries as parameters α_k is non-zero only for support vector points.

SVM-RFE technique was developed to rank genes for cancer classification [12]. In SVM-RFE, starting with all genes in the subset, iteratively one can remove gene with least importance for sample classification, given by the weights. This SVM weight vector w is computed using α_k corresponding to support vector points as follows:

$$w = \sum_{k=1}^{m} \alpha_k y_k x_k \tag{4}$$

Support vectors denote data points on the boundaries of and within the separating margins. It can be shown that α_k are zero for non-support vector points. If w_i represents the corresponding component of above weight vector after normalization, the ith gene with smallest ranking score, ${w_i}^2$, is removed from the gene subset. For the computational efficiency, more than one feature can be removed at each step [12] though it may have negative effect on performance of feature selection method if a large portion of features are removed at a time.

2.2 T-Score Based Support Vector Backward Feature Elimination (SV-RFE)

Support vector points represent samples with $0 < \alpha_k \leq \zeta$, i.e., points either lie on the decision boundary or on the wrong side of the margin. In our method, we only concentrate on these points to compute the t-score. The non-support vector points need not be considered for gene ranking. This idea is based on SVM-RFE method where points only with $\alpha_k > 0$ are used for gene ranking.

Let M_+ and M_- subscripts represent set of support vector points corresponding to positive and negative samples. The ranking score for the proposed method is given by [2],

$$r_i = \frac{\left|\mu_{i,M_+} - \mu_{i,M_-}\right|}{\sqrt[2]{\frac{m_{M_+}\sigma^2_{i,M_+} + m_{M_-}\sigma^2_{i,M_-}}{m_{M_+}+m_{M_-}}}} \tag{5}$$

where μ_i and σ_i^2 represent mean and variance of expression values of gene i in respective support vector groups, (M_+ or M_-), m_{M_+} and m_{M_-} denote the number of positive and negative support vector points respectively.

T-statistics compare means of two sets of samples assuming equal variances for both sets. Gene which has higher t-score between the desired and undesired class labels is assumed to have higher class separability. The filter methods utilizing t-statistics have been proven successful in gene selection [1,2]. In standard t-test, all the sample points are considered for score computation. Refering to Eq. (5), instead of taking only M_+ and M_- points (which are support vector points), the previous t-statistics based methods use all points in positive and negative class to compute standard t-score [2].

The psuedocode for t-score based Support Vector Backward Feature Elimination (SV-RFE) is described in Algorithm 1.

Algorithm 1. T-score based Support Vector Backward Feature Elimination

Begin : Ranked gene set R = [ ], and gene subset $S = [1, 2, \ldots, n]$
repeat
Train linear SVM with gene set S in input variable
Obtain the support vector points and compute the ranking score r_i
Select the gene with smallest ranking score $e = \arg\min(r_i)$
Update $R = [e, R]\,; S = S - [e]$
until all genes are ranked
end : output R

Looking from different point of view, the proposed method has some resemblence to original SVM-RFE with certain assumptions. From Eq. (2), it is clear that $\alpha_k \leq \zeta$. After normalizing α vector, this constraint becomes $\alpha_k \leq 1$. Assuming all support vector points have $\alpha_k = 1$ and substituting it in Eq. (4), SVM-RFE weight becomes a simple summation of each gene's expression values. Instead of simple summation, we propose to use statistically more correct t-score based ranking. In a way, proposed method does not use α parameter obtained from SVM learning and in each iteration, model is trained to obtain optimum support vector points. Due to this, our method differs from SVM-RFE significantly. This algorithm is computationally expensive than standard t-score.

3 Experiments and Results

3.1 Data

To evaluate the performance of proposed t-score based SV-RFE method, we performed extensive experiments on three microarray gene expression datasets, namely, Colon [15], Leukemia [1], and Prostate [16] cancer dataset. These are widely used benchmark datasets to evaluate gene ranking methods. In Colon cancer, no separate testing set is available. Hence we divided the original dataset into separate training set and testing set. The number of samples and genes are given in Table 1.

3.2 Preprocessing

To obtain the support vector points, we normalized the training dataset to zero mean and unit variance based on gene expression of a particular gene. These continuous datasets were directly used in SVM-RFE after normalization.

Table 1. Sample Sizes of Three Gene-Expression Datasets

Dataset	# Training	# Testing	# Total Genes
Colon	40	22	2000
Leukemia	38	34	7129
Prostate	102	34	12600

For t-score computation, we use mean centered gene expression dataset (without shifting by unit variance). For t-score based method, we obtain support vector points using zero mean unit variance training data while t-score in each iteration was computed using corresponding sample points in mean centered original gene expression training set.

3.3 Parameter Estimation

Obtaining optimal support vector points is one of the key steps in the proposed method. This depends on sensitivity parameter η in case of linear SVMs. η values were chosen from finite set $\left\{2^{-20}, \ldots, 2^{0}, \ldots, 2^{15}\right\}$ using 10-fold cross-validation (CV). This set was also used for SVM-RFE and test performance evaluation.

CV error is generally employed by either, k-fold CV or *Leave-One-Out*. In present work, we use Matthew's Correlation Coefficient (MCC[1]) with 10-fold cross-validation for training performance evaluation and parameter tuning. MCC was choosen as the error measure because sample size was small and imbalanced in lables in most datasets.

To increase the speed of the numerical simulations with both SVM-RFE and proposed method, we employ following heuristic strategy:

$$\text{Number of genes removed} = \begin{cases} 100 \text{ if } n' \geq 10000 \\ 10 \quad \text{if } 1000 \leq n' < 10000 \\ 1 \quad n' < 1000 \end{cases} \tag{6}$$

where n' is the number of genes in the gene set.

3.4 Performance Evaluation

Ranking of genes in each dataset was obtained using simple t-score, SVM-RFE, and proposed method. Only training data was used to rank the genes using a linear SVM. Using the gene ranking list, we tested gene subsets starting from top ranked gene and then successively adding one gene at a time in testing subset till total number of genes in subset equals 100.

Small sample size in gene expression datasets present a peculiar problem while dividing into training and testing sets. It will not give correct performance evaluation if only one set of testing set is used. This is known as "unfortunate" partitioning of training and testing sets. To solve this "unfortunate" partitioning problem, we merge the training and testing datasets before testing. After that, we employ stratified sampling to partition the total samples into separate training and testing sets by maintaing number of samples in each set as before. Then, the classifier is trained on the training set and tested on the corresponding testing set. This process is followed for 100 times and performance measure such as, test accuracy, sensitivity and specificity were computed for these 100 trials. Finally, total number of genes required for best classification accuracy corresponds to subset with the least average test error.

[1] $MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$

Table 2. Performance of t-score, SVM-RFE and Proposed method on Various *Cancer* Datasets

Dataset	Measurement	T-score	SVM-RFE	Proposed Method
Colon	# Genes	95	90	**83**
	Accuracy	88.18 ± 5.29	91.00 ± 5.17	$\mathbf{91.14 \pm 5.22}$
	Sensitivity	82.50 ± 11.92	86.75 ± 10.18	$\mathbf{87.12 \pm 11.16}$
	Specificity	91.43 ± 6.09	$\mathbf{93.43 \pm 5.53}$	$\mathbf{93.43 \pm 5.71}$
Leukemia	# Genes	88	**47**	64
	Accuracy	96.88 ± 3.44	97.88 ± 2.07	$\mathbf{98.41 \pm 1.79}$
	Sensitivity	92.64 ± 8.40	95.00 ± 5.13	$\mathbf{96.21 \pm 4.24}$
	Specificity	99.85 ± 1.11	99.90 ± 0.70	$\mathbf{99.95 \pm 0.50}$
Prostate	# Genes	85	85	**21**
	Accuracy	93.41 ± 3.79	96.24 ± 3.37	$\mathbf{97.18 \pm 2.89}$
	Sensitivity	92.84 ± 4.93	95.88 ± 4.08	$\mathbf{96.88 \pm 3.49}$
	Specificity	95.00 ± 7.80	97.22 ± 5.56	$\mathbf{98.00 \pm 4.57}$

Table 3. Comparison of accuracies with the published results

Method/Dataset	Colon		Leukemia		Prostate	
	Accuracy	# of Genes	Accuracy	# of Genes	Accuracy	# of Genes
Bayes + KNN [8]	90.32	6	100.00	3	94.12	11
Bayes + SVM [8]	87.10	20	**100.00**	**2**	96.08	13
t-test + Fisher Classifier [19]	88.30	...	88.00	...	92.00	...
MMC-RFE + NMC [20]	88.80	100	99.20	100	90.10	**10**
Proposed Method + SVM	**91.14**	83	98.41	64	**97.18**	21

We also compared the results with SVM-RFE method. This method was performed in exactly the same way as that of proposed method except ranking criteria. In all gene selection methods and testing the classifier, we used LIBSVM - 2.84 software [17].

3.5 Results

The proposed method has remarakably good performance than t-score method in all three gene expression dataset. Both sensitivity and specificity are improved in all datasets. Figures 1,2, and 3 represent the average test misclassification error rate in each of the three datasets. Also, except Prostate Cancer dataset, our method needed less number of genes for classification compared to t-score method. The proposed method also have comparable performance with SVM-RFE method.

Table 3 shows a comparison of classification accuracy with other methods available in the literature. As seen in the table, our method performed reasonably well in all three datasets. Classification performance is much better in Prostate cancer dataset. In Leukemia dataset, out method is inferior to Leave one out

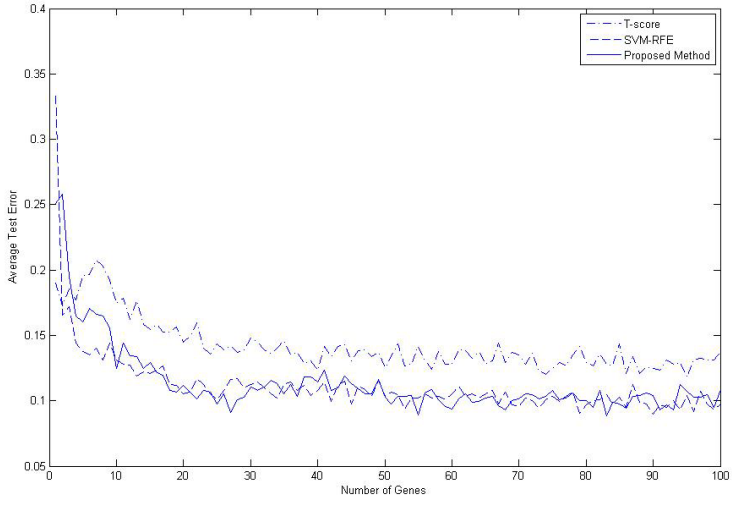


Fig. 1. Average misclassification error rate for all three methods on Colon Cancer Dataset against the number of genes

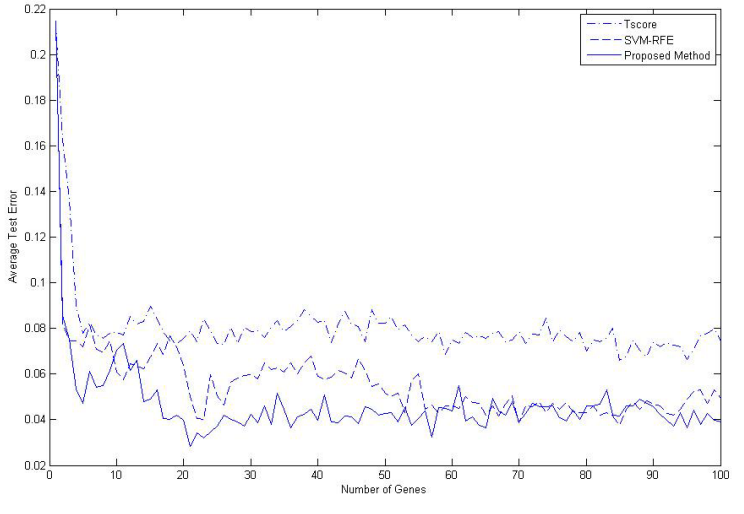


Fig. 2. Average misclassification error rate for all three methods on Prostate Cancer Dataset against the number of genes

(LOO) method but better than both 10-fold and 100-split testing. As compared and discussed in [18], LOO gives optimistic accuracy estimations compared to both k-fold cross validation and bootstrap method.

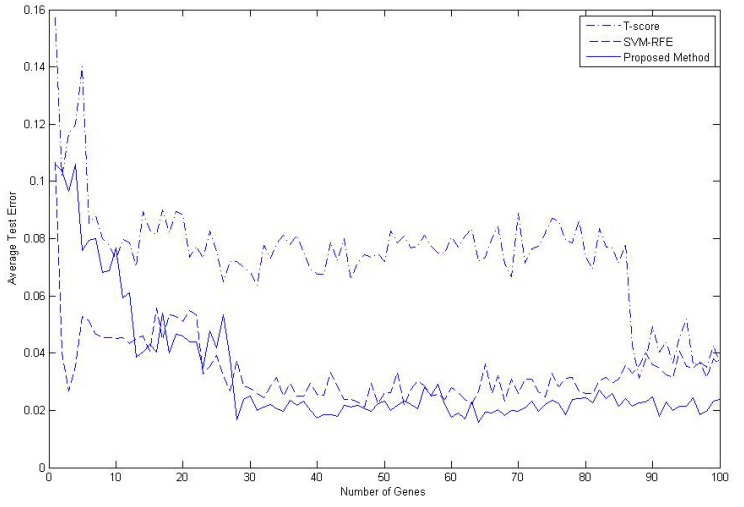


Fig. 3. Average misclassification error rate for all three methods on Leukemia Cancer Dataset against the number of genes

4 Discussion

We propose a support vector based t-score method for gene ranking. We evaluated performance of the proposed method on three benchmark datasets and showed remarkable improvement in accuracy compare to standard t-score. Performance results are quite comparable to SVM-RFE.

In practice, standard t-score based approach considers all the data points in the training set. But as shown in SVM based classification, only the data points which lie on the boundary are important for decision making. Based on success of such strategy, our approach only considers data points obtained from SVM model. Because of only considering support vector points, statistically we lose some degree of freedoms. But as shown in the results, only concentrating on support points improves the classification performance.

Removal of one gene can change support vector points, and hence t-score will change. To incorporate such effect, we use backward elimination based SVM-RFE approach with t-score criteria in gene ranking. This approach is different from standard t-score method where all the genes are ranked in one iteration.

We would like to reemphasize that the proposed method does not use α parameter obtained from SVM models. As discussed in the methods section, if α value is assumed to be 1 for all support vector points, SVM-RFE weight criteria is simple summation of gene expression values. In the proposed method, we use statistical t-score, which ranks genes based on mean and variance of gene expression values in cancerous and benign tissue samples. This results improved the classification performance. Only similarity with SVM-RFE is that, in each iteration, specified numbers of genes were removed and new t-score was calculated

for new SVM model. As number of support vector points change in each iteration, and hence mean and variance of gene, our method formulation indirectly changes univariate t-score into multivariate system. It would be interesting to see if same hypothesis of using only support vectors can be applied with other filter criteria.

In conclusion, we proposed a novel support vector based t-score computation in SVM-RFE formulation. Extensive testing on three benchmark cancer classification gene-expression dataset revealed that proposed method perfoms significantly better than standard t-score appraoch and results are comparable with SVM-RFE.

References

1. Golub, T., Slonim, D., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J., Coller, H., Loh, M., Downing, J., Caligiuri, M., Bloomfield, C., Lander, E.: Molecular classification of cancer: Class discovery and class prediction by gene expression. Science 286, 531–537 (1999)
2. Inza, I., Larranaga, P., Blanco, R., Cerrolaza, A.: Filter versus wrapper gene selection approaches in DNA microarray domains. Artificial Intelligence Medicine 31, 91–103 (2004)
3. Battiti, R.: Using mutual information for selecting features in supervised neural net learning. IEEE Trans Neural Network 5, 537–550 (1994)
4. Liu, X., Krishnan, A., Mondry, A.: An entropy-based gene selection method for cancer classification using microarray data. BMC Bioinformatics 6, 76 (2005)
5. Ding, C., Peng, H.: Minimum redundancy feature selection from microarray gene expression data. J. Bioinformatics Computational Biology 3, 185–205 (2005)
6. Peng, H., Long, F., Ding, C.: Feature selection based on mutual information: Criteria of max-dependency, max-relevance, and min-redundancy. IEEE Trans. Pattern Analysis Machine Intelligence 27, 1226–1237 (2005)
7. Ooi, C., Chetty, M., Teng, S.: Differential prioritization between relevance and redundancy in correlation-based feature selection techniques for multiclass gene expression data. BMC Bioinformatics 7, 320–339 (2006)
8. Zhang, J., Deng, H.: Gene selection for classification of microarray data based on bayes error. BMC Bioinformatics 8, 370 (2007)
9. Rakotomamonjy, A.: Variable selection using svm criteria. J. Machine Learning Research (Special Issue on Variable Selection) 3, 1357–1370 (2003)
10. Kai-Bo, D., Rajapakse, J., Wang, H., Azuaje, F.: Multiple SVM-RFE for gene selection in cancer classification with expression data. IEEE Trans. Nanobioscience 4, 228–234 (2005)
11. Mundra, P., Rajapakse, J.: SVM-RFE with relevancy and redundancy criteria for gene selection. In: Rajapakse, J., Schmidt, B., Volkert, L.G. (eds.) PRIB 2007. LNCS (LNBI), vol. 4774, pp. 242–252. Springer, Heidelberg (2007)
12. Guyon, I., Weston, J., Barhill, S., Vapnik, V.: Gene selection for cancer classification using support vector machines. Machine Learning 46, 389–422 (2002)
13. Rajapakse, J., Kai-Bo, D., Yeo, W.: Proteomic cancer classification with mass spectrometry data. American. J. Pharmacogenomics 5, 281–292 (2005)
14. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer, Heidelberg (2001)

15. Alon, U., Barkai, N., Notterman, D., Gish, K., Ybarra, S., Mack, D., Levine, A.: Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. PNAS 96, 6745–6750 (1999)
16. Singh, D., Febbo, P., Ross, K., Jackson, D., Manola, J., Ladd, C., Tamayo, P., Renshaw, A., D'Amico, A., Richie, J., Lander, E., Loda, M., Kantoff, P., Golub, T., Sellers, W.: Gene expression correlates of clinical prostate cancer behavior. Cancer Cell 1, 203–209 (2002)
17. Chang, C., Lin, C.: Libsvm: A library for support vector machines (2001), www.csie.ntu.edu.tw/~cjlin/libsvm
18. Azuaze, F.: Genomic data sampling and its effect on classification performance assessment. BMC Bioinformatics 4, 5 (2003)
19. Lai, C., Reinders, M., van't Veer, L., Wessels, L.: A comparison of univariate and multivariate gene selection techniques for classification of cancer datasets. BMC Bioinformatics 7, 235 (2006)
20. Niijima, S., Kuhara, S.: Recursive gene selection based on maximum margin criterion: a comparison with svm-rfe. BMC Bioinformatics 7, 543 (2006)

Prediction of Transcription Factor Families Using DNA Sequence Features

Ashish Anand[1], Gary B. Fogel[2], Ganesan Pugalenthi[1], and P.N. Suganthan[1]

[1] School of Electrical and Electonic Engineering, Nanyang Technological University, 50 Nanyang Avenue, 639798, Singapore
{ashi0005,ganesan,epnsugan}@ntu.edu.sg
[2] Natural Selection, Inc. 9330 Scranton Road, Suite 150, San Diego, CA 92121
gfogel@natural-selection.com

Abstract. Understanding the mechanisms of protein-DNA interaction is of critical importance in biology. Transcription factor (TF) binding to a specific DNA sequence depends on at least two factors: A protein-level DNA-binding domain and a nucleotide-level specific sequence serving as a TF binding site. TFs have been classified into families based on these factors. TFs within each family bind to specific nucleotide sequences in a very similar fashion. Identification of the TF family that might bind at a particular nucleotide sequence requires a machine learning approach. Here we considered two sets of features based on DNA sequences and their physicochemical properties and applied a one-versus-all SVM (OVA-SVM) with class-wise optimized features to identify TF family-specific features in DNA sequences. Using this approach, a mean prediction accuracy of ~80% was achieved, which represents an improvement of ~7% over previous approaches on the same data.

Keywords: Transcription factor family prediction, multi-class classification.

1 Introduction

Protein-DNA interactions play a central role in many cellular processes including transcription and translation. A key aspect of transcriptional regulation requires the binding of a class of proteins (called transcription factors (TFs)) to cis-acting DNA regulatory sequences (known as transcription factor binding sites (TFBS)). Understanding the mechanisms of these interactions and identifying associations between each TF and DNA regulatory elements are key challenges for experimental and computational biology.

TFBS are usually very short (≤12 base pairs) [1] and some proteins are capable of binding to many TFBSs. Binding of a TF to the appropriate TFBS depends on two factors: A three-dimensional protein structure of the TF that presents an appropriate DNA-binding domain and the specific sequence of nucleotides recognized by TF. Though variability in TFBSs does exist, TFBSs share enough similarity such that they can be easily recognized in the nucleus by TF proteins. TFs can be classified into families based on these protein and DNA-binding characteristics and catalogs of

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 154–164, 2008.

TFBS and TF proteins can be found including JASPAR [2], Transfac [3]. Transfac uses the sequence similarities in TFBS as a basis of classification of TFs, whereas JASPAR uses the binding profiles (discussed below) to classify TFs into families.

Several approaches to identify TFBS and associate them with the binding TF exist. These include phylogenetic footprinting [4], position-specific scoring matrix (PSSM)-based approaches [5], Gibbs sampling [6], and expectation-maximization [7]. Phylogenetic footprinting has been applied to genomic sequences to identify novel TFBSs [8]. Clustering the results of phylogenetic footprint analysis on sets of co-regulated genes can result in novel TFBS as well as the identification and annotation of previously identified TFBSs [9]. Approaches based on comparative genomics or phylogenetic footprinting require genomic sequences from several species and as a result are computationally intensive. In some cases there may not simply exists sufficient representation over the phylogenetic history to generate meaningful comparisons at the species level.

A PSSM or position-weight matrix (PWM) is used commonly as probabilistic representation of a TFBS. These matrices store frequencies of each nucleotide at each position of the binding site. Such models generally assume independence between nucleotides over all positions must use a fixed-length (typically arbitrary) TFBS, are unable to represent sequence properties such as sequence-dependent physicochemical properties [10].

These methods are used generally to identify TFBS in newly sequenced data and do not attempt to predict a putative TF for each identified binding site.

Recently, several approaches have been proposed to handle this problem. Narlikar and Hartemink [11] used sparse multinomial logistic regression (SMLR) [12], to predict TF family given a set of TFBSs. For a given set of DNA sequences, a set of nucleic-acid based sequence features were generated. These features were used to generate the model and predict the TF family.

Sandelin and Wasserman [2] used binding sites profiles to classify well-characterized TFs into "familial binding profiles." The database JASPAR was generated using such "familial binding profiles" with corresponding TFs and associated binding sites. First a collection of PSSM models for a TF were assembled and similarities between models were calculated. Finally, an assembly algorithm was used to compile all models into a single familial binding profile. Using this approach, TF binding sites were classified into 11 families. A brief description of the families examined in this study is given in the Materials and Methods section below. Prediction of TF-family for a given set of DNA binding sites can be accomplished using "familial profiles."

Tan et al. [13] utilized the information of comparative genomics to connect TFs with their corresponding TFBSs. Three mutually independent information methods were used to connect a DNA binding motif to a given TF. Comparative analysis of multiple genomes was used to generate two of these sources of information and the third was derived from similarities of TFBS interactions. For a given TF and DNA motif, the three types of information were combined to obtain the probability that such a pair was a true pair.

Narlikar and Hartemink [11] showed that the nucleic-acid based features of TFBS can be used to predict the families of corresponding binding TF. They demonstrated that the selected features are family-specific. Motivated by these results, we used

sequence-based conformational and physicochemical features [10, 14] in addition to the features proposed by Narlikar and Hartemink [11] to develop models with an SVM-based classifier [15].

The results of this approach were compared directly to SMLR [11]. The addition of physicochemical features to the nucleic-acid based features led to significant improvement in predictive accuracy. The SVM-based classifier outperformed SMLR when only the nucleic-acid based features were used. Both SVM-based classifier and SMLR resulted in competitive predictive accuracies using additional set of physicochemical features.

2 Materials and Methods

2.1 Datasets

JASPAR is the largest, curated, and open-access collection of eukaryotic TFBS profile matrices [16]. TFBSs are classified in JASPAR into the 11 structural families shown in Table 1. As part of our experimental design, we only made use of those TFBS classes with 4 or more samples (see below). Given this requirement, two TFBS-families (bZIP-cEBP and TRP (MYB)) were removed (Table 1). The remaining 55 TFs from 9 TFBS-families were used for modelling. These families are briefly described below.

ETS Family: TFs belonging to this family contains a region of 85-90 AAs known as the erythroblast transformation specific (ETS) domain. This domain is quite rich in positively-charged and aromatic residues. The ETS domain binds to purine-rich segments of DNA [17].

bZIP-CREB Family: cAMP responsive element binding proteins (bZIP/CREB) are conserved, nuclear, bZIP-domain, dimeric transcription factors. TFs of this family

Table 1. TF families of JASPAR database. Abbreviations for some families are provided in square brackets.

TF Family	Number of Samples	Considered in this Study
ETS	7	Yes
bZIP-CREB	4	Yes
REL	5	Yes
Nuclear Receptor [NR]	8	Yes
Forkhead [Fkh]	4	Yes
bZIP-cEBP	3	No
bHLH (zip)	9	Yes
MADS	5	Yes
TRP (MYB)	3	No
Homeobox [Hbox]	7	Yes
HMG	6	Yes
Total Samples	61	55

contact the DNA through a basic region generally found in the amino-terminus of the TF. They contain leucine zipper segments consisting of leucine or similar hydrophobic AA spaced roughly every 7 or 8 residues.

REL Family: The Rel homology domain is found mainly in eukaryotic TFs. TFs containing the domain do not use well-defined secondary structure for DNA-binding [18]. The domain is composed of two immunoglobulin-like beta barrel sub-domains which grips the DNA in major groove.

Nuclear Receptor: The DNA-binding domain of nuclear receptors is composed of two zinc finger motifs that differ in size, composition, and function. Each finger contains four cysteine residues coordinating one zinc ion. The zinc coordinating motif is characterized by two anti-parallel alpha-helices capped by loops at their amino-terminal ends. Normally TFs of this class function as homo- or heterodimers. Each monomer typically consists of ligand-binding, DNA-binding, and transcription regulatory domains.

Fork head: The fork head domain contains neither homeodomains nor zinc-finger characteristics of other TFs. It contains a distinct type of DNA binding region of around 100 AAs and binds B-DNA as monomer.

bHLH (zip): TFs of this family contain a tripartite DNA binding domain consisting of a basic region, a helix-loop-helix (HLH), and a leucine zipper. The domain mediates dimerization as a prerequisite for DNA-binding. The basic region dictates DNA-binding specificity. The leucine zipper consists of repeated leucine residues at every seventh position.

MADS: The MADS box is a highly conserved sequence motif found in a family of TFs. The conserved domain was recognized after the first four members of the family, MCM1, AGAMOUS, DEFICIENS, and serum response factor (SRF) and named after them by taking their initials. TFs belonging to this class function as dimers. The primary DNA-binding element is an anti-parallel coiled coil of two amphipathic α-helices, one from each subunit. The MADS domain is a 56-residue motif consisting of a pair of anti-parallel coiled coil α-helices packed against an anti-parallel, double-stranded, β-sheet.

Homeobox: The homeodomain binds through a helix-turn-helix (HTH) structure. HTH motifs are characterized by two α-helices, joined by a short turn. Protein-DNA contacts are conserved, especially those made by positions R3, R5, I47, Q50, N51 and M54 [19]. This domain binds to DNA both as monomer and dimer. Some proteins are capable of both.

HMG: Proteins of this class comprise a region of homology with HMG proteins such as HMG1. Generally HMG domains bind DNA to non-sequence-specific manner. The domain exhibits an L-shaped configuration by 3 alpha helices. The 1st and 2nd helices contact DNA and the 3rd helix is exposed to solvents.

2.2 Feature Formulation

Nucleic acid-based sequence features were used to represent each TF. Two main sets of features were defined: DNA features and DNA-Physico features (described in

greater detail below). Features corresponding to DNA were calculated using only known binding sites or DNA motifs obtained from the JASPAR database. Flanking sequences were avoided given there is no general consensus on the notion of an "ideal" length for such flanking sequences in model development. Only the DNA-binding domain of TFs was used for feature calculation. For each TF, a list of binding DNA motifs or sites was obtained from the JASPAR database. Features corresponding to each DNA motif were calculated and average was taken to get single feature vector representing each TF. All features and their combinations are described in greater detail below.

1. DNA Features: We used the same set of features as discussed in [11]. These features included:

   A) The frequency of subsequence features representing the counts of all subsequences of length 1 to 5 in each TFBS. Only the four nucleotides A, T, G, C were considered. The full 15-letter code was not considered as no consensus sequences in any form were taken as binding sites. 1,364 features were generated in this manner.
   B) Ungapped palindrome features: Binary variables representing the presence or absence of palindrome subsequence of half-length 3, 4, 5 or 6 spanning entire site and that of palindrome subsequence not spanning the whole length. Thus there were total 8 such binary features. It is important to mention here this set of features will be sparse. For example, the presence of subsequence of half-length 3 spanning entire site will make the rest 7 binary variables 0.
   C) Gapped palindrome features: The same as the above with one difference of possibility of gaps. Here gap indicates the insertion of some non-palindrome nucleotides exactly in the middle of two palindrome halves. Similar to the previous case, total count of such features was 8.
   D) Special features: Narlikar and Hartemink [11] identified 7 special sequence features from literature which are found to be over-represented in the binding sites of certain TF families. These seven features were G **.** **.** G, G **.** **.** G **.** **.** G, [GC] **.** **.** [GC] **.** **.** [GC], AGGTCA | TGACCT, CA **.** **.** TG, TGA **.** * TCA, and TAAT | ATTA. Here '**.**' means presence of any single nucleotide, '**.***' means presence of at least one nucleotide, [XY] means presence of one of the letters X or Y, and 'XYZ | ABC' means presence of one of the strings 'XYZ' or 'ABC'. The presence or absence of each of these features was used as additional features.

When concatenated, features 1A-D above resulted in a single feature vector of length 1,387. The classifier model using solely this feature vector was referred to as the "DNA-model".

2. DNA-Physico Features: Conformational and physicochemical properties have been shown to affect the activity of cis-regulatory DNA elements [10, 14]. The mean values of 38 conformational and physicochemical properties of di-nucleotides were downloaded from the Property subdirectory of the Activity database [20]. For a given DNA site $'S = s_1,s_2,...,s_i,..s_L'$ of length L a value representing each of the 38 features was calculated as follows:

$$X_q^i(S) = \frac{\sum_{j=1}^{L-1} P_q(s_j, s_{j+1})}{L} \quad (1)$$

where P_q is the qth property of dinucleotides (s_j, s_{j+1}). There are only 38 such properties and when these were combined with the DNA feature set above, a total of 1,425 features resulted. The classifier model based on this feature set was referred to as the "DNA-Physico model."

2.3 Model Description

Background of SVM, OVA-SVM and SVM-RFE (OVA-RFE). Support vector machines (SVMs) [21] belong to the family of margin-based classifiers and can often achieve superior classification performance when compared to other classification algorithms across many domains. SVMs were originally designed to solve binary classification problems. Several algorithms have extended binary SVMs for application in multi-class problems [22-26]. One-versus-all (OVA) is one such simple and early extension of SVM for multi-class problems [27].

SVM-recursive feature selection (SVM-RFE) [28] was originally proposed for binary classification problems. The method of SVM-RFE begins with the set of all features and selectively eliminates one feature at a time. Features are scored and ranked on squared coefficients w_j^2 $(j=1,2,....,p)$ of weight vector **w**. The feature with smallest w_j^2 is eliminated in each iterative step. The procedure is repeated until a predetermined number of features remain. This procedure can also be generalized to remove more than one feature per step [28]. SVM-RFE is also extended in OVA fashion by many researchers [29-31]. This extension is generally known as OVA-RFE.

Feature selection was performed in OVA-RFE fashion. Selected features were then used with corresponding OVA-SVM classifiers. Final class-prediction was made using probabilities scores obtained from OVA-SVMs. We have shown [15] that conversion of decision function values into probability scores increases predictive performance. Among the three methods of converting decision function values into probability scores that were evaluated previously [15]; Platt's approach [32] was determined to provide better or equivalent predictive accuracy over several data sets. Thus for the purpose of the current investigation, we used Platt's approach to convert the decision function values into probability scores.

2.4 Experimental Design

During pre-processing, for each feature type, redundant features with identical values over all samples were removed. The remaining features were normalized to [-1, 1]. Table 2 lists number of features before and after pre-processing step.

Table 2. Feature Statistics

Feature Type	Number of Features	# Features After Pre-processing
DNA	1387	1305
DNA-Physico	1425	1343

The performance of all models was assessed using *k*-fold external cross-validation (CV) following Ambroise and McLachlan [33] to provide an unbiased estimate of generalization error. CVs were performed 100 times to provide more reliable estimates of prediction accuracy. A linear kernel was used for the SVM and hence only one SVM parameter (*C*) required tuning. For each model, a range of *C* was evaluated $\{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1\}$. The model and *C* setting with best average CV (4-fold CV) accuracy over all 100 runs was selected as the most appropriate setting of *C*. For ``this purpose, all features were used and no feature selection was performed.

For feature selection, ¾ of the data were considered were used for the feature selection process and the best features were re-evaluated on the remaining ¼ of the data for performance. Average 4-fold accuracy was calculated. This procedure was repeated for 100 different stratified (i.e., the class-wise proportion in training set was kept the same as was in the whole set) partitions of 4-fold. Average CV accuracies over 100 runs were calculated to estimate the prediction accuracy of models. We started with all features and successively eliminated 1% of remaining features in each iteration of OVA-RFE until a minimum of 10 features were left.

For fair comparison to SMLR [11], we used the same normalized data as was used in OVA-experiments. Different values of the parameter λ were tried and the one giving best average 4-fold CV accuracy over 100 runs was reported. We used SMLR software [12] available from the http://www.cs.duke.edu/~amink/software/smlr/.

3 Results and Discussion

3.1 Comparison between SMLR and Class-Wise Optimized OVA-SVM

Table 3 lists prediction accuracies obtained by models using different feature-types. OVA-SVM approach performed significantly better than SMLR using only DNA features (*t*-test *p*-value = 1.65×10^{-43}), however there was no significant difference found between the two approaches when DNA-Physico features was used. The mean predictive accuracy of SMLR approach improved significantly (from 74.05% ± 3.34% to 81.91% ± 2.69%; t-test p-value = 1.68×10^{-44}) with use of DNA-Physico features.

Table 3. Performance comparison of SMLR and Class-wise optimized OVA-SVM

Type of Features	OVA-SVM (All-Feats)	OVA-SVM (Feat-Selection)	SMLR
DNA	81.44±3.04	**81.86±2.77** (773) 80.88±2.98 (200) 80.06±3.07 (35)	74.05±3.34 (λ=0.001)
DNA-Physico	81.99±3.07	**82.56±2.82** (471) 81.73±2.64 (60) 80.09 ±2.88 (30)	81.91±2.69 (λ= 1.0E-5)

3.2 DNA-Model and DNA-Physico-Model Features

The addition of physicochemical properties improved mean prediction accuracy, though not significantly for the OVA-SVM approach (t-test p-value = 0.20). Figure 1 compares the average error obtained by models using different number of DNA and DNA-Physico features. Feature selection did not lead to any significant improvement in mean predictive accuracy however; more parsimonious models could be obtained using far fewer features with similar mean accuracy. For example, using only 35 DNA features, a mean prediction accuracy of 80.06% ± 3.07% was generated. Similarly, using 30 DNA-Physico features generated a mean prediction accuracy of 80.09% ± 2.88%. To compare the best predictive accuracy obtained by the two models irrespective of the number of features used, the model using DNA features only obtained best accuracy of 81.86 (±2.77) by using 773 features per class and the best accuracy of 82.56 (±2.82) was obtained by the model using DNA-Physico features with 471 features per class (Table 3).

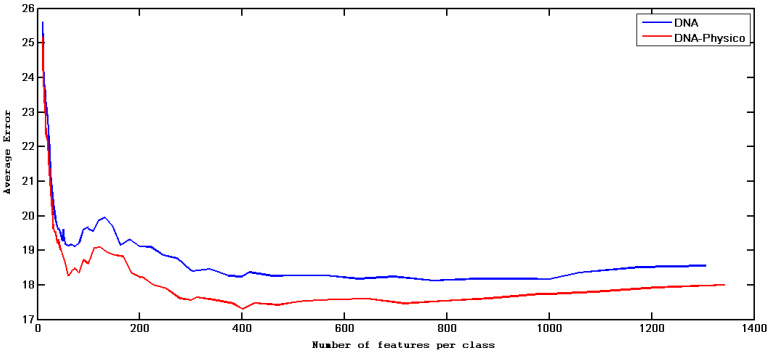


Fig. 1. Comparison between DNA and DNA-Physico Models

These results show that the model using DNA-Physico features was always able to provide slightly improved predictive accuracies than model using DNA features only. This suggests that the conformational and physicochemical features might influence the prediction of some of the TF families. We reviewed only the features which were selected more than 50% of the time by DNA-Physico models when using 30 features per TF family and separated these into conformational or physicochemical features (Table 4) to check our hypothesis. DNA-Physico features appeared to be important for the bHLH-ZIP family. We compared the prediction accuracy obtained by the two models for bHLH-ZIP family. The DNA-Physico model obtained an accuracy of 95.78% ± 6.27% whereas the DNA model obtained an accuracy of 90.56% ± 6.58%. The statistical significance of this difference was evaluated using a proportion test but no statistical significance was observed (p-value = 0.49) but reviewing the results as number of correct predictions made in each partitions shows a significant difference between the two models. The DNA-model predicts all 9 samples from the bHLH-ZIP family correctly only 26 times out of the 100 runs whereas the DNA-Physico model corrects all 9 samples of this family 66 times, for a difference of 40 out of 100 runs (Table 5).

Table 4. Class-wise statistics of different feature types: Features with frequency more than 200 were considered only. This statistics is obtained from the model using DNA-Physico Features with 30 features per class.

TF-Family	ETS	bZIP-CREB	REL	NR	Fkh	bHLH (zip)	MADS	Hbox	HMG
#DNA Features	28	22	31	26	27	19	27	23	19
#Physico Features	2	0	0	0	0	5	0	2	0

Table 5. Number of true classifications for bHLH-ZIP family by the two models in 100 partitions. Total number of samples in bHLH-ZIP was 9. Numbers in bracket in the first column indicates number of features used by OVA-SVM classifier.

# True Classifications	**7**	**8**	**9**
DNA-model (35)	11	63	26
DNA-Physico-model (30)	4	30	66

4 Conclusion

In this paper, features based on the TFBS sequences and their physico-chemical properties were used to build an OVA-SVM based multi-class classifier to predict the family of an associated binding TF protein. A detailed study was conducted to investigate the importance of different feature types for this decision. The performance of OVA-SVM based multi-class classifier and SMLR were compared and a significant improvement was found in the performance of SMLR when additional physico-chemical features were added to the nucleic-acid based features. While OVA-SVM outperformed SMLR based on only DNA-features, performance of the methods were competitive when DNA-Physico features were used.

Acknowledgments. Authors acknowledge the financial support offered by the A*Star (Agency for Science, Technology, and Research, Singapore) under the grant #052 101 0020.

References

1. Fogel, G.B., Weekes, D.G., Varga, G., Dow, E.R., Craven, A.M., Harlow, H.B., Su, E.W., Onyia, E., Su, C.: A Statistical Analysis of the TRANSFAC database. Biosystems 81(2), 137–154 (2005)
2. Sandelin, A., Wasserman, W.W.: Constrained Binding Site Diversity within Families of Transcription Factors Enhances Pattern Discovery Bioinformatics. J. Mol. Biol. 338, 207–215 (2004)

3. Matys, V., et al.: TRANSFAC: Transcriptional Regulation, from Patterns to Profiles. Nucleic Acids Res. 31, 374–378 (2003)
4. Blanchette, M., Tompa, M.: Discovery of Regulatory Elements by a Computational Method for Phylogenetic Footprinting. Genome Research 12(5), 739–748 (2002)
5. Stormo, G.: DNA Binding Sites: Representation and Discovery. Bioinformatics 16, 16–23 (2000)
6. Lawrence, C.E., Altschul, S.F., Boguski, M.S., Liu, J.S., Neuwald, A.F., Wootton, J.C.: Detecting Subtle Sequence Signals: A Gibbs Sampling Strategy for Multiple Alignment. Science, 208–214 (1993)
7. Bailey, T.L., Elkan, C.: Fitting a Mixture Model by Expectation Maximization to Discover Motifs in Biopolymers. In: Second International Conference on Intelligent Systems for Molecular Biology, pp. 28–36. AAAI Press, Menlo Park (1994)
8. McCue, L.A., Thompson, W., Carmack, C.S., Lawrence, C.E.: Factors Influencing the Identification of Transcription Factor Binding Sites by Cross-Species Comparis. Genome Res. 12, 1523–1532 (2002)
9. Qin, Z.S., McCue, L.A., Thompson, W., Mayerhofer, L., Lawrence, C.E., Liu, J.S.: Identification of Co-reguated Genes through Bayesian Clustering of Predicted Regulatory Binding Sites. Nature Biotechnology 21, 435–443 (2003)
10. Ponomarenko, J., Ponomarenko, M., Frolov, A., Vorobyev, D., Overton, G., Kolchanov, N.: Conformational and Physicochemical DNA Features Specific for Transcription Factor Binding Sites. Bioinformatics 15, 654–668 (1999)
11. Narlikar, L., Hartemink, A.J.: Sequence Features of DNA Binding Sites Reveal Structural Class of Associated Transcription Factor. Bioinformatics 22(2), 157–163 (2006)
12. Krishnapuram, B., Figueiredo, M., Carin, L., Hartemink, A.: Sparse Multinomial Logistic Regression: Fast Algorithms and Generalized Bounds. IEEE Transactions on Pattern Analysis and Machine Intelligence 27, 957–968 (2005)
13. Tan, K., McCue, L.A., Stormo, G.D.: Making Connections between Novel Transcription Factors and their DNA Motifs. Genome Res. 15, 312–320 (2005)
14. Anand, A., Fogel, G., Tang, E.K., Suganthan, P.N.: Feature Selection Approach for Quantitative Prediction of Transcriptional Activities. In: IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology, Toronto, Canada, pp. 57–62 (2006)
15. Anand, A., Pugalenthi, G., Suganthan, P.N.: Predicting Protein Structural Class by SVM with Class-wise Optimized Features and Decision Probabilities. Journal of Theoretical Biology (2008), doi:10.1016/j.jtbi.2008.02.031
16. Vlieghe, D., Sandelin, A., De Bleser, P.J., Vleminckx, K., Wasserman, W.W., Roy, F., Lenhard, B.: A New Generation of JASPAR, the Open-Access Repository for Transcription Factor Binding Site Profiles. Nucleic Acids Research 34, 95–97 (2006)
17. Karim, F.D., et al.: The ETS-domain: a new DNA-binding motif that recognizes a purine-rich core DNA sequence. Genes Dev. 4, 1451–1453 (1990)
18. Luscombe, N.M., Austin, S.E., Berman, H.M., Thornton, J.M.: An overview of the structures of protein-DNA complexes. Genome Biology. 1(1), reviews001.1—001.10 (2000)
19. Svingen, T., Tonnisen, K.F.: Hox Transcription Factors and their Elusive Mammalian Gene Targets. Heredity 97, 88–96 (2006)
20. Ponomarenko, J.V., Furman, D.P., Kolchanov, N.A., Sarai, A.: Activity: A database on DNA/RNA sites activity adapted to apply sequence-activity relationships from one system to another. Nucleic Acids Research 29(1), 284–287 (2001)
21. Vapnik, V.: Statistical learning theory. Wiley-Interscience, New York (1998)

22. Kreβel, U.H.-G.: Pairwise classification and support vector machines. In: Scholkopf, B., Burges, C.J.C., Smola, A.J. (eds.) Advances in kernel methods: Support vector learning, pp. 255–268. MIT press, Cambridge (1999)
23. Weston, J., Watkins, C.: Support vector machines for multiclass pattern recognition. In: Proc 7th European symposium on artificial neural networks (1999)
24. Crammer, K., Singer, Y.: On the algorithmic implementation of multiclass kernel-based vector machines. Journal of machine learning research 2, 265–292 (2001)
25. Hsu, C.-W., Lin, C.-J.: A comparison of methods for multi-class support vector machines. IEEE Trans. Neural Networks 13, 415–425 (2002)
26. Lee, Y., et al.: Multicategory support vector machines: theory and application to the classification of microarray data and satellite radiance data. J. Am. Stat. Assoc. 99, 67–81 (2004)
27. Bottou, L. et al.: Comparison of classifier methods: A case study in handwriting digit recognition. In: Proc. Int. Conf. Pattern Recognition. pp. 77-87 (1994)
28. Guyon, I., Weston, J., Barnhill, S., Vapnik, V.: Gene selection for cancer classification using support vector machines. Machine Learning 46, 389–422 (2002)
29. Chai, H., Domeniconi, C.: An evaluation of gene selection methods for multi-class microarray data classification. In: Scheffer, T. (ed.) Proceedings of the Second European Workshop on Data Mining and Text Mining in Bioinformatics, pp. 3–10 (2004)
30. Ramaswamy, S., Tamayo, P., Rifkin, R., Mukherjee, S., Yeang, C.H., Angelo, M., Ladd, C., Reich, M., Latulippe, E., Mesirov, J.P., et al.: Multiclass cancer diagnosis using tumor gene expression signatures. Proc. Natl. Acad. Sci. USA 98, 15149–15154 (2001)
31. Rifkin, R., Mukherjee, S., Tamayo, P., Ramaswamy, S., Yeang, C.-H., Angelo, M., Reich, M., Poggio, T., Lander, E.S., Golub, T.R., Mesirov, J.P.: An analytical method for multiclass molecular cancer classification. Siam Review 45(4), 706–723 (2003)
32. Platt, J.: Probabilistic outputs for support vector machines and comparison to regularized likelihood methods. In: Smola, A.J., Bartlett, P.L., Scholkopf, B., Schuumans, D. (eds.) Advances in Large Margin Classifiers, pp. 61–74. MIT Press, Cambridge (2000)
33. Ambroise, C., McLachlan, G.J.: Selection bias in gene extraction on the basis of microarray gene-expression data. Pro. Nat. Acad. Sci. 99, 6562–6566 (2002)

g-MARS: Protein Classification Using Gapped Markov Chains and Support Vector Machines

Xiaonan Ji, James Bailey, and Kotagiri Ramamohanarao

NICTA Victoria Laboratory
Department of Computer Science and Software Engineering
University of Melbourne, Australia
{xji,jbailey,rao}@csse.unimelb.edu.au

Abstract. Classifying protein sequences has important applications in areas such as disease diagnosis, treatment development and drug design. In this paper we present a highly accurate classifier called the g-MARS (gapped Markov Chain with Support Vector Machine) protein classifier. It models the structure of a protein sequence by measuring the transition probabilities between pairs of amino acids. This results in a Markov chain style model for each protein sequence. Then, to capture the similarity among non-exactly matching protein sequences, we show that this model can be generalized to incorporate gaps in the Markov chain. We perform a thorough experimental study and compare g-MARS to several other state-of-the-art protein classifiers. Overall, we demonstrate that g-MARS has superior accuracy and operates efficiently on a diverse range of protein families.

1 Introduction

With the development of genome sequencing techniques, biologists have accumulated huge numbers of protein sequences and new ones are being discovered daily. Predicting the class or the main function of a new protein sequence can assist experts in understanding its nature. It is a difficult problem, however, and it is not easy to advance the state of the art. Successful protein classifiers must be able to compare sequences efficiently, detect important features and also show good predictive capability.

A number of algorithms have been developed for classifying proteins into families or into clusters of functions or localizations. The basic assumption mostly used is the first fact of biological sequence analysis: "In biomolecular sequences (DNA, RNA or amino acid sequences), high sequence similarity usually implies significant functional or structural similarity." [7]. So, to create highly-accurate classifiers, we need a way to compare the similarity of a large number of diverse sequences precisely and efficiently.

Our contribution. In this paper, we describe a new protein classifier called the g-MARS (gapped Markov Chain with Support Vector Machine) classifier. The g-MARS approach has two main stages. Firstly, each protein sequence is

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 165–177, 2008.

individually modeled using what we call a "gapped markov chain", to capture its statistically important features. Next, a new dataset is derived from the collection of all gapped markov chains and it is passed to a support vector machine for decision making. The prime advantage of g-MARS is its superior accuracy compared to several existing protein classification methods. This is a claim validated in our experimental study, which considers a diverse range of protein families with different characteristics. The technique also scales well for large datasets. We first begin with a review of related work in the area.

Related work. Amino acid composition-based algorithms measure the similarity of proteins from the compositions of their amino acids. For each protein in the training dataset, the algorithm[6] calculates the frequency of each of its amino acids. For a new protein to be classified, its amino acid frequency histogram is calculated and compared with the compositions of the proteins in each class of training data. The protein is then classified to the class containing the protein with the smallest composition difference. The shortcomings of this approach are the loss of the ordering relationship among amino acids and the simplistic comparison in the composition difference. These compositions may be biased for small training datasets.

Amino acid composition with gaps [8] is an improvement of the pure amino acid composition algorithm [6]. The first improvement is that it considers pairs of the amino acids rather than individual ones. The second improvement is that it uses a support vector machine to make decisions, which is useful to alleviate the potential bias introduced by the limited information from the training datasets. The limitation is that the measurement is still based on the percentages of the pairs of amino acids among the whole protein sequence. When two proteins have different lengths, although they share some similar sections, certain amino acid pairs may have composition differences.

The spectrum kernel [11] is a support vector machine algorithm that calculates the similarity of two sequences by their common k-mers. In practice, the spectrum kernel works quite well [11]. However, there are limitations: it is far more computationally expensive than the amino acid composition algorithm. Secondly the choice of k in practice must be small, since the number of k-mers increase exponentially with k (so $k = 3$ is generally used). Thirdly, since k-mers must be contiguous, there can be less tolerance when proteins contain errors or mutations. In the mismatch kernel[10], the sharing of the similar k-mers, along with the identical ones, is used to measure the similarity.

Previous work by Wang et al [14] presents an interesting, but very general framework (GMM) for using markov models to classify proteins using amino acid feature combinations which may include gaps. Our g-MARS algorithm can roughly fit into this framework, but with a number of key differences: i) GMM requires the configuration of between six and ten different parameters and does not provide any general strategy for choosing them, a difficult challenge for a user . Thus it is better described as a large space of possible algorithms, rather than a single algorithm (and so it is not feasible to try to experimentally benchmark against), ii) Different combinations of features are used. Only the prior and

posterior pair with the highest order is used for classifying a protein by GMM. In g-MARS, however, we consider variable gaps and use all resulting prior-posterior pairs for the classification decision, iii) The GMM classification/decision model is essentially a set of prior-posterior pairs which work as rules and classification relies on aggregating scores of these rules. In contrast, g-MARS learns a classification model based on training a support vector machine.

The Fisher kernel [9] combines the support vector machine and the hidden markov model. Our g-MARS approach is different from the Fisher kernel. Firstly, we do not use a hidden markov model generated from the whole training dataset. Instead, we use the gapped markov chain generated from each individual training protein. Secondly, the "distance" between two proteins in the SVM is not calculated directly by the kernel function[9]. It is instead calculated by a classic relational kernel such as the RBF kernel.

Work[13,15] has been done on building a series of classifiers which make use of frequent substring patterns and the support vector machine. The algorithms firstly mine the frequent substrings from the training proteins that are frequent and discriminative for their own class (each pattern is mined with high confidence). Then they reform each sequence (training and testing sequences) by verifying which patterns are contained in it. An SVM is used for decision making on the reformatted dataset.

Preliminaries. A sequence $p = a_1a_2a_3...a_n$ is a length n sequence. Each character a_k in p is chosen from an alphabet set $\mathbb{A}$ and referred to as $p(k)$. Throughout this paper, we consider protein primary structure (amino acid sequences), but our technique is easily adapted to classification of other types of sequences as well.

In protein classification problems, a training dataset $TrDB$ contains proteins whose classes are known to the classifier. The class label for each protein p is denoted as $p.c$. A testing dataset $TeDB$ contains proteins whose classes are unknown to the classifier. The task is to predict the class label of each unknown protein sequence according to the training dataset. The predicted class label for each such protein p is denoted as $p.pc$. Given a testing protein p, if the predicted class label is the same as its real class label, that is, $p.pc = p.c$, we say it is correctly classified by the classifier, otherwise it is misclassified.

If the dataset only contains proteins from two classes, it is a binary-class classification problem. For the multi-class classification problem, where the testing dataset contains proteins belonging to more than two classes, we choose proteins from one class and merge the rest of the proteins into another class. In this way the multi-class classification problem can be reduced to a binary-class classification problem. The task is then to predict whether a testing protein belongs to the chosen class or not. The chosen class is called the positive class (or the target class) and can be denoted as T. The merged set of instances (named the negative class) containing all other proteins is denoted as $\neg T$. $TrDB_T = \{p \in TrDB \mid p.c = T\}$ is called the training positive set and $TrDB_{\neg T} = \{p \in TrDB \mid p.c \neq T\}$ is called the training negative set. Corresponding definitions exist for sets of testing instances $TeDB_T$ and $TeDB_{\neg T}$.

2 g-MARS Methodology

Training the g-MARS classifier has two main phases. Firstly, g-MARS builds for each $p \in TrDB$, a gapped markov chain. Secondly, g-MARS passes the vectorial expressions of the gapped markov chains to a support vector machine (SVM) for decision making.

Markov chains are a well known method for modeling sequences. The system consists of a set of states, where each is labelled by a character $a \in \mathbb{A}$ and a set of transitions which are associated with some probabilities. From one position to the next one of the sequence, the system undergoes a change of state (possibly a self-loop to the same state), according to the transition probability between the states. An important special case is the first order markov chain, where the transition probability depends only on the current and the predecessor position, i.e., $Pr[p(i) = a_k \mid p(i-1) = a_j, p(i-2) = a_m, ...] = Pr[p(i) = a_k \mid p(i-1) = a_j]$.

Furthermore, the markov chains we will consider are independent of the sequence positions. In other words, the probabilities of a transition from item a_m to a_n do not depend on the position in the sequence where transition occurs.

A markov chain modeling a sequence p consists of two kinds of components. One is the set of the states $\{S_i\}$ representing each character from $\mathbb{A}$ and the other is the set of transition probabilities $\{t_{ij}\}$ between states. The formal definition of transition probability t_{ij} leading from state S_i to S_j is: $t_{ij} = Pr[p(k) = a_j \mid p(k-1) = a_i]$.

In order to build a markov chain of the sequence p, we have to decide the probability of each pair of the states. A maximum likelihood estimation procedure is applied to calculate these probabilities: $t_{ij} = \frac{c_{ij}}{\sum_k c_{ik}}$, where c_{ij} is the number of times amino acid j follows amino acid i in p and $\sum_k c_{ik}$ is the number of times the amino acid i is followed by any other amino acid.

Example 1. *Consider the sequence $p = ABACCAB$. The markov chain for p has three states and we have $t_{AA} = 0$, $t_{AB} = \frac{2}{3}$, $t_{AC} = \frac{1}{3}$, $t_{BA} = 1$, $t_{BB} = 0$, $t_{BC} = 0$, $t_{CA} = \frac{1}{2}$, $t_{CB} = 0$ and $t_{CC} = \frac{1}{2}$.*

The purpose of building the markov chain for each protein is that similar global or local structures of two proteins can be captured by their markov chains. E.g., the probability for amino acid X followed by amino acid Y can be discriminative for proteins from two different classes. This is true if the proteins from the same class share a lot of common sections and those common sections are different between different classes. One issue is that it is rare for many proteins from the same class to share long common sections. The common parts may be similar, but not exactly the same. An example to further illustrate is:

Example 2. *Consider two sequences $p_1 = ABC$ and $p_2 = ADC$. The first order markov chains of them are quite different. For p_1, the non-zero probability transitions are $t_{AB} = 1$ and $t_{BC} = 1$. For p_2, the non-zero probability transitions are $t_{AD} = 1$ and $t_{DC} = 1$. There is no common non-zero transition probability between the markov chains of p_1 and p_2. However p_1 and p_2 share two out of three characters, which may indicate some similarity.*

2.1 Introduction to Gapped Markov Chains

To overcome the limitation of traditional markov chains which only model successive state transitions, we modify the traditional markov chain in two ways. The first is to model the ending of the sequence and the second is to add the concept of gaps.

Modelling the ending of the sequence. In Example 1, the transition probability t_{BA} is 1, meaning that in sequence p, if B is followed by any amino acid, it must be A. This does not consider the last character $p(7)$, which has no character following. A more complete model should illustrate that in p, the probability for B to be followed by A is 0.5 and the probability for B to be followed by nothing is 0.5. This can be reflected by changing the transition probability definition to $t_{ij} = \frac{c_{ij}}{c_i}$, where c_{ij} is the number of times amino acid j follows amino acid i in p and c_i is the number of times the amino acid i appears in p.

Although we consider the ending of the sequence, our markov chain won't contain the null (end of sequence) state and state transitions from other states to the null state (null transitions). There are two reasons: Firstly, when the transitions from one state to another non-null state are determined, its null transitions are also implicitly determined. Including the the null transition is redundant. Secondly, by removing the null transitions, we reduce the model size, which benefits for the classification process used later. In practice, the exclusion of these transitions does not impair classification accuracy.

Since we remove the null state and the null transitions, the sum of all the out-going transition probabilities in our markov chain model won't necessarily be 1. This is different from the markov chain introduced in the last section. From another point of view, the "rest" of the probability of a state goes to the null state which is "hidden".

The concept of gaps. In a g-gapped markov chain, we determine the probabilities of amino acid transitions, where there may be gaps between the amino acid pairs being considered. In particular, we allow contiguous (with no gap), jumping of one amino acid (with the gap as 1), jumping of two amino acids (with the gap as 2) and so on up to the g-th gap. The state transition probabilities are redefined as $t^k_{ij} = \frac{c^k_{ij}}{c_i}$, $0 \leq k \leq g$, where t^k_{ij} is the probability of a transition from amino acid i to amino acid j with gap as k in p; c^k_{ij} is the number of times amino acid i has gap k to amino to amino acid j in p. c_i is the number of times amino acid i appears in p.

Suppose we allowed a character $\emptyset$ called "The-Character-Don't-Care". Our gapped markov chain can be used to directly model sequences containing $\emptyset$. An example is given in Example 3.

Example 3. *Given a sequence $p = AB\emptyset BC$, the probability for it to be produced by a gapped markov chain can be calculated as $Pr(p) = t^0_{AB} * t^1_{BB} * t^0_{BC}$. The probability of p can be directly reflected by the gapped markov chain. Note that the probability of p could also be calculated by the traditional markov chain indirectly: $Pr(p) = t_{AB} * (\sum_i (t_{Bi} * t_{iB})) * t_{BC}$.*

The purpose of being able to model sequences containing $\emptyset$ is to capture the approximate similarity between protein sequences.

Example 4. *Consider two sequences $p_1 = ABC$ and $p_2 = ADC$. Comparing contiguous amino acid pairs gives no similarity between their transition probabilities (c.f. Example 2). If we ignore their second characters, the sequences become $p'_1 = A\emptyset C$ and $p'_2 = A\emptyset C$, which are the same. This commonality is reflected when we compare p_1 and p_2 allowing gaps in the markov chain: for gap equal to 1, we have the non-zero transition probabilities of p_1 as $t^0_{AB} = 1$, $t^0_{BC} = 1$ and $t^1_{AC} = 1$. The non-zero transition probabilities of p_2 are $t^0_{AD} = 1$, $t^0_{DC} = 1$ and $t^1_{AC} = 1$. We can see p_1 and p_2 now share one common transition probability.*

Given that we can generate a g-gapped markov chain for a sequence, how do we compare two markov chains to obtain the similarity between two sequences? A direct way would be, for each pair of states, compare their transition probabilities and count the number which are identical to get a score of the similarity of the two sequences. E.g., considering $p_1 = ABC$ and $p_2 = ADC$ from the previous example, the number of transitions having the same non-zero probability under a 0-gapped markov chain model is 0, so the similarity of p_1 and p_2 under gap 0 would be 0. The similarity score for a 1-gapped markov chain model would be 1, because they share exactly one common transition, namely t^1_{AC}.

In practice, we should not expect two similar proteins to share many such common transition probabilities. Instead, we would expect transition probabilities of proteins from the same class to have smaller variance and transition probabilities of proteins from different classes to have larger variance. SVMs are good at detecting such differences and so we use them for deriving a decision hyperplane that can separate gapped markov chain features of proteins from different classes.

Support vector machines using classic kernel functions require the input format to be vectors. We must therefore be able to represent gapped markov chains as vectors. This is straightforward: simply form a vector where each dimension corresponds to a transition and the value for that dimension is the probability of the transition. Transitions are annotated with gaps, so t^0_{AA} is considered to be a different dimension to t^1_{AA}. The ordering of the transitions does not matter as long as it is consistent for all the sequences in *TrDB* as well as *TeDB*.

Differences between gapped markov chains and traditional markov chains. As we can see, there are two main differences between our gapped markov chain and traditional markov chains. First of all, the summation of all out-going transition probabilities of a state is not necessarily to be 1 in our gapped markov chain, but it is a property of traditional markov chains. Secondly, the traditional markov chains describe successive states of a system. Our gapped markov chain can do that because any gapped markov chain contains the 0-th transition matrix, which is the traditional markov chain. But it can also model sequences containing $\emptyset$. As we discussed earlier, these two changes enhance their suitability for protein classification.

Differences between sequence modeling by gapped markov chains and by amino acid compositions. Recall the amino acid pair composition technique [8] we discussed earlier. There are several differences between that technique and our gapped markov chain technique. In the former case, the discriminative information is measured by the frequencies of the amino acid pairs. If the amino acid pairs are treated as patterns, these algorithms model each protein by their length-2 pattern frequencies. The model can be interpreted as: given an ordered pair of amino acids, how likely is it that this pair occurs in the protein? In our case, the discriminative information is measured by the probabilities of the amino acid transitions. The gapped markov chain models each protein by these pairwise amino acid transition probabilities. The model can be interpreted as: given a specific amino acid m, how likely is it that another amino acid n follows it? An important advantage is that the probability model is much less likely to be affected by the protein lengths.

2.2 The g-MARS Algorithm

g-MARS takes a set of training data *TrDB* and a gap parameter g as the input. For each protein in *TrDB*, g-MARS builds a g-gap markov chain. The associated vector for this chain has $(g+1)*20*20$ dimensions (unlike the k^{20} dimensions for the Spectrum kernel[11]). This is because there are $20*20$ possible amino acid pairs and we need to consider transitions with gap up to g for for each pair. In practice, we can easily set g to be as large as 10 and not incur dimensionality overload in classification. The markov chains for proteins from $TrDB_T$ and $TrDB_{\neg T}$ are passed as input to an SVM and it builds a classification model using these inputs. Any kernels available for the traditional SVM can be used, such as linear and RBF kernels. Building a g-gap markov chain for a set of n proteins requires $O(n*g*l)$ time, where l is the average length of the n proteins. The training time for g-MARS is the markov chain building time plus the SVM training time. Given a testing protein, the same g-gap markov chain is computed and passed to the SVM and it makes the classification decision is made by the SVM. The testing time for a length l protein in g-MARS is the markov chain building time($O(g*l)$) plus the SVM prediction time.

Although the discussion above is for the binary-class classification problem, g-MARS can be easily generalized to handle the multi-class classification problem. We turn the m-class classification problem into m reduced binary-class classification problems. Each time we pick one class out from the m classes as T and merge all the rest of the proteins as $\neg T$. In this way, way we build m SVMs, one for each target class. Given a testing protein, if there is an SVM classifying it to its target class, we classify it to that class. If more than one SVM classifies it to their target class, we classify it to the class with the highest score.

3 Experimental Results

Datasets. In order to test the general performance of g-MARS, we choose several different benchmark datasets, which cover a diverse range of characteristics.

The first set of data is chosen from PSORTb [4]. It contains proteins from different localizations of the bacteria. We pick out the proteins from the outer membrane of the Gram negative as the positive class and merge the proteins from the inner membrane, cytoplasmic and extra-cellular of the Gram negative as the negative class. Part of this data was used to evaluate the classifiers built on frequent substring patterns [13,15]. The positive class contains **352** proteins and the negative class contains **1013** proteins. The second set of data is proteins from different subcellular localizations from the Proteome Analyst Project [12]. We choose the proteins from the extracellular localization (**127** proteins) as the positive class and the proteins from the intracellular localization as the negative class (**3166** proteins). The third set of data is the outer membrane proteins versus the globular proteins which was used to evaluate the classifier built on amino acid compositions [6]. It contains **377** proteins from bacterial outer membrane and **674** Globular proteins.

The fourth set of data uses the G Protein-Coupled Receptor (GPCR) [2], the biggest known protein family. The GPCR database contains five level-0 GPCR classes (level-0 subfamilies). The largest subfamily is the Class A Rhodopsin like subfamily. It can be further divided into 16 level 1 subfamilies and more level 2 subfamilies. Classifiers have been developed to classify GPCR proteins from non-GPCR ones, the GPCR proteins from level 1 subfamilies, as well as the GPCR proteins from level 2 subfamilies [2]. We perform two experiments on this data. For the first experiment, we try to classify proteins from the level-0 subfamilies. Besides the five GPCR level-0 subfamilies, we add a non-GPCR family in order to test the ability for g-MARS to separate the GPCR proteins from non-GPCR ones. All six families can be obtained from *http://www.gpcr.org* [5]. For the second experiment, we try to classify proteins from the level 2 subfamilies. We select 4 level-2 subfamilies belonging to the Amine subfamily under level-0 subfamily Class A Rhodopsin like, namely, acetylcholine, adrenoceptors, dopamine and serotonin. These two experiments are multi-class classification problems. The specification of all the five experiments is listed in Tables 1 and 2.

Algorithms. We compare the accuracy of g-MARS against several algorithms: i) the spectrum kernel [11](Spectrum for short), which has been claimed to be better than Fisher kernel [11], ii) an amino acid composition classifier [6](AAC for short), iii) an amino acid pair composition with gap constraints classifier [8](AAPC for short), iv) simple markov chain classifier [3](MC for short), v) Frequent Substring Pattern based SVM [15](FS for short), vi) Generalised markov model (GMM [14]). The reasons for choosing these algorithms are: 1.g-MARS, AAPC and FS are all SVM-based hybrid algorithms. The difference between them is the way they "translate" sequences into vectors. 2.Spectrum is a famous protein classifier which makes use of the SVM and self-defined kernel function. 3.The AAC, GMM and MC methods are not based on support vector machines. They simply sum up the scores computed in each way up to make decisions. They are simple, well-known methods. We implemented all algorithms in Java using JDK version 1.4. All the experiments were conducted on a UNIX system with a 3.0GHz CPU and 1.5GB memory. We used the LIBSVM [1] Java package.

Table 1. G Protein-Coupled Receptor dataset list

Subfamily	#protein	% of Dataset
Class A Rhodopsin like	**1884**	**69.4%**
Amine		
Acetylcholine	66	15%
Adrenoceptors	120	27.3%
Dopamine	94	21.4%
Serotonin	159	36.2%
Class B Secretin like	**309**	**11.4%**
Class C Metabotropic glutamate/ pheromone	**206**	**7.6%**
Class D Fungal pheromone	**65**	**2.4%**
Class E cAMP receptors	**10**	**0.4%**
Class F Frizzled/Smoothened family	**130**	**4.8%**
Class Z Archaeal/bacterial/ fungal opsins(Non-GPCR)	**110**	**4.1%**
Total	**2714**	**100%**

Table 2. Binary-class classification dataset list

Dataset Description		# Protein		% of Dataset	
D	$\neg D$	D	$\neg D$	D	$\neg D$
Outer Membrane Proteins (OMP)*	Inner Membrane, Extracellular, Cytoplasm	352	1013	25.8%	74.2%
Extracellular proteins	Intracellular proteins	127	3166	3.9%	96.1%
Outer Membrane Proteins (OMP)*	Globular proteins	377	674	35.9%	64.1%

MC required no parameter settings. For the Spectrum kernel, we used $k = 3$. For g-MARS, AAC and AAPC, for each dataset we used the gap that gave the best average performance (according to f-measure, see below), using 5-fold cross validation with a verification dataset (a subset of the training data whose class labels are known to the classifiers, but which is not used in training). For the FS algorithm we mined the frequent substring patterns from the target class having minimum length as 3, minimum support as either 0.1% or 3 (whichever is greater) and minimum confidence of 90% [15]. For g-MARS, FS and AAPC, we used the RBF kernel. The gamma and cost parameters for this kernel were chosen using the tool in the LIBSVM package [1]. For g-MARS, one can use gamma as **0.0078125** and cost as **32.0** or **2048.0** to expect generally good performance. For GMM, we tested the three configurations provided by the authors. The first of these (standard single item 6th order Markov model) produced the best results in all datasets and we list its performance in the tables.

Table 3. Results of the three binary-class experiments

	OMP vs. Inn+Ext+Cyt				Extra vs. Intra				OMP vs. Globular			
Alg.	A%*	P%	R%	F%	A%	P%	R%	F%	A%	P%	R%	F%
g-MARS	95.16	**94.97**	85.8	90.15	**98.66**	93.68	70.08	**80.18**	**96.76**	**95.98**	94.96	**95.47**
Spectrum	94.36	92.83	84.66	88.56	98.15	92.31	56.69	70.24	95.62	95.59	92.04	93.78
FS	90.04	79.35	82.95	81.11	98	**95.52**	50.39	65.98	91.53	82.88	**96.29**	89.08
AAC	78.38	51.49	78.69	62.25	88.59	18.64	58.27	28.24	80.02	67.88	84.08	75.12
AAPC	**95.6**	94.24	88.35	**91.2**	98.45	93.18	64.57	76.28	92.86	85.78	96.02	90.61
MC	82.49	61.06	88.64	72.31	94.02	36.74	**76.38**	49.62	86.77	76.44	91.25	83.19
GMM	34.10	42.74	**100**	59.89	97.40	65.20	39.40	36.25	90.00	88.55	77.34	82.27

∗ *A: accuracy, P: precision, R: recall, F: f-measure.*

Evaluation. In order to give a comprehensive analysis of how good the classifiers are, we use 4 metrics accuracy (a), precision (p), recall (r) and f-measure (f) as:

$a = \frac{|\{t \subseteq TeDB | t.pc = t.c\}|}{|TeDB|}$; $p = \frac{|\{t \subseteq TeDB_T | t.pc = T\}|}{|\{t \subseteq TeDB | t.pc = T\}|}$;

$r = \frac{|\{t \subseteq TeDB_T | t.pc = T\}|}{|\{t \subseteq TeDB_T\}|}$; $f = \frac{2 * p * r}{(p+r)}$.

For multi-class classification, a different overall accuracy measurement is used: $a = \sum_{T_i} \frac{|\{t \subseteq TeDB_{T_i} | t.pc = t.c\}|}{|TeDB|}$. The accuracy for each target class T_i is calculated as: $a_i = \frac{|\{t \subseteq TeDB_{T_i} | t.pc = t.c\}|}{|TeDB_{T_i}|}$. The accuracy measurement tells how many proteins are classified correctly overall. For the rare-class classification case, precision and recall are more meaningful. When comparing algorithms, the f_measure is a standard way of combining precision and recall to get a single measure. We used stratified 5-fold cross validation for testing.

Performance on binary-class data. The accuracies of the five algorithms on the 3 binary-class classification problems are listed in Table 3. g-MARS performs strongly for the first set of data, the outer membrane proteins vs. the inner membrane, extracellular and the cytoplasm proteins. In this set the amino acid pair composition algorithm works quite well. g-MARS gives generally good performances on all the datasets, because the discriminative information of markov chains in g-MARS does not rely on any particular property of proteins being in specific domains. The MC classifier given in the last row uses the log odd ratio score to classify the proteins [3]. The performance tells that simply adding up the score ratios from different classes does not give good answers. This partly shows the superiority of using the SVM to make the decisions.

Performance on GPCR subfamilies. The classification results for the GPCR level-2 and level-0 subfamilies are given in Tables 4 and 5, respectively. The diversities of subfamilies are greater for level-0 proteins than for level-2 proteins. That is the reason why generally we gain better results for level-2 proteins. There are certain subfamilies in level-0 that are easily separated from other subfamilies such as Class E cAMP receptors. Most of the classifiers do not make

Table 4. The accuracy (%) of the GPCR level 2 subfamilies prediction

Level-2 Subfamily	g-MARS	Spectrum	FS	AAC	AAPC	MC	GMM
Acetylcholine	**100**	95.45	95.45	87.88	93.93	95.45	85.52
Adrenoceptors	**100**	**100**	**100**	62.5	**100**	95.83	83.67
Dopamine	**98.93**	95.74	94.68	76.6	85.11	85.11	80.21
Serotonin	98.11	**100**	97.48	77.99	94.97	94.34	75.48

Table 5. The accuracy (%) of the GPCR level 0 subfamilies prediction.

Level-0 Subfamily	g-MARS	Spectrum	FS	AAC	AAPC	MC	GMM
Class A Rhodopsin like	**99.84**	99.52	98.57	78.66	99.73	91.77	77.93
Class B Secretin like	**99.38**	98.06	95.47	60.2	95.46	96.12	94.00
Class C Metabotropic glutamate/pheromone	**98.06**	95.15	91.26	76.21	97.09	93.69	82.70
Class D Fungal pheromone	89.23	86.15	81.54	89.23	83.07	**95.38**	76.20
Class E cAMP receptors	**100**	**100**	**100**	90	90	**100**	83.00
Class F Frizzled/Smoothened family	**98.46**	97.69	96.15	93.85	92.31	90.77	85.53
Class Z Archaeal/bacterial/fungal opsins (non-GPCR)	**96.36**	94.55	86.36	92.73	92.73	95.45	90.12

mistakes for proteins from this family. By looking at this family we know that the structures of the proteins within this family are quite different from proteins of other families. Some proteins contain long contiguous asparagine and histidine. The performances for most classifiers are quite good for identifying which protein belongs to Class A Rhodopsin like subfamily (Hight percentages in the first row of Table 5). This is due to the abundant proteins of this family. So as an observation about Table 5, we can say that for the classifiers tested here, having more testing data means a more accurate the model can be built. The more distinctive the data is, the easier it is for the model to make correct decision. This is generally true for most feature-based classifiers. From both the tables we can also see that g-MARS performs generally better than all the other classifiers.

T-test. We conducted t-tests with a 95% confidence on the results. g-MARS wins **16** times, draws **9** times and loses **0** times. The Spectrum ranks the second best with **11** wins, **14** draws and **1** loses. The third best algorithm is the AAPC and the performance is **8** times winning, **14** times drawing and **3** times losing. From a statistical point of view, g-MARS wins the most which means it performs generally the best. Comparing directly against the Spectrum, g-MARS wins on 1 dataset and on the rest draws. It's running time is generally at least 10% faster than the Spectrum, even for high gaps.

How to choose the proper gap. The gap can be chosen by performing cross validation on the training dataset. Set aside a portion of training data as test

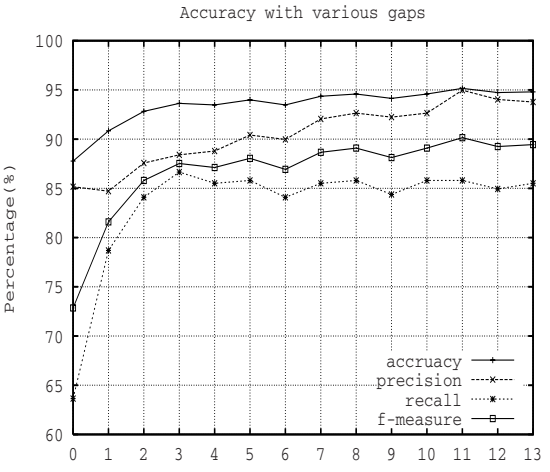


Fig. 1. g-MARS performance for varying gap on OMP vs. Inner, Extra, Cytoplasm dataset

data, try different gaps and choose the one which yields best accuracy. We can also make some general remarks about gap behaviour. Figure 1 shows the change in accuracy for g-MARS with various gaps. For gap 0, which is the case in the traditional markov chain, the performance is poor. With the increase of the gap, overall performance becomes stable. The f-measure achieves its peak value when the gap is set to 11. So instead of using cross validation, one could also begin by using the gap as 7 and then increase and decrease the gap from this value, finishing when the result remains stable (changes are smaller than a certain θ).

Running time. For classifiers working on large volumes of data, time efficiency is an important factor. We discussed the time complexity of g-MARS in Section 2.2. We also measured the running time for g-MARS on the OMP vs. Inner, Extra and Cytoplasm dataset with various gaps. The time includes the time 5-fold cross validation. and increases roughly linearly with the increment of the gap. For gap as 0, the executable time is less than 25 seconds and for the largest gap it only takes slightly more than 350 seconds, which is quite acceptable.

4 Conclusion and Future Work

In this paper we have extended the traditional markov chain to the gapped markov chain. We proposed the g-MARS classifier, which uses gapped markov chains and support vector machines to classify proteins. Compared to other work, it has the following merits: It is computationally efficient and can handle large volumes of proteins. It does not need prior knowledge to achieve good performance and can be generalized to any sequence classification problem. The growth of the gap length increases the dimension of the vectors linearly rather than exponentially like the Spectrum kernel, so it is realistic to use large gaps.

Experimental results show it has generally superior accuracy for a range of protein datasets with diverse characteristics. Overall, g-MARS is a very practical algorithm to handle protein classification.

References

1. Chang, C., Lin, C.: LIBSVM: a library for support vector machines (2001)
2. Cheng, B., Carbonell, J., Klein-Seetharaman, J.: Protein classification based on text document classification technique. PROTEINS: Structures, Function and Bioinformatics. 58, 955–970 (2005)
3. Durbin, R., Eddy, S., Krogh, A., Mitchison, G.: Biological sequence analysis—-Probabilistic models of proteins and nucleic acids. Cambridge University Press, Cambridge (1998)
4. Gardy, J.L., Laird, M.R., Chen, F., Rey, S., Walsh, C.J., Ester, M., Brinkman, F.S.L.: Psortb v.2.0: Expanded prediction of bacterial protein subcellular localization and insights gained from comparative proteome analysis. Bioinformatics 21(5), 617–623 (2005)
5. GPCRDB, http://www.gpcr.org
6. Gromiha, M.M., Suwa, M.: A simple statistical method for discriminating outer membrane proteins with better accuracy. Bioinformatics 21(7), 961–968 (2005)
7. Gusfield, D.: Algorithms on Strings, Trees, and Sequences - Computer Science and Computational Biology. Cambridge University Press, Cambridge (1997)
8. Huang, S., Liu, R., Chen, C., Chao, Y., Chen., S.: Prediction of outer membrane proteins by support vector machines using combinations of gapped amino acid pair compositions. In: BIBE, pp. 113–120 (2005)
9. Jaakkola, T., Diekhans, M., Haussler, D.: Using the fisher kernel method to detect remote protein homologies. In: ISMB, pp. 149–158 (1999)
10. Leslie, C.S., Eskin, E., Cohen, A., Weston, J., Noble, W.S.: Mismatch string kernels for discriminative protein classification. Bioinformatics 20(4) (2004)
11. Leslie, C.S., Eskin, E., Noble, W.S.: The spectrum kernel: A string kernel for svm protein classification. In: Pacific Symp. on Biocomputing, pp. 566–575 (2002)
12. Liu, Z.: Predicting protein subcellular localization from homologs using machine learning algorithms. Master's thesis, Dept of Computer Science, University of Alberta (2002)
13. She, R., Chen, F., Wang, K., Ester, M., Gardy, J.L., Brinkman, F.S.L.: Frequent-subsequence-based prediction of outer membrane proteins. In: KDD, pp. 436–445 (2003)
14. Wang, J., Hannenhalli, S.: Generalizations of markov model to characterize biological sequences. BMC Bioinformatics 6(219) (2005)
15. Zhou, S., Wang, K.: Localization site prediction for membrane proteins by integrating rule and svm classification. IEEE Trans. Knowl. Data Eng. 17(12), 1694–1705 (2005)

Domain-Domain Interaction Identification with a Feature Selection Approach

Xing-Ming Zhao and Luonan Chen

Institute of Systems Biology,
Shanghai University, 200444 Shanghai, China
xm_zhao@shu.eud.cn, chen@eic.osaka-sandai.ac.jp

Abstract. The protein-protein interactions (PPIs) are generally assumed to be mediated by domain-domain interactions (DDIs). Many computational methods have been proposed based on this assumption to predict DDIs from available data of PPIs. However, most of the existing methods are generative methods that consider only PPI data without taking into account non-PPIs. In this paper, we propose a novel discriminative method for predicting DDIs from both PPIs and non-PPIs, which improves the prediction reliability. In particular, the DDI identification is formalized as a feature selection problem, which is equivalent to the parsimonious principle and is able to predict both DDIs and PPIs in a systematic and accurate manner. The numerical results on benchmark dataset demonstrate that formulating DDI prediction as a feature selection problem can predict DDIs from PPIs in a reliable way, which in turn is able to verify and further predict PPIs based on inferred DDIs.

Keywords: Discriminative approach, domain-domain interaction, feature selection, protein-protein interaction.

1 Introduction

Proteins exert their functions by interacting with each other [1]. Generally, one protein interacts with its partner by binding one of its domains to the domain(s) in its target protein. In other words, proteins interact with each other through domain-domain interactions (DDIs) [2] [3]. Recently, many computational methods have been proposed to identify domain interactions from protein interactions. For example, Sprinzak and Margalit [4] proposed the Association method for predicting domain interactions based on the frequency of observed protein interactions that contain the pair of domains. Deng et al. [5] presented a maximum likelihood estimation (MLE) method as well as an Expectation-Maximization (EM) algorithm to infer underlying domain interactions from protein interactions. Liu et al. [6] combined protein interactions from multiple species to identify interacting domain pairs. Riley et al. [7] developed a new method, namely Domain Pair Exclusion Analysis (DPEA), to predict domain interactions based on all of the protein interactions from Database of Interacting Proteins (DIP) [8], where a new score, i.e. *E*-value, is developed to assess the contribution of each possible domain pair

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 178–186, 2008.

to the likelihood of a set of observed protein interactions. It is shown that the DPEA method outperforms both MLE [5] and Association method [4]. Recently, Guimaräes et al. [9] developed a linear programming model, namely Parsimonious Explanation (PE), to predict domain interactions based on the parsimonious principle with the assumption that a given set of protein-protein interactions are accomplished through the minimal set of domain interactions. The PE method is shown to outperform DPEA [7], Association [4] and MLE [5] based on numerical simulations. Moreover, Lee et al. [10] developed a Bayesian approach to predict high confidence domain interactions based on the integration of multiple data sources from multiple species, where the integration of multiple data sources significantly improves the prediction accuracy compared with single data source analysis.

The methods described above assume that protein interactions are mediated by domain-domain interactions, and they try to identify the DDIs underlying the PPIs. Despite the success on specific datasets, most of the existing computational methods are generative methods, where a model is constructed based on the protein-protein interaction data with the assumption that proteins interact with each other through domain-domain interactions. However, the existing methods use only available data of PPIs without the consideration of the non-PPIs, which may result in imbalance problem of information [11] [12]. Since the proteins are assumed to interact through domain interactions, domain pairs occurring in the non-PPIs are more likely false DDIs [7]. Therefore, the non-PPI data can provide insight into the domain interaction.

In this paper, we proposed a novel discriminative approach, namely domain interaction prediction in a discriminative way (DIDD), to predict domain interactions based on protein interactions. Different from the existing methods, both PPIs and non-PPIs are considered in DIDD, thereby not only alleviating the imbalance problem of information but also improving prediction accuracy. In particular, DDI prediction is formulated as a feature selection problem in machine learning, which is in consistent with parsimonious principle that protein interactions are accomplished through the minimum set of domain interactions. In feature selection, the possible domain pairs are assessed according to their contributions to the discrimination between PPIs and non-PPIs. The proposed method is able to predict DDIs based on PPIs, which in turn can predict and verify PPIs based on the inferred DDIs, i.e. selected features in this case. The numerical results on benchmark datasets demonstrate the effectiveness and efficiency of the proposed method.

The rest of the paper is organized as follows: Section 2 describes the methods that are proposed for identifying DDIs from PPIs; Section 3 presents the numerical results on benchmark dataset; The conclusions are drawn finally.

2 Methods

The idea behind DDI identification in PPIs is that PPIs are mediated by DDIs. Therefore, the domain pairs that best discriminate PPIs and non-PPIs are more

likely the true DDIs. The PE [9] method assumes that the given protein interactions can be approximately accomplished by the minimum set of domain-domain interactions, which is in consistent with the idea behind feature selection that tries to find out as few informative features (e.g. DDIs) as possible for classification (e.g. discrimination between PPIs and non-PPIs). However, feature selection works in a discriminative way that take into account non-PPIs. Therefore, higher prediction accuracy is expected.

2.1 Feature Vector Construction

The discrimination between PPIs and non-PPIs is actually a binary classification problem. To utilize the discriminative methods, the positive samples (i.e. PPIs) and negative samples (i.e. non-PPIs) should be represented as feature vectors. In this work, each sample is a protein pair (either interacting pair or non-interacting pair) and is represented as a vector. The positive samples are PPIs from protein interaction database, whereas the negative samples are protein pairs that are randomly generated except the known PPIs. The rationality behind the method generating negative samples is that only one out of six hundred randomly generated protein pairs is possibly the true PPI, and this method has also been employed widely in the literature [13] [14]. For the PPIs, all the possible combinations of two domains are found and kept in order, where each domain pair exists in at least one interacting protein pair. After getting all the domain pairs, each sample is represented as a feature vector, where the feature value is 1 if the corresponding domain pair occurs in the sample and otherwise 0.

2.2 Classifier

After constructing the feature vectors, we need to design classifier to discriminate the PPIs from non-PPIs. It can be seen that the vectors generated above have following properties: 1) sparse content, i.e. most of the feature values are 0; 2) high dimension due to the large number of possible combinations among domains; 3) few positive samples but large number of negative smaples. Therefore, the conventional classifiers such as Support Vector Machines and Nearest neighbor classifier cannot be used here.

In this paper, we designed a simple classifier to discriminate PPIs from non-PPIs based on the specific data structure and the assumption that DDIs mediate PPIs. For a given protein pair vector $\mathbf{x_i}$, the class label y_i corresponding to it can be defined as:

$$y_i = \begin{cases} 1, & \text{if } \sum_j x_{ij} \geq 1, \\ -1, & \text{otherwise,} \end{cases} \tag{1}$$

where x_{ij} is the value of the jth feature in the ith sample. The idea behind the classifier is that if the domain combination corresponding to x_{ij} is the true DDI and the protein pair $\mathbf{x_i}$ contains the domain combination (i.e. $x_{ij} = 1$ in this case), then the protein pair $\mathbf{x_i}$ is an interacting pair and $y_i = 1$ accordingly. Since the classifier does not involve any model training procedure, there is not any problem of overfitting.

2.3 Feature Selection

Given a set of PPIs and non-PPIs, we want to find out which domain combinations mediate the PPIs, i.e. the putative DDIs. With the constructed vectors and the assumption that PPIs are mediated by DDIs, we can formulate DDI prediction as a feature selection problem. In feature selection, the purpose is to find out as few informative features as possible to build a reliable and accurate learning model. In this case, feature selection aims to find out the domain pairs that discriminate PPIs from non-PPIs. It can be seen that the idea behind feature selection is equivelant to the parsimonious principle that the domain-domain interactions are well approximated by the minimum set of DDIs mediating the given set of PPIs [9]. However, non-PPIs are also taken into account in this work. The domain pairs kept in feature selection are assumed to be the putative DDIs.

Considering the specific data structure and imbalance between positive and negative samples, the unbalanced correlation score proposed in [15] is utilized to rank the features, which is defined as:

$$s_j = \sum_{y_i=1} x_{ij} - \lambda \sum_{y_i=-1} x_{ij} \tag{2}$$

where s_j is the score for the jth feature, y_i is the label for sample $\mathbf{x}_i$, x_{ij} is the value for the jth feature in vector x_i, and λ is a penalty parameter to punish the occurrance of the feature in negative samples. Generally, a large value is adopted for λ, e.g. $\lambda = 5$ in this work. The higher the score s_j is, the higher feature j is ranked. The idea behind the method is that the more frequently the feature occurs in positive samples and less in negative samples, the more informative it is, thereby more likely true DDIs.

2.4 Performance Evaluation

To see the performance of the classifier, the following measures are adopted in this work, including *precision*, *recall*, *F1*-measure:

$$precision = \frac{TP}{TP + FP} \tag{3}$$

$$recall = \frac{TP}{TP + FN} \tag{4}$$

$$F1 = \frac{2 * precision * recall}{precision + recall} \tag{5}$$

where TP means the number of positive samples that are predicted correctly, FN means the number of positive samples that are predicted as negative samples, FP means the number of negative samples that are predicted as positive samples, and FP means the number of negative samples that are predicted correctly.

3 Results and Discussion

To test the performance of the proposed method, the DIDD method was applied to predict DDIs based on a set of protein-protein interactions from 69 organisms, which was constructed by Riley and colleagues [7]. This dataset is denoted as "Riley test set" here, and was also used by the PE method [9]. The Riley test set contains all the protein interactions from DIP database [8], and domains were assigned to proteins by employing the Pfam hidden markov model profiles [16]. Note that only Pfam-A domains were assigned to the proteins here. Table 1 lists the datasets used in this work, where PPI denotes protein-protein interactions, DDI denotes domain-domain interactions, proteins are proteins involved in PPIs, and number represents the number of PPIs, non-PPIs, proteins and potential DDIs used in this work.

Table 1. The datasets used in this work

	PPIs	non-PPIs	Proteins	Potential DDIs
Number	26,032	11,651,400	11,403	27,617

The DIDD method was first applied to predict the DIP protein interactions by selecting informative features (i.e. domain pairs), where the unbalanced correlation score [15] was employed in feature selection. This procedure continues until the prediction accuracy does not improve any more. Consequently, the domain pairs corresponding to the selected features were seen as putative DDIs. With the iPfam dataset as the gold standard, we compared DIDD with previous methods, i.e. DPEA and PE methods, with respect to precision and recall, where the results by PE are those predicted with PPI reliability of 50% and pw-score ≤ 0.01, and the top 3005 predictions by DPEA are seen as its predictions. For fair comparison, only the predictions by PE and DPEA that involve Pfam-A domains are considered. In particular, we investigated the number of difficult predictions by different methods as described in [9] and [7], where the necessity of assessing predictions with respect to the difficulty has been justified in [7]. In this work, the definition of difficult prediction described in [9] was adopted to validate the proposed method, and a DDI is assumed to be difficult to predict if the domain pair is not contained in any single-domain interacting protein pair. Table 2 summarizes the prediction results by different methods on this dataset. From the results, we can see that the DIDD method outperforms the other two methods. Especially, DIDD can predict more difficult tasks compared with other two methods. The results clearly demonstrate the prediction power of the proposed method especially on difficult gold standard pairs, and thereby is a good complement to existing methods. Furthermore, the results also demonstrate that the non-PPIs can really help to improve the prediction accuracy.

In addition, to test the performance of DIDD, the datasets from DOMINE [17] database were employed to test how much of our predictions can be validated by at least one other existing computational method. DOMINE is a database that contains known and predicted domain interactions, including DDIs from 3DID

Table 2. Comparison of different methods on DDI prediction based on Riley test set

method	precision	recall	difficult predictions
DPEA	10.21%	23.63%	5
PE	12.21%	29.63%	75
DIDD	20.80%	29.76%	157

Table 3. Comparison of various methods on Riley test set, where percentage means the percentage of predictions that can be validated by at least one other existing method.

Methods	DIDD	DPEA+PE	RCDP	Bayesian
Percentage	50.71%	23.80%	39.60%	41.50%

Table 4. Performance of various methods in predicting PPIs based on inferred DDIs for the Riley test set

Methods	precision	recall	F1
DIDD	100.00%	6.07%	11.44%
DPEA	3.30%	16.24%	5.48%
PE	3.68%	19.19%	6.17%

[18], iPfam [19] and those predicted by different computational approaches. In this example, the DIDD method was compared against DPEA [7], PE [9], RCDP [20] and Bayesian approach [10]. Table 3 shows the comparison of various methods on how much of their prediction can be validated by at least one other existing computational method, where the statistics of the other methods are from the DOMINE database [17]. From the results, we can see that most of our predictions have been validated by at least one other computational method and has higher prediction accuracy compared with other methods, which confirms the efficiency and effectiveness of DIDD. In the DOMINE [17] database, each predicted or known DDI is associated with a confidence score. To see the performance of DIDD, we further investigated how much of the our predictions have high confidence scores. Figure 1 shows the distribution of confidence scores for the predictions by DIDD, where we can see that a relatively small number of our predictions have low confidence scores, which clearly demonstrate the prediction power of DIDD.

In addition, to validate the predicted DDIs by the proposed method, we predicted PPIs based on the inferred DDIs. In addition, DIDD was compared against PE and DPEA in predicting PPIs based on the inferred DDIs to test the predicted DDIs. In this work, the classifier and samples (i.e PPIs and non-PPIs) used by DIDD were employed to test the predicted DDIs by PE [9] and DPEA [7], where the inferred DDIs were treated as the selected features as described in Methods. Table 4 shows the comparison of performance of various methods in predicting PPIs based on inferred DDIs. From Table 4, it can be seen that the proposed DIDD method outperforms other existing methods in discriminating the PPIs and non-PPIs based on inferred DDIs. The DIDD method got high precision means that

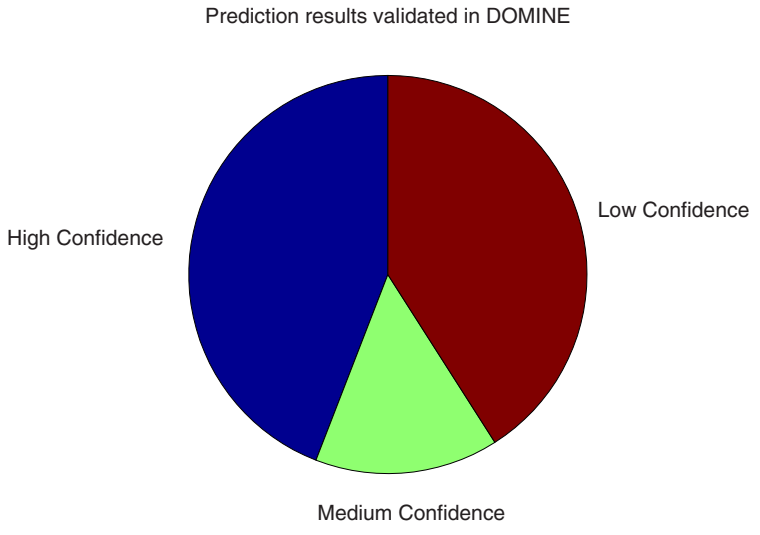


Fig. 1. Distribution of confidence scores for the predictions by DIDD on Riley test set

most of its predictions are true DDIs, while a low recall means that DIDD only predict a small number of domain pairs as domain interactions. The results demonstrate that the DDIs predicted by DIDD are more possibly the true DDIs that mediate PPIs compared against other existing methods, which confirms the effectiveness and efficiency of DIDD in predicting DDIs from PPIs. Furthermore, the DIDD can also validate and predict PPIs based on the inferred DDIs.

4 Conclusions

Understanding PPIs at domain level can provide insight into protein function and evolutionary history of PPIs. In this paper, a novel method, namely domain interaction prediction in a discriminative way (DIDD), is presented for predicting DDIs from data of available PPIs. Unlike existing methods, DIDD considers both PPIs and non-PPIs. Since PPIs are typically assumed to be mediated by DDIs, the domain combinations occurring in the non-PPIs are more possibly false DDIs. Therefore, higher prediction accuracy is expected for DIDD by taking non-PPIs into account. In particular, in this work, DDI prediction is formalized as feature selection, which is in consistent with parsimonious principle that DDIs can be approximated by the minimum set of DDIs that mediate the given PPIs [9]. By selecting the informative features, DIDD can predict those DDIs that really mediate the given PPIs, and in turn help to verify and predict PPIs based on the inferred DDIs. The results on benchmark data prove the predictive power of our method. In addition, the overlap between the predictions by DIDD and published results demonstrate the effectiveness and efficiency of the proposed method.

Acknowledgements

This work was partly supported by the National High Technology Research and Development Program of China (2006AA02Z309), and JSPS-NSFC collaboration project.

References

1. Eisenberg, D., Marcotte, E., Xenarios, I., Yeates, T.: Protein function in the post-genomic era. Nature 405, 823–826 (2000)
2. Itzhaki, Z., Akiva, E., Altuvia, Y., Margalit, H.: Evolutionary conservation of domain-domain interactions. Genome Biology 7(12), R125 (2006)
3. Zhao, X., Wang, Y., Chen, L., Aihara, K.: Protein domain annotation with integration of heterogeneous information sources. Proteins: Structure, Function, and Bioinformatics (in press, 2008)
4. Sprinzak, E., Margalit, H.: Correlated sequence-signatures as markers of protein-protein interaction. J. Mol. Biol. 311(4), 681–692 (2001)
5. Deng, M., Mehta, S., Sun, F., Chen, T.: Inferring domain-domain interactions from protein-protein interactions. Genome Res. 12(10), 1540–1548 (2002)
6. Liu, Y., Liu, N., Zhao, H.: Inferring protein-protein interactions through high-throughput interaction data from diverse organisms. Bioinformatics 21(15), 3279–3285 (2005)
7. Riley, R., Lee, C., Sabatti, C., Eisenberg, D.: Inferring protein domain interactions from databases of interacting proteins. Genome Biol. 6(10) (2005)
8. Xenarios, I., Rice, D., Salwinski, L., Baron, M., Marcotte, E.M., et al.: DIP: the Database of Interacting Proteins. Nucl. Acids Res. 28(1), 289–291 (2000)
9. Guimaraes, K., Jothi, R., Zotenko, E., Przytycka, T.: Predicting domain-domain interactions using a parsimony approach. Genome Biology 7(11), R104 (2006)
10. Lee, H., Deng, M., Sun, F., Chen, T.: An integrated approach to the prediction of domain-domain interactions. BMC Bioinformatics 7(1), 269 (2006)
11. Zhao, X., Li, X., Chen, L., Aihara, K.: Protein classification with imbalanced data. Proteins: Structure, Function, and Bioinformatics 70(4), 1125–1132 (2008)
12. Zhao, X., Chen, L., Aihara, K.: Gene function prediction using labeled and unlabeled data. BMC. Bioinformatics 9, 57 (2008)
13. Qi, Y., Bar-Joseph, Z., Klein-Seetharaman, J.: Evaluation of different biological data and computational classification methods for use in protein interaction prediction. Proteins: Structure, Function, and Bioinformatics 63(3), 490–500 (2006)
14. Tong, A., Lesage, G., Bader, G., Ding, H., Xu, H., et al.: Global Mapping of the Yeast Genetic Interaction Network. Science 303(5659), 808–813 (2004)
15. Weston, J., Perez-Cruz, F., Bousquet, O., Chapelle, O., Elisseeff, A., et al.: Feature selection and transduction for prediction of molecular bioactivity for drug design. Bioinformatics 19(6), 764–771 (2003)
16. Bateman, A., Coin, L., Durbin, R., Finn, R., Hollich, V., et al.: The Pfam protein families database. Nucl. Acids Res. 32(suppl.1), D138–141 (2004)
17. Raghavachari, B., Tasneem, A., Przytycka, T., Jothi, R.: DOMINE: a database of protein domain interactions. Nucl. Acids Res. 36(suppl.1), D656–661 (2008)

18. Stein, A., Russell, R., Aloy, P.: 3did: interacting protein domains of known three-dimensional structure. Nucl. Acids Res. 33(suppl.1), D413–417 (2005)
19. Finn, R., Marshall, M., Bateman, A.: iPfam: visualization of protein-protein interactions in PDB at domain and amino acid resolutions. Bioinformatics 21(3), 410–412 (2005)
20. Raja, J., Praveen, F., Asba, T., Teresa, M.: Co-evolutionary analysis of domains in interacting proteins reveals insights into domain-domain interactions mediating protein-protein interactions. Journal of Molecular Biology 362, 861–875 (2006)

Dividing Protein Interaction Networks by Growing Orthologous Articulations

Pavol Jancura[1], Jaap Heringa[2], and Elena Marchiori[1,⋆]

[1]Intelligent Systems, ICIS, Radboud Universiteit Nijmegen, The Netherlands
{jancura,elenam}@cs.ru.nl
[2]IBIVU, Vrije Universiteit Amsterdam, The Netherlands
{heringa}@few.vu.nl

Abstract. The increasing growth of data on protein-protein interaction (PPI) networks has boosted research on their comparative analysis. In particular, recent studies proposed models and algorithms for performing network alignment, the comparison of networks across species for discovering conserved modules. Common approaches for this task construct a merged representation of the considered networks, called alignment graph, and search the alignment graph for conserved networks of interest using greedy techniques. In this paper we propose a modular approach to this task. First, each network to be compared is divided into small subnets which are likely to contain conserved modules. To this aim, we develop an algorithm for dividing PPI networks that combines a graph theoretical property(articulation) with a biological one (orthology). Next, network alignment is performed on pairs of resulting subnets from different species. We tackle this task by means of a state-of-the-art alignment graph model for constructing alignment graphs, and an exact algorithm for searching in the alignment graph. Results of experiments show the ability of this approach to discover accurate conserved modules, and substantiate the importance of the notions of orthology and articulation for performing comparative network analysis in a modular fashion.

Keywords: Protein network dividing, modular network alignment.

1 Introduction

With the exponential increase of data on protein interactions obtained from advanced technologies, data on thousands of interactions in human and most model species have become available (e.g. [1,2]). PPI networks offer a powerful representation for better understanding modular organization of cells, for predicting biological functions and for providing insight into a variety of biochemical processes.

Recent studies consider a comparative approach for the analysis of PPI networks from different species in order to discover common protein groups which are likely to be related to shared relevant functional modules [3,4,5].

⋆ Corresponding author.

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 187–200, 2008.

This problem is also known as *pairwise network alignment*. Algorithms for this task typically construct a merged graph representation of the networks to be compared, called alignment (or orthology) graph, and model the problem as an optimization problem on such graph. Due to the computational intractability of such optimization problem, greedy algorithms are commonly used [6,7,8,9,10].

1.1 Problem Statement

Conserved modules, discovered by computational techniques such as [6], have in general small size compared to the size of the PPI network they belong to. Moreover, PPI networks are known to have a scale-free topology where most proteins participate in a small number of interaction while a few proteins, called *hubs*, contains a high number of interaction. As indicated by recent studies, hubs whose removal disconnects the PPI network (articulation hubs) are likely to appear in conserved interaction patterns [11,12]. These observations motivate the focus of this paper on the problem of performing modular network alignment. Specifically, we propose a two phases approach for this task: divide and align. The divide phase transforms each PPI network into a set of small subnets which are expected to cover conserved complexes. The align phase uses an existing evolution-based alignment graph model to merge suitable pairs of subnets from each species, and an exact search technique for extracting conserved modules from each alignment graph.

1.2 Contributions

We introduce an heuristic algorithm for dividing a PPI network into subnets, which combines biological (orthology) and graph theoretical (articulation) information. The algorithm starts by identifying groups of orthologous articulations, called centers, which are expanded into subsets consisting of orthologous nodes.

The algorithm automatically determines the number of subsets and has the property of being parameterless.

We use this algorithm for performing network alignment, by merging pairs of resulting subnets from different species, and applying exact optimization for searching conserved modules across species. We introduce a new notion, *modular alignment*, because we align only particular PPI subnets achieving conserved modules inside of them while current methods of global or local network alignment try to align whole PPI networks.

In order to test the performance of this approach, we consider an instance of the method that uses a state-of-the-art evolution-based alignment graph model [6]. Results of experiments show effectiveness of the proposed approach, which is capable of detecting accurate conserved complexes. Furthermore, we show that improved performance can be achieved by merging modules detected with our algorithm with those identified by Koyuturk et al. algorithm [6]. In general, these results substantiate the important role of the notions of orthology and articulation in modular comparative PPI network analysis.

1.3 Related Work

Recent overviews of approaches and issues in comparative biological networks analysis are presented in [4,5]. Based on the general formulation of network alignment proposed in [3], a number of techniques for (local and global) network alignment have been introduced [6,7,8,9,10,13].

Techniques for local network alignment commonly construct an orthology graph, which provides a merged representation of the given PPI networks, and search for conserved subnets using greedy techniques [6,7,8,9,10].

While the above algorithms focus on alignment of whole global networks, we focus on 'modular' network alignment. Modular network alignment is an alignment of particular subnets of given networks to be compared. To the best of our knowledge, we propose the first algorithm which directly tackles the modularity issue in network alignment in the meaning that dividing step achieves conserved modules inside of particular subnets and therefore one can perform only modular alignment for local network alignment problem.

Many papers have investigated the importance of hubs in PPI networks and functional groups [12,14,15,16,17,18]. In particular, it has been shown that hubs with a central role in the network architecture are three times more likely to be essential than proteins with only a small number of links to other proteins [16]. Moreover, if we take functional groups in PPI networks, then, amongst all functional groups, cellular organization proteins have the largest presence in hubs whose removal disconnects the network [12]. Computational techniques for identifying functional modules in PPI networks generally search for clusters of proteins forming dense components [19,20]. The scale-free topology of PPI networks makes difficult to isolate modules hidden inside the central core [21]. In [22] several multi-level graph partitioning algorithms are described addressing the difficulty of partitioning scale-free graphs.

The approach we propose differs from the above mentioned works because it does not address (directly) the problem of identifying functional modules in a PPI network, but uses homology information and articulations for dividing PPI networks into subnets in order to perform network alignment in a modular fashion.

2 Graph Theoretic Background

Given a graph $G = (U, E)$, nodes joined by an edge are called *adjacent*. A *neighbor* of a node u is a node adjacent to u. The degree of u is the number of elements in E containing the vertex u.

Let $G(U, E)$ be a connected undirected graph. A vertex $u \in U$ is called *articulation* if the graph resulting by removing this vertex from G and all its edges, is not connected.

A *tree* is a connected graph not containing any circle. A tree is called *rooted tree* if one vertex of the tree has been designated as the root. Given a rooted tree $T(V, F)$, the depth of a vertex $v \in V$ is the number of edges from the root to v without repetition of edges. Leaves of the tree T are vertices which have only

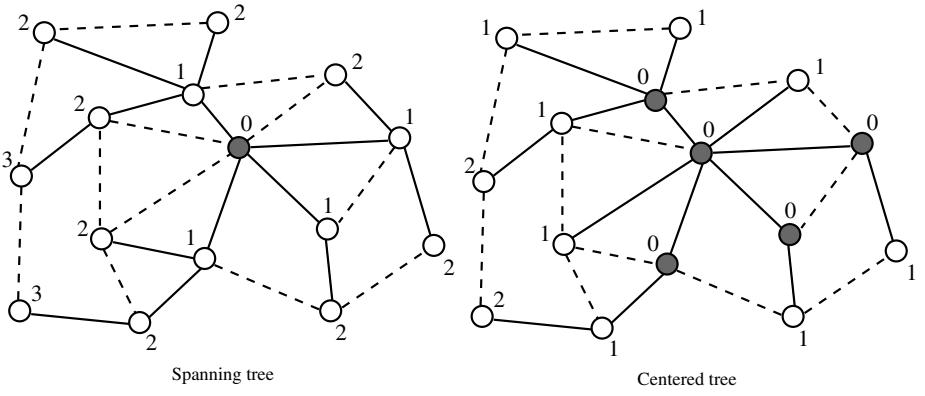


Fig. 1. Examples of spanning and centered tree in the same graph. The dark grey node in the left figure represents a root. Dark grey nodes in the right figure represent a center. Numbers indicate depths of nodes in trees. Solid edges are edges of a spanning tree. Dash edges are other edges of the graph.

one neighbor. The depth of a tree is the highest depth of its leaves. A spanning tree $T(V, F)$ of a connected undirected graph $G(U, E)$ is a tree where $V = U$ and $F \subseteq E$.

Given an edge-weighted (or node-weighted) graph $G(U, E)$ with a scoring function $w : e \in E \rightarrow \Re$ (or $w : u \in U \rightarrow \Re$). *Total weight* $w(G)$ of G is the sum of weights of all edges (or nodes) in the graph:

$$w(G) = \sum_{\forall e \in E} w(e) \quad (or \;\; w(G) = \sum_{\forall u \in U} w(u) \;\;).$$

Suppose a connected undirected graph $G(U, E)$ and a vertex $u \in U$ are given. Let $N(u)$ a set of all neighbors of u and $N'(u) \subseteq N(u)$ be. A *center* of u is the set $C(u) \equiv N'(u) \cup \{u\}$.

Observe that a center can be expanded to a spanning tree of $G(U, E)$. Moreover, the center as an initial set of expansion can be consider as a root if we merge all vertices of center to one node. Such spanning tree created from a center, called *centered tree*, has zero depth all vertices of center and the vertices of i- depth are new nodes added in ith iteration of expansion to the spanning tree. Therefore a centered tree , can be generated as follows:

- The 0-depth of the centered tree is the center
- The i-th depth of the centred tree consists of all neighbors of $(i-1)$-th depth which are not yet in any lower depth of the centered tree yet.

Examples of a spanning and centered tree are on Figure 1.

A PPI network is represented by an undirected graph $G(U, E)$. U denotes the set of proteins and E denotes set of edges, where an edge $uu' \in E$ represents the interaction between $u \in U$ and $u' \in U$. Given PPI networks $G(U, E)$ and

$H(V, F)$. A *vertex* $u \in U$ *is orthologous* if there exists at least one vertex $v \in V$ such that uv is an orthologous pair. *Orthologous articulation* is an orthologous vertex which is an articulation. An *orthology path* is a path containing only orthologous vertices.

3 From Orthologous Articulations through Centers to Trees

Given a PPI network $G(U, E)$ and the set of vertices $O \subseteq U$, which are orthologous w.r.t. the vertices of the other PPI network to be compared with G. Let $n = |O|$. We generate centers from orthologous articulations, and expand them into centered subtrees containing only orthologous proteins. The resulting algorithm, called `Divide`, is sketched in pseudo-code in Algorithm 1, and described in more detail below.

Computing Articulations (Line 1)**.** Computation of articulations can be performed in linear time by using, e.g., Tarjan's algorithm described in [23] or [24].

Greedy Construction of Centers (Lines 3-10)**.** The degree (in G) of all orthologous articulations is then used for selecting seeds for the construction of centers. Networks with scale-free topology appear to have edges between hubs systematically suppressed, while those between a hub and a low-connected protein seem favored [25]. Guided by this observation, we greedily construct centers by joining one orthologous articulation hub with its orthologous articulation neighbors, which will more likely have low degree.

Specifically, let A be the set of orthologous articulations of G. The first center consists of the element of A with highest degree and all its neighbors in A. The other centers are generated iteratively by considering, at each iteration, the element of A with highest degree among those which do not occur in any of the centers constructed so far, together with all its neighbors in A which do not already occur in any other center. The process terminates when all elements of A are in at least one center. Then an unambiguous label is assigned to each center.

Initial Expansion (Lines 11-16)**.** By construction, centers cover all orthologous articulations. Articulation hubs are often present in conserved subnets detected by means of comparative methods such as [6]. Therefore, assuming that the majority of the remaining nodes belonging to conserved modules are neighbors of articulation hubs, we add to each center all its neighboring ortholog proteins, regardless whether they are or not articulations. We perform this step for all centers in parallel.

We mark these new added proteins with the label of the centers to which they have been added. These new added proteins form the first depth centered trees.

Observe that there may be a non-empty overlap between first depth centered trees (as illustrated in the right part of Figure 2).

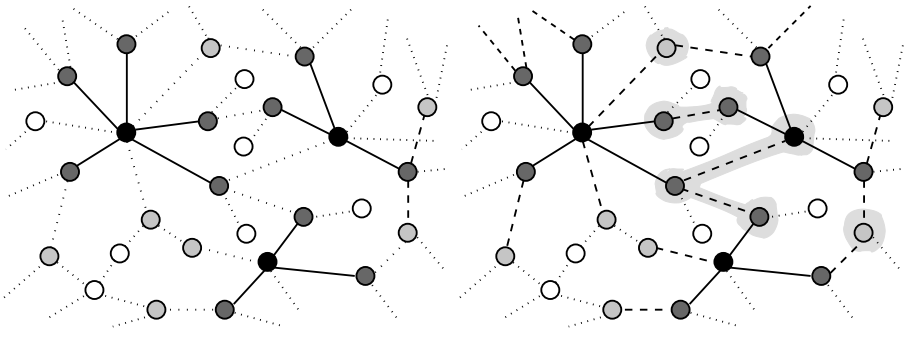


Fig. 2. Examples of centers of centered trees (left figure) and of their initial expansion (right figure). Seeds of centers are solid nodes. Dark grey nodes are the rest of centers connected to a seed by solid edges. Light grey nodes are orthologous proteins which are not articulations. Empty nodes are non-orthologous proteins. Dot edges are the rest of edges in the graph. In the second (right) graph dash edges indicate the expansion and connect nodes of centers (zero depth centered trees) with nodes of the first depth centered trees. Nodes on the grey background indicate the overlap among centered trees.

Parallel Expanding of Trees (Lines 17-27)**.** Successive depths of trees are generated by expanding all nodes with only one label which occur in the last depth of each (actual) centered tree. We add to the corresponding trees all orthologous neighbors of these nodes which are not yet labelled. Then we assign to the newly added nodes the labels of the centered trees they belong to. This process is repeated until it is impossible to add unlabeled orthologous proteins to at least one centered tree.

Observe that each iteration yields to possible overlap between newly created depths (see the left part of Figure 3).

Assigning Remaining Nodes to Trees (Lines 28-42)**.** The remaining orthologous nodes, that is, those not yet labelled, are processed as follows. First, unlabeled nodes which are neighbors of multi-labelled nodes are added to the corresponding centered trees. Then the newly added nodes are marked with these labels. This process is iterated until there are no unlabeled neighbors of multi-labelled nodes.

Nodes which are not neighbors of any labelled protein are still unlabeled. We assume that they may possibly be part of conserved complexes which do not contain articulations. Hence we create new subtrees by joining together all unlabeled orthologous neighbor proteins.

An example of these final steps is shown on the right part of Figure 3.

Complexity. The algorithm divides only orthologs of a given PPI network where the number of all orthologs is $n = |O|$. It performs a parallel breadth-first search (BFS). It general, BFS has $O(|V| + |E|)$ complexity, where V and E denote

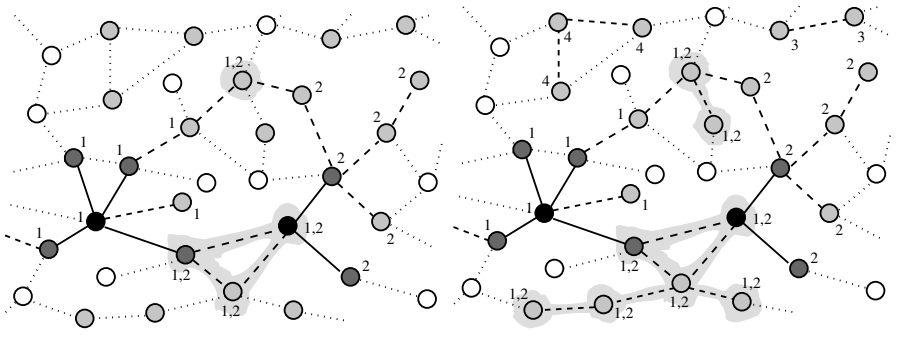


Fig. 3. Examples of parallel expansion of trees (left figure) and of the final assigning remaining nodes (right figure). Seeds of centers are solid nodes. Dark grey nodes are the rest of centers connected to a seed by solid edges. Light grey nodes are orthologous proteins which are not articulations. Empty nodes are non-orthologous proteins. Dash edges indicate the process of expansion. Dot edges are the rest of edges in the graph. Nodes on the grey background create the overlap. Numbers are labels of trees assigned to nodes during expansion.

the number of nodes and edges, respectively. However, `Divide` constructs trees considering only orthologous nodes, so the number of edges, which are traversed, is $|O'| - 1$, where $|O'|$ is the number of orthologs vertices of the constructed subtree. The possible overlap between trees can increase the number of traversed edges and visited vertices. In the worse case all orthologous vertices are visited by each center (all nodes are in the overlap). So, if the number of centers is k, the complexity of `Divide` is $O(kn)$.

4 Divide and Align Algorithm

The Divide algorithm divides orthologous proteins of the PPI network into overlapping subtrees. We separately apply this algorithm to each of the two PPI networks from the distinct species to be compared. Nodes of each constructed subtree induce a PPI subnetwork. Pairs of such induced subnetworks from different species are merged into small orthology graphs if at least two orthologous pairs exist between proteins of those subnetworks.

To this aim we use a common approach, based on the construction of a weighted metagraph between two PPI networks of different species. In this metagraph each node corresponds to an homologous pair of proteins, one from each of the two PPI networks. The metagraph is called *alignment* or *orthology graph*. Weights are assigned either to edges, like in [6], or to nodes, like in [7], of the alignment graph using a scoring function. The function transforms conservation and eventually also evolution information to one real value for each edge or node.

In our experiments we use the evolution-based alignment graph model introduced in [6]. In that model, a weighted alignment graph is constructed from a

Algorithm 1 Divide algorithm

Input: G: PPI network, O: orthologous nodes of G
Output: S: list of subsets of O

$A =$ { orthologous articulations of G}
$S =<>$
repeat {Construction of centers}
  $root =$ element of A with highest degree not already occurring in S
  $s = \{root\} \cup$ { neighbors of $root$ in A not already occurring in S}
  $S =< s, S >$
until all members of A occur in S
$d = 0$
Assign depth d to all elements of S
Assign label l_s to each s in S and to all its elements
for s in S **do**
  $s = s \cup$ { all neighbors of s in O}
  Assign label l_s to all neighbors of s in O
end for
$d = 1$
Assign depth d to all elements of S having yet no depth assigned
repeat {Expand one depth centered trees from nodes with one label}
  $N =$ { unlabeled neighbors in O of elements in s of depth d having only one label }
  for n in N **do**
    Assign to n all labels of its neighbors of depth d having only one label
    for $l_s \in n$ **do**
      $s = s \cup \{n\}$
    end for
  end for
  $d = d + 1$
  Assign depth d to all elements of S having yet no depth assigned
until S does not change
repeat {Expand centered trees from nodes multiple labels}
  $R =$ { unlabeled proteins in O with at least one multi-labelled protein as neighbor }
  for r in R **do**
    Assign to r all labels of its neighbors
    for $l_s \in r$ **do**
      $s = s \cup \{r\}$
    end for
  end for
until S does not change
repeat
  choose an unlabeled element u of O
  $t = \{u\} \cup$ {all elements of O which can be reached alongside an orthology path from u}
  Assign label l_t to t and to all its elements
  $S =< t, S >$
until O does not contain any unlabeled node

pair of PPI networks and a similarity score S, which quantifies the likelihood that two proteins are orthologous. A node in the alignment graph is a pair of ortholog proteins. Each edge in the alignment graph is assigned a weight that is the sum of three scoring terms: for protein duplication, mismatches for possible divergence in function, and match of a conserved pair of orthologous interactions. We refer to [6] for a detailed description of these terms. Induced subgraphs of the resulting weighted alignment graph with total weight greater than a given threshold are considered as relevant *alignments*. This problem is reduced to the optimization problem of finding a maximal induced subgraph. In [6], an approximation greedy algorithm based on local search is used because the maximum induced subgraph problem is NP-complete. This greedy algorithm selects at first

one seed which can likely contribute at most to the overall weight of a potential subgraph. Such seed is expanded by adding (removing) nodes to (from) the subgraph while the actual subgraph weight increases.

In this study, after the diving step and aligning possible pairs of PPI subnetworks a set of small alignment graphs is produced. We use exact optimization [26] for searching in those graphs. We call the resulting algorithm `DivA` (Divide and Align).

Finally, redundant alignments are filtered out as done, e.g., in [6]. A subgraph G_1 is said to be *redundant* if there exists another subgraph G_2 which contains $r\%$ of its nodes, where r is a threshold value that determines the extent of allowed overlap between discovered protein complexes. In such a case we say that G_1 *is redundant for* G_2.

5 Experimental Results

In order to assess the performance of our approach, we use the state-of-the-art framework for comparative network analysis proposed in [6], called `MaWish`. The two following PPI networks, already compared in [6], are considered: *Saccharomyces cerevisiae* and *Caenorhabditis elegans*, which were obtained from BIND [1] and DIP [2] molecular interaction databases. The corresponding networks consist of 5157 proteins and 18192 interactions, and 3345 proteins and 5988 interactions, respectively. All these data are available at the webpage of `MaWish`[1]. Moreover, the data already contain the list of potential orthologous and paralogous pairs, which are derived using BLAST E-values (for more details see [11]). 2746 potential orthologous pairs created by 792 proteins in S. cerevisiae and 633 proteins in C. elegans are identified.

5.1 Divide Phase

Results of application of the Divide algorithm to these networks are summarized as follows.

For *Saccharomyces cerevisiae*, 697 articulations, of which 151 orthologs, and 83 centers are identified. Expansion of these centers into centered trees results in 639 covered orthologs. The algorithm assigns the remaining 153 orthologous proteins to 152 new subtrees.

For *Caenorhabditis elegans*, 586 articulations, of which 158 orthologs, are computed, and 112 centers are constructed from them. Expansion of these centers into centered trees results in 339 covered orthologs. The algorithm assigns the remaining orthologous 294 proteins to 288 new subtrees.

We observe that the last remaining orthologs assigned to subtrees are 'isolated' nodes, in the sense that they are rather distant from each other and not reachable from ortholog paths stemming from centers.

The divide part of algorithm run only less than half of a second on a desktop machine (AMD Athlon 64 Processor 3500+, 2 GB RAM) in practical.

[1] www.cs.purdue.edu/homes/koyuturk/mawish/.

5.2 Alignment Phase

We obtain 235 subtrees for *Saccharomyces cerevisiae* and 400 subtrees of *Caenorhabditis elegans*. Nodes of each such tree induce a PPI subnetwork. By constructing alignment graphs between each two PPI subnetworks containing more than one ortholog pair, we obtain 884 alignment graphs, where the biggest one consists of only 31 nodes. For each of such alignment graphs, the maximum weighted induced subgraph is computed by exact optimization. Zero weight threshold is used for considering an induced subgraph a legal alignment. Redundant graphs are filtered using $r = 80\%$ as the threshold for redundancy. In this way `DivA` discovers 72 alignments.

The computation of induced subgraphs by an exact search took a few minutes compared to around a second in `MaWish` on a desktop machine (AMD Athlon 64 Processor 3500+, 2 GB RAM).

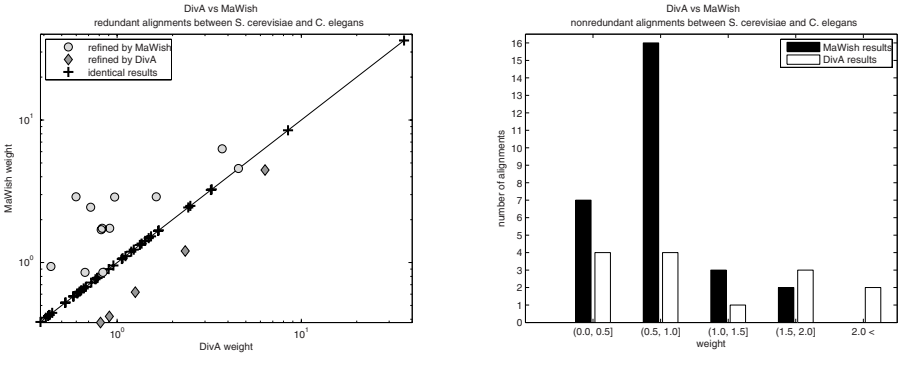


Fig. 4. Left figure: Distribution of pairs of weights of paired redundant alignments, one obtained from `MaWish` and one from `DivA`. Weights of alignments found by `DivA` are on the *x-axis*, those found by `MaWish` on the *y-axis*. Right figure: Interval weight distributions of non-redundant alignments discovered by `MaWish` (solid bars) and `DivA` (empty bars). The x-axis show weight intervals, the *y-axis* the number of alignments in each interval.

5.3 Comparison between `DivA` and `MaWish`

We performed network alignment with `MaWish` using parameter values as reported in [11]. The algorithm discovered 83 conserved subnets.

A *paired redundant alignment* is a pair (G_1, G_2) of alignments, with G_1 discovered by `DivA` and G_2 discovered by `MaWish`, such that either G_1 is redundant for G_2 or vice versa. For a paired redundant alignment (G_1, G_2) we say that G_1 *refines* G_2 if the total weight of G_1 is bigger than the total weight of G_2.

`DivA` finds 14 new alignments not detected by `MaWish`. Figure 5 shows the best new alignment found by `DivA` (left) and the alignment of `DivA` which best refines an alignment of `MaWish`.

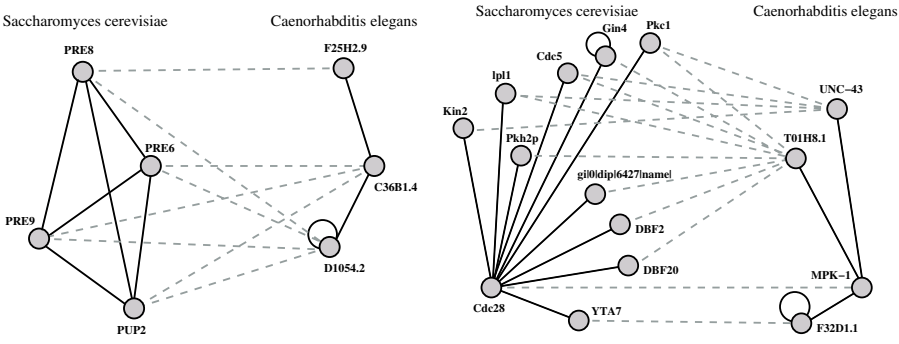


Fig. 5. Left: The best new alignment. Dash lines mark orthologous pairs. Solid line is protein-protein interaction. Right: The refined alignment with the greatest weight. Dash lines mark orthologous pairs. Solid line is protein-protein interaction. A loop on a protein means duplication.

There are 58 paired redundant alignments, whose total weights are plotted in the left part of Figure 4. Among these, 40 (55.6%) are equal (crosses in the diagonal), and 18 (25%) different. 5 (6.9%) (diamonds below the diagonal) with better `DivA` alignment weight, and 12 (16.7%) (circles above the diagonal) with better `MaWish` alignment weight (for 1 pair it is undecidable because of rounding errors during computation).

The right plot of Figure 4 shows the binned distribution of total weights of the 14 (19.4%) found by `DivA` but not `MaWish`, and 28 found by `MaWish` and not by `DivA`. The overall weight average of the `DivA` ones (1.197) is greater than the overall average of the `MaWish` ones (0.7501), indicating the ability of `DivA` to find high score subnets, possibly due to the exact search strategy used.

Of the 14 new alignments detected by `DivA`, 8 of them have a intersection with a true MIPS complex (cf. Table 1). Three of these alignments (6., 12. and 14.) have equal (sub)module in their true S. cerevisiae complex.

Table 1. HG= hypergeometric, Size = number of alignment nodes of an alignment, N = number of proteins of alignment nodes which are annotated in the best (according to hypergeometric score) true S. cerevisiae's MIPS complex of the alignment. M = number of proteins of alignment nodes in S. cerevisiae. Intersection = $|N|/|M|$.

Align.	Score	Size	$\|M\|$	MIPS category	Intersection	$-log(HG)$
1.	4.28	8	4	20S proteasome	100(%)	7.25
4.	1.65	5	2	19/22S regulator	100(%)	3.45
6.	1.41	5	2	19/22S regulator	50(%)	1.71
7.	0.62	2	2	20S proteasome	100(%)	3.56
8.	0.61	2	2	Replication fork complexes	100(%)	3.22
9.	0.53	2	2	19/22S regulator	100(%)	3.45
12.	0.43	2	2	19/22S regulator	50(%)	1.71
14.	0.39	2	2	19/22S regulator	50(%)	1.71

Table 2. True complexes associated to `MaWish` refined alignments

Align.	Score	Size	$\|M\|$	MIPS category	Intersection	$-log(HG)$
1.	4.46	10	10	Cdc28p complexes	10(%)	1.47
2.	0.62	2	2	Casein kinase II	100(%)	4.81
3.	0.38	2	2	SNF1 complex	50(%)	2.16

Table 3. True complexes associated to `DivA` refined alignments

Align.	Score	Size	$\|M\|$	MIPS category	Intersection	$-log(HG)$
1.	6.35	15	11	Cdc28p complexes	9(%)	1.47
2.	1.26	4	4	Casein kinase II	100(%)	10.39
3.	0.81	3	2	SNF1 complex	50(%)	2.16

From the refined alignments, three of them have intersection with a true MIPS complex.

Note that alignments 1. and 3. in both Table 2 and 3 have equal hypergeometric score, showing that the coverage, that is, number of proteins of an alignment contained in its best true MIPS module, does not change. Alignment 2. in Table 2 covers 50% of the true complex, while its refinement in Table 3 covers the entire true complex (Casein kinase II, consisting of 4 proteins).

Three of these alignments have equal (sub)module in their true S. cerevisiae complex.

By considering the union of all alignments of `MaWish` and `DivA` and by filtering out the redundant ones, 97 alignments are obtained, from which 26% consist of new or refined `DivA` ones. In particular, conserved modules of three new true MIPS classes are detected: replication fork complexes, mRNA splicing, SCF-MET30 complex. Moreover, the alignment by `MaWish` which covers 50% of the true complex Casein kinase II (this complex consists of 4 proteins) is refined by `DivA` in such a way that the entire true complex is covered (all four proteins).

In this experiment we searched for the best solution in each orthology graph only. A full-search, where all possible solutions are found for each orthology graph, has been used in [27]. This yielded to a considerable increase of the number of results. Statistical evaluation of those results indicated their biological relevance. In general, the results show that `DivA` can be successfully applied to 'refine' state-of-the-art algorithms for network alignment.

6 Conclusion

The comparative experimental analysis with `MaWish` indicates that `DivA` is able to discover new alignments which seem to be on average more conserved because of higher weight than those discovered by `MaWish` but not by `DivA`. Improved performance is shown to be achieved by combining results of `MaWish` and `DivA`, yielding new and refined alignments.

The selection of centers is biased on the orthology information but it can be changed for another property. Hence the divide algorithm can be applied to perform modular network alignment of other type of networks.

Finally, we considered here an instance of our approach based on the evolution-based alignment graph model by Koyuturk et al. [11]. We intend to analyze instances of our approach based on other methods, such as [7].

Acknowledgments

We would like to thank Mehmet Koyuturk for discussion on the `MaWish` code.

References

1. Bader, G.D., Donaldson, I., Wolting, C., Ouellette, B.F.F., Pawson, T., Hogue, C.W.V.: Bind–the biomolecular interaction network database. Nucleic Acids Res. 29(1), 242–245 (2001)
2. Xenarios, I., Salwínski, Ł., Duan, X.J., Higney, P., Kim, S.M., Eisenberg, D.: Dip, the database of interacting proteins: a research tool for studying cellular networks of protein interactions. Nucleic Acids Research 30(1), 303–305 (2002)
3. 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 100, 11394–11399 (2003)
4. Sharan, R., Ideker, T.: Modeling cellular machinery through biological network comparison. Nature Biotechnology 24(4), 427–433 (2006)
5. Srinivasan, B.S., Shah, N.H., Flannick, J., Abeliuk, E., Novak, A., Batzoglou, S.: Current Progress in Network Research: toward Reference Networks for kKey Model Organisms. Brief. in Bioinformatics (Advance access, 2007)
6. Koyutürk, M., Grama, A., Szpankowski, W.: Pairwise local alignment of protein interaction networks guided by models of evolution. In: Miyano, S., Mesirov, J., Kasif, S., Istrail, S., Pevzner, P.A., Waterman, M. (eds.) RECOMB 2005. LNCS (LNBI), vol. 3500, pp. 48–65. Springer, Heidelberg (2005)
7. Sharan, R., Ideker, T., Kelley, B.P., Shamir, R., Karp, R.M.: Identification of protein complexes by comparative analysis of yeast and bacterial protein interaction data. Journal of Computional Biology 12(6), 835–846 (2005)
8. Hirsh, E., Sharan, R.: Identification of conserved protein complexes based on a model of protein network evolution. Bioinformatics 23(2), 170–176 (2007)
9. Sharan, R., Suthram, S., Kelley, R.M., Kuhn, T., McCuine, S., Uetz, P., Sittler, T., Karp, R.M., Ideker, T.: From the Cover: Conserved patterns of protein interaction in multiple species. Proceedings of the National Academy of Sciences 102(6), 1974–1979 (2005)
10. Flannick, J., Novak, A., Srinivasan, B.S., McAdams, H.H., Batzoglou, S.: Graemlin: General and robust alignment of multiple large interaction networks. Genome Res. 16(9), 1169–1181 (2006)
11. Koyutürk, M., Kim, Y., Topkara, U., Subramaniam, S., Grama, A., Szpankowski, W.: Pairwise alignment of protein interaction networks. Journal of Computional Biology 13(2), 182–199 (2006)

12. Pržulj, N.: Knowledge Discovery in Proteomics: Graph Theory Analysis of Protein-Protein Interactions. CRC Press, Boca Raton (2005)
13. Singh, R., Xu, J., Berger, B.: Pairwise global alignment of protein interaction networks by matching neighborhood topology, pp. 16–31 (2007)
14. Pržulj, N., Wigle, D., Jurisica, I.: Functional topology in a network of protein interactions. Bioinformatics 20(3), 340–384 (2004)
15. Rathod, A.J., Fukami, C.: Mathematical properties of networks of protein interactions. CS374 Fall 2005 Lecture 9, Computer Science Department, Stanford University (2005)
16. Jeong, H., Mason, S.P., Barabasi, A.L., Oltvai, Z.N.: Lethality and centrality in protein networks. NATURE v 411, 41 (2001)
17. Ekman, D., Light, S., Björklund, A.K., Elofsson, A.: What properties characterize the hub proteins of the protein-protein interaction network of saccharomyces cerevisiae? Genome Biology 7(6), R45 (2006)
18. Ucar, D., Asur, S., Catalyurek, U., 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)
19. Bader, G.D., Lssig, M., Wagner, A.: Structure and evolution of protein interaction networks: a statistical model for link dynamics and gene duplications. BMC Evolutionary Biology 4(51) (2004)
20. Li, X.L., Tan, S.H., Foo, C.S., Ng, S.K.: Interaction graph mining for protein complexes using local clique merging. Genome Informatics 16(2), 260–269 (2005)
21. Yook, S.H., Oltvai, Z.N., Barabsi, A.L.: Functional and topological characterization of protein interaction networks. PROTEOMICS 4, 928–942 (2004)
22. Abou-Rjeili, A., Karypis, G.: Multilevel algorithms for partitioning power-law graphs. In: 20th International Parallel and Distributed Processing Symposium (IPDPS) (2006)
23. Tarjan, R.: Depth-first search and linear graph algorithms. SIAM Journal on Computing 1(2), 146–160 (1972)
24. Hopcroft, J., Tarjan, R.: Algorithm 447: efficient algorithms for graph manipulation. Commun. ACM 16(6), 372–378 (1973)
25. Maslov, S., Sneppen, K.: Specificity and stability in topology of protein networks. Science 296, 910–913 (2002)
26. Wolsey, L.A.: Integer Programming, 1st edn. Wiley, Chichester (1998)
27. Jancura, P., Heringa, J., Marchiori, E.: Divide, align and full-search for discovering conserved protein complexes. In: Marchiori, E., Moore, J.H. (eds.) EvoBIO 2008. LNCS, vol. 4973, pp. 71–82. Springer, Heidelberg (2008)

Constraint Minimization for Efficient Modeling of Gene Regulatory Network

Ramesh Ram[1], Madhu Chetty[1], and Dieter Bulach[2]

[1] Gippsland School of IT, Monash University,
Churchill, Victoria 3842, Australia
{Ramesh.ram,Madhu.chetty}@infotech.monash.edu.au
[2] CSIRO Livestock Industries, Australian Animal Health Laboratory,
5 Portarlington Rd, Geelong VIC 3220, Australia
Dieter.Bulach@csiro.au

Abstract. Due to various complexities, as well as noise and high dimensionality, reconstructing a gene regulatory network (GRN) from a high-throughput microarray data becomes computationally intensive. In our earlier work on causal model approach for GRN reconstruction, we had shown the superiority of Markov blanket (MB) algorithm compared to the algorithm using the existing Y and V causal models. In this paper, we show the MB algorithm can be enhanced further by application of the proposed constraint logic minimization (CLM) technique. We describe a framework for minimizing the constraint logic involved (condition independent tests) by exploiting the Markov blanket learning methods developed for a Bayesian network (BN). The constraint relationships are represented in the form of logic using K-map and with the aid of CLM increase the algorithm efficiency and the accuracy. We show improved results by investigations on both the synthetic as well as the real life yeast cell cycle data sets.

Keywords: Causal model, Markov blanket, Constraint minimization, Gene regulatory network.

1 Introduction

Gene regulatory networks (GRNs) represent gene-gene regulatory interactions in a genome to display relationships between various gene activities. Amongst different approaches available, these networks can also be modeled accurately by Markov blanket (MB) graph a powerful versatile method for modeling any dynamic physical system. This technique was first proposed by Sprites et al [1] who stated that MB can adequately represent all connections and interactions in a network. Since then, the work on MB has been rapidly expanding with a focus on the study of causality which plays an important role in modeling, analysis and design of GRNs. Learning any Markov blanket Bayesian network structure and inferring gene networks [2, 3] involves application of constraints. Although these constraints are typically conditional-independence statements, the non-independence based constraints may also be entailed by the structure where latent variables exist [3]. The conditional independence tests

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 201–213, 2008.

used in practice are statistical tests such as partial correlation, mutual information, and conditional probabilities etc. that indicate a causal influence. In order to use the conditional independence tests to reconstruct the structure, several assumptions have to be made, e.g. causal sufficiency, causal Markov and faithfulness [2]. With these assumptions, we can ascertain the existence of an edge, its direction and whether it is positive or negative. The Sprites-Glymour-Scheines (SGS) algorithm [1], used for obtaining a causal DAG from a dataset, assumes that graphs are acyclical. It is formulated using the concept of d-separation [2] in which all possible combinations are tried before determining the existence of an edge between every pair of variables in the dataset. However, the SGS algorithm fails to always assign directions to each of the edges. This limitation of SGS algorithm is overcome by the inductive causation (IC) algorithm [3], which is capable of assigning directions. Some algorithms do not make use of independence tests but take into account d-separation in order to discover structure from data [2]. For example, Cheng *et al* [4], applied mutual information instead of conditional independence tests. All these algorithms are referred as constraint-based algorithms [1, 4]. Constraint-based algorithms have certain limitations such as poor robustness or computation time which increases exponentially with the number of constraints. These limitations make these approaches impractical for large datasets of tens or even hundreds of variables.

In our recently proposed causal model [5, 6] approach for constructing GRN , the network was inferred by applying the following three sequential steps to identify the sub-structures of a larger network: i) Perform conditional independence (CI) tests for each node's Markov blanket ii) Assign direction to the edges and iii) Assign sign of regulation to the edges. However, due to the huge size of network search space and the limited amount of microarray data, it was impractical to test each and every constraint. Moreover, with the increase in the condition set needed for causal discovery, more and more CI tests had to be performed, resulting eventually in lower accuracy.

By simplifying the complex logic involved with the constraints in the Markov blanket algorithm, the computational efficiency of the MB algorithm [6] can be enhanced thereby resulting in improved accuracy for network reconstruction. In this paper, we propose a technique for minimizing the constraints and hence the condition set needed for testing the structure with respect to data. The statistical tests following the logic is translated into a Boolean function after which a logic gate minimization technique such as K-map [7] is applied and the minimized logic is translated back to the constraints and used on the data. We have achieved this by a novel independence-based algorithm which we refer here as the Markov blanket-Constraint Logic Minimisation (MB-CLM) algorithm. The MB-CLM algorithm heuristically uses Markov Blanket neighborhood of a node and makes model evaluation simple. In order to evaluate and validate a Markov Blanket, there is invariably a need for checking a set of conditions. However, from the available set of alternatives, it is possible to have a potentially smaller set of conditions that can establish the desired conclusion for the given network but with a faster computation speed and increased reliability. This is because a conditioning set S splits the data set into 2^S partitions. With a smaller conditioning set, the data set is split into larger partitions thereby making dependence tests more reliable. This smaller or minimal set will fulfill the necessary and sufficient conditions required for GRN reconstruction.

The rest of the paper is organized as follows. Section 2 provides a background on causal model and also explains the fundamentals of the technique of logic minimization. Section 3 describes the methodology. Section 4 gives the results of the experiments from the synthetic and real datasets. Finally, Section 5 provides conclusion and the future work.

2 Background

In this section, we briefly present the causal model approach for GRN reconstruction and also the notion of Markov blanket both of which are important for understanding the MB-CLM algorithm.

2.1 Causal Model Approach

A causal GRN structure is represented by a directed graph whose nodes represent the genes and the directed edges between nodes indicate the causal relationships. Pearl *et al* and Sprites *et al* [1, 2] proposed algorithms to infer a causal structure from experimental data by using partial correlations, if the underlying causal structure is a directed acyclic graph (DAG). Recently, we reported a technique for causal modeling by means of a novel scoring function [5]. In this work, the central step of determining the fitness of the data given a whole network, is decomposed into a task of determining a set of scores of the local models that includes: i) Fitness of structure ii) Direction of causality and iii) Sign (positive/ negative) of regulation. The task of network reconstruction is cast into a search for candidate gene networks with high scores. This highly computationally expensive search is usually carried out stochastically by using, for example a genetic algorithm (GA). The search creates and evolves different networks and eventually obtains a network that best fits the microarray data. Due to the stochastic nature of the GA, the GA is repeated several times and the resulting network structures are combined in a predefined manner to reconstruct the final gene network. While evaluating the fitness, the putative network is actually decomposed into MB and conditional independence tests are applied in order to detect whether or not connections are direct or indirect. The direction and sign of regulation are recovered by estimating the time delay and correlation between expression profiles of pairs of genes. The entire methodology has already been validated by using a synthetic dataset reported in our earlier work [6] and *Saccharomyces cerevisiae* (yeast) [8] microarray dataset. The results of validation are found to be in agreement with the known biological findings.

2.2 Markov Blanket

A Markov blanket [2], central to the concept of causal modeling, includes the node X under consideration and also its parents and children. It is denoted as MB(X) and is a minimal set of variables such that every other variable is independent of X given MB(X), i.e.

$$\forall Y \in \{X_1,\ldots, X_n\} \setminus \{MB(X), X\}, X \perp\!\!\!\perp Y \mid MB(X)$$

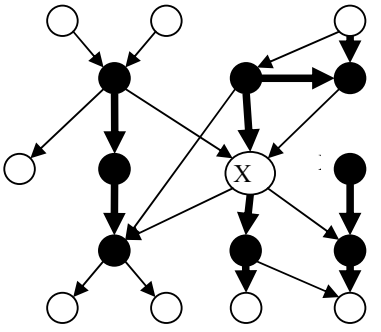


Fig. 1. Markov Blanket of *X*

An example of MB is shown in Fig. 1. Several studies [9-11] have sought to identify the MB of a target node (*X* in Fig. 1) by filtering nodes using statistical decisions from conditional independence tests. In Fig.1, the shaded nodes (black) are Markov blanket neighbours since there is either an edge or a child in common between target node and shaded node. The non-shaded nodes (white) are independent of the target node, *X*. A MB DAG can be constructed by combining the MB's of all the nodes in the dataset.

A MB for a node *X* in a GRN dataset has two important features. First, all the nodes within a MB have a similar set of dependencies and therefore exhibit a similar behavior. In a similar manner, genes in a cell are also organized into small groups and the sets of genes required for a similar biological function or response are co-regulated by the same inputs in order to coordinate their joint activity. In other words, the MB neighbours (shaded nodes in Fig. 1) of a target node (gene) show the gene expression patterns emerging only due to a disruption of that gene. Second, they can also have a causal interpretation: a directed edge from one gene to another, $X \rightarrow Y$, represents the claim that *X* is a direct cause of *Y* with respect to other genes in a DAG. Keeping other genes fixed, if *X* is varied by an intervention (e.g., activation or repression), then both *X* and *Y* would co-vary [1, 2]. A MB DAG can thus provide both biological and causal insight into relations between a reduced set of predictor nodes (parents, children, spouses) and the target node.

The technique of logic gate minimisation, well known for electronic circuit minimisation, and on which the proposed constraint logic minimisation algorithm is based, is presented next.

3 Logic Gate Minimization Technique

The proposed CLM algorithm is uses the K-map technique applied for logic gate minimisation. To illustrate the minimisation technique, let us consider an arbitrarily chosen four input Boolean network. Let the network be, for example, characterized by the following Boolean function to give an output of 1(i.e. true output):

$$f(a, b, c, d) = \sum m\,(0, 3, 4, 7, 8, 11, 15) \quad (1)$$

ab \ cd	00	01	11	10
00	1		1	
01	1		1	
11			1	
01	1		1	

Fig. 2. K-Map for the function given by Eqn. (1)

Here, f is the boolean function. a, b, c and d are the four independent inputs. The numbers on the RHS are minterms (i.e. decimal value equivalent of the 4 bit inputs). For example, the value 3 on RHS, means that the four bit input combination of 0011 (i.e. a'b'cd or the decimal equivalent value of 3) results in 1. *m* indicates that all the values within bracket are minterms.

The above equation indicates that any combination of the inputs with values of either 0, 3, 4, 7, 8, 11 or 15 would result in a true output. Since the default value of *a* is considered as 1, it implies that a'=0. Thus, the above function in Eqn. 1 can be expanded as

$$f=a'b'c'd'+a'b'cd+a'bc'd'+ a'bcd+abc'd'+ab'cd+abcd \quad (2)$$

The above equation is known as a Sum of Product (SOP) equation and the products are the minterms mentioned above.

The K-map for the above function is shown in Fig. 2. All rows and columns in the K-map above are unique since only one variable changes its value within its square. The relevant K-map elements are given a value of 1 to include all possible constraints with true outputs. The first row, for example has input a = 0 (i.e. a') and input b = 0 (i.e. b'). Similarly column 3, for example has both c = 1 and d =1. Thus, an element, for example in row 1, column 3 corresponds to input a'b'cd =1. It can be noted that this corresponds to the second term on RHS in Eqn. (2) above. It can be further noted that between two adjacent columns (or rows), only one of the variables changes its value. For example, in Fig. 2, the input cd given as 00, 01, 11, 10 in the columns ensures that there is only one input change.

Now let us consider grouping the common terms and minimisation of the function using the K-map shown in Fig. 2. By grouping:

i) Four 1 in column 3 (all rows), we get the common term cd
ii) Two 1 in column 1 (row 1 and row 2), we get common term a'c'd'

Considering the above groupings, we can rearrange the RHS terms from Eqn.(2) appropriately to facilitate logic minimisation. Further, noting that

$$a+a'=1 \quad (3)$$

We can simplify Eqn. 2 as follows:

$$\begin{aligned} f(a,b,c,d) &= (a'b'cd+abcd+a'bcd+ab'cd)+(a'b'c'd'+a'bc'd') +ab'c'd' \\ &= cd(a'b'+ab+a'b+ab')+ a'c'd' (b+b')+ ab'c'd' \\ &= cd + a'c'd' + ab'c'd' \end{aligned} \quad (4)$$

Since the constraint when applied to the MB scoring for GRN reconstruction will also evaluate to either true or false, the principles of logic gate minimisation presented in this section can easily be extended and applied to GRN modeling. The variables a, b, c, d in Eqn. (4) above will correspond to constraints that can be either CI tests or tests involving delays and directions. This technique is presented next.

4 Method

In our GRN reconstruction method reported earlier [6, 12], the network is evaluated at the MB of every node with respect to data resulting in a set of constraints to be satisfied per MB. In general, all these constraints should always be satisfied to validate a true MB with respect to data. Since the dataset under consideration is noisy and high dimentional, it is acceptable if all the constraints are not necessarily satisfied for MB validation. For example, consider a MB having say three constraints. A combination of say two constraints may leave the evaluation of third constraint (*don't care*) unnecessary. However, if the two constraints fail, only then the need to evaluate the third constraint may arise. Since the Markov blanket scoring can be viewed as a logic circuit minimisation, we can get a function similar to Eqn. 1 and the underlying logic constraints can thus be represented using K-map explained in the previous section for optimising the computations. In order to show how this can be achieved, the algorithm for learning MB is presented next.

4.1 The Markov Blanket Network Inference Algorithm

A static causal directed acyclic graph (CDAG) model for representing GRN consists of nodes representing genes and arc giving direction and sign of regulation. A matrix element E(a,b) of the gene expression matrix E indicates the expression ratio of gene *a* at time *b*. The overall inference approach (*Learn_MB* algorithm) is as follows:

i) *Gene Expression Matrix E*: Obtain E corresponding to the set of nodes from dataset D that are affected by node X. This set involves parents, children and spouse nodes of node X.

ii) *Causal relation R*: In the putative MB network H(X), the causal relationships are defined as gene *a* affects gene *b* either directly or indirectly. We thus create *n* binary causal relation R using the causal relationship.

iii) *Adjacency matrix A*: The adjacency matrix A is derived directly from the binary relation R. If there is a relation that gene *a* affects gene *b*, then the value of element (a, b) in the adjacency matrix A is set to 1, i.e. A(a, b) = 1.

iv) *Skeleton matrix S*: A skeleton matrix S includes direct and indirect effects observed in a putative MB. The adjacency matrix A (of size $n \times n$ where n is the number of nodes in the MB) includes direct relationship between genes. The indirect effects are included as follows: The row i and column j in adjacency

matrix A and skeleton matrix S represent direct and indirect causal relationship between gene *i* and gene *j* respectively. For example, consider an indirect relationship between gene *i* and *k* (where *k* is any other column corresponding to gene *k*) such that A(i, j) and A(j, k) both equal 1. Then S(i,k) is set to Binary{A(i, j) AND A(j, k)} (for k = 1, . . , n). In this manner, all indirect effects are captured in the skeleton matrix S from the adjacency matrix A.

v) *Constraints set C*: The direct and indirect effects from adjacency matrix A and skeleton matrix S are converted as conditional dependence (CD) and conditional independence (CI) constraints respectively. A conditioning set is needed for each of the constraints which will contain all the nodes in H(X) minus the variables involved in the constraint. For example, if there is an indirect relation, such that gene *a* and *b* are conditionally independent then the condition set is given as H(X)-{a,b}. The outcome of the CI and CD constraints is either a 1 (constraint fits the data) or a 0 otherwise. The test is done using statistical methods namely partial correlation. The constraints that are not CI or CD determine the direction and sign of the arcs in the MB and are similar in nature to independence tests. In our case, the direction and sign between gene *a* and *b* is obtained by the following two equations:

$$f(dir_H(a,b), dir_D(a,b)) = 0 \mid 1 \tag{5}$$

$$f(\mathrm{sgn}_H(a,b), \mathrm{sgn}_D(a,b)) = 0 \mid 1 \tag{6}$$

In Eqn. (5), the function f compares the direction between the putative network H and the dataset D while in Eqn. (6), f compares the sign. Furthermore, there are additional constraints which compare estimated time delay with the actual time delay. All these constraints comprise the constraint set C.

vi) *Constraint set reduction*: If a relation exists such that gene *b* and *d* are conditionally independent (conditioned on gene *a*) and further gene *c* and *d* are conditionally independent (conditioned on gene *a*), then gene *b* and *c* are conditionally dependent (conditioned on gene *a*). Such tests are therefore unnecessary to implement and can be eliminated from the constraint set C resulting in updating the Adjacency matrix A and the Skeleton matrix S. Further, the condition set for the CI and CD constraints is also reduced such that the CI/CD test outcome is independent of the removal of a variable from the condition set and is in conformance with d-separation theory [2].

vii) *Constraints Evaluation*: Next, a table of constraint set C is created. Here, the combination of constraints that entail the validity of the putative Markov blanket with respect to the dataset or otherwise are computed. A threshold value (explained in Section 4.2 below) is used when constraints are tested with respect to data.

viii) *Fitness Score*: Comparison based on the value of each element in the skeleton matrix and the evaluation table determines the goodness of fit. This will show if the putative network is consistent with the experiment data D and the causal relation R.

4.2 MB Threshold Setting

We now discuss the factors involved in determining the threshold value used in step vii) of the MB algorithm presented above. The p-values, commonly used in any statistical analysis, are used for determining the threshold. For large p-values, the Learn_MB algorithm begins to rapidly increase the number of false positives without any corresponding increase in true positives. An appropriate value for the MB threshold, producing a near optimal result, can be selected a-priori using the Bonferroni-corrected p-value based on the number of potential network interactions. Alternatively, the threshold can be identified by analyzing the distribution of MB scores as a function of the length of the shortest path connecting each gene pair (degree of connectivity). The algorithm depends on the MB being enriched for directly matching interaction among genes and decreases rapidly with its distance from the hub. There is no unique choice for the threshold which can separate directly and indirectly interacting genes, and most methods that attempt to use a single threshold either recover many indirect connections or miss a substantial number of direct ones.

4.3 Complexity Analysis and Discussion of the Plain MB Algorithm

The order of complexity for each conditional dependence/independence test taken is $O(nD)$, where D is the dataset of input to the algorithm. The computations are required for constructing the table of constraints and for each combination of the variables (genes) included in the constraint test that exists in the data set. As a worst case scenario, each dependence test uses $O(D)$ space to store each variable combination of the conditioning constraint set that appears in the data.

The number of constraints tested is usually reported as a measure of the performance of Bayesian net reconstruction algorithms [1, 4]. To determine the number of tests in this algorithm, we assume that the steps 2 and 3 go through MB variables (parents, children, spouses) in an unspecified but fixed order. Therefore, the order of the entire algorithm is $O(n)$ in the number of independence tests. The algorithm benefits by further computational optimizations from constraint minimization using the proposed K-map technique.

In the next section, we present the CLM algorithm which when combined with learn_MB algorithm presented earlier results in an integrated MB-CLM algorithm.

4.4 CLM Algorithm

The Constraint minimization approach is given as follows:

1. Obtain the Markov blanket H(X). Let the set of constraints be C.
2. Get the constraint set C from step v) of Learn_MB algorithm in section 3.1
3. Assign binary codes for constraints in constraint set C. Use the constraint evaluation table to generate a truth table and the logic diagram.
4. Perform minimization with the help of K-map.
5. Remove unnecessary constraints before performing constraint evaluation.
6. Execute the minimized logic on the dataset D.

```
1. C ← 0
2. LEARN_MB Step F → C
3. CLM phase ← C
4. LEARN_MB Step G ← C min
```

Fig. 3. Learn_MB and CLM integration step

In Fig. 3, C represents the set of constraints. Initially, the constraints are obtained from the Learn_MB algorithm. The above mentioned CLM approach, shown as a CLM phase, takes the constraint set C as input and returns a minimized set C_{min} back to the learn_MB algorithm for evaluation and validation.

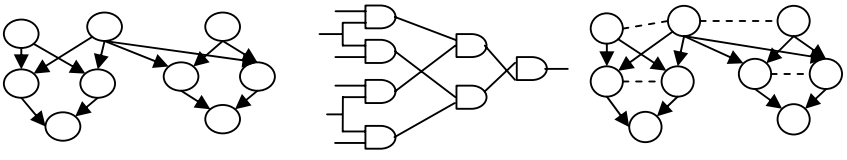
5 Experiments and Results

For examining the effect of the minimization technique on the GRN reconstruction, simulations are next conducted using both, the synthetic and the real datasets. The synthetic datasets are realistic and are generated by systematic approach reported earlier [12] for synthetic GRN modeling. The real life data set chosen for investigations is the widely studied yeast cell cycle data set.

5.1 Synthetic Datasets

Figure 4 shows an example of reconstruction of an artificially constructed synthetic network using MB-CLM technique. Figure 4a shows the original synthetic network. Amongst various network architectures possible, we chose a network type referred as random network. The generated network is of 3x3 dimensions with an up/down branching factor of 2. The branching factor refers to the number of parents, children and spouses connected to a node. The up branching factor specifies the number of parents of each node directly above it, excluding nodes in the boundary of the network as they are exogenous (without parents). Figure 4b shows the logic circuit corresponding to the constraints involved and Fig. 4c shows the reconstructed network using MB-CLM algorithm.

In our simulations we used plain MB algorithm and MB-CLM algorithm with a MB threshold value of 0.90 in both cases and tested the algorithms using synthetic network 5 x 4 nodes and corresponding synthetic data of upto 100 samples.



(a) Synthetic network (b) Logic circuit (c) Reconstructed Network

Fig. 4. Synthetic network and minimized constraint logic

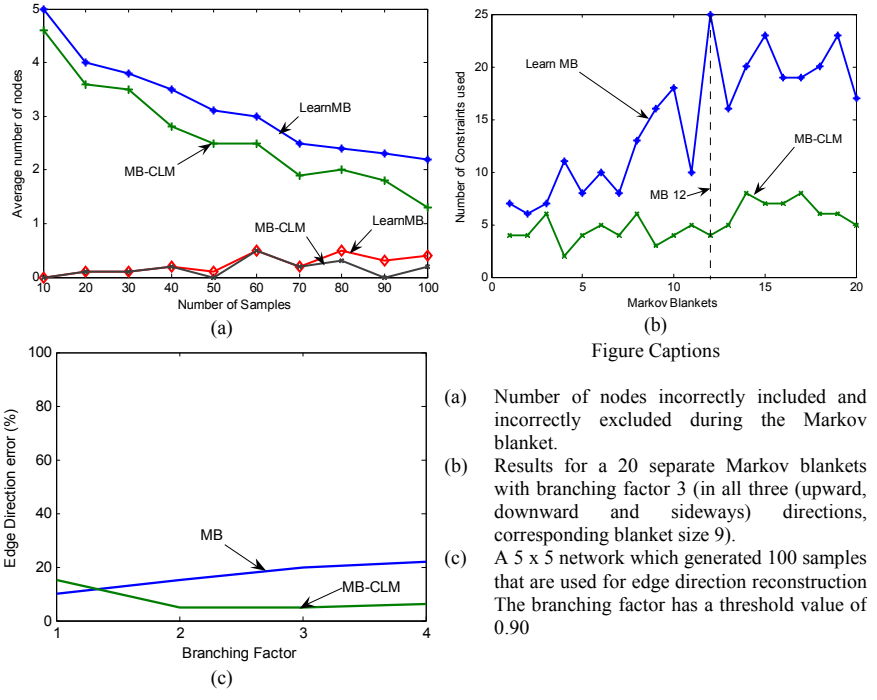


Figure Captions

(a) Number of nodes incorrectly included and incorrectly excluded during the Markov blanket.

(b) Results for a 20 separate Markov blankets with branching factor 3 (in all three (upward, downward and sideways) directions, corresponding blanket size 9).

(c) A 5 x 5 network which generated 100 samples that are used for edge direction reconstruction The branching factor has a threshold value of 0.90

Fig. 5. Simulation Results

Figure 5a shows a plot of the number of nodes of the MB incorrectly included or excluded for plain Learn_MB algorithm and MB-CLM algorithm, averaged over all nodes in the domain. It can be observed that due to the constraint minimization, the accuracy of results have increased, as a result the number of nodes incorrectly included is less for the MB-CLM algorithm compared to the Learn_MB algorithm. Hence, there is better accuracy and reliability with the MB-CLM algorithm. On the other hand, as can be seen from Fig.5a, the use of Learn_MB algorithm resulted in a slightly higher number of missing nodes. Although the nodes incorrectly included are very low for both Learn_MB and MB-CLM algorithm, the nodes incorrectly excluded fall more rapidly with increasing sample size in the case of MB-CLM algorithm compared to Learn_MB algorithm. From Fig. 5b, it can be observed that MB 12 has very high constraints which are minimized by MB-CLM algorithm. The CLM algorithm thus can help with large reduction of constraints in certain circumstances. The effect on percentage Direction Error (DE) by increasing MB (via branching factor increase) is shown in Fig. 5c. DE for the MB and the MB-CLM algorithm remains close for lower branching factors but decreases slightly for MB-CLM algorithm with increase in branching factor. The decrease is due to the large number of parents for each node (i.e. more V structures) which provides greater opportunities to recover the directionality of an edge with increased number of tests.

5.2 Real Datasets

For testing MB-CLM algorithm for inferring genetic regulatory interactions using real life data set, studies were also carried on the dataset from Spellman *et al.* (1998) [8] obtained for *S.cerevisiae* cell cultures that were synchronized by three different methods.

In our study, we considered a group of 20 important genes (CLB2, CLN1, SIC1, LB6, CLB1, SWI4, CDC34, SWI5, CDC20, CLN2, MCM1, CLB5, SWI6, LB4, CLN3, BP1, SKP1, CDC28, HCT1) to be involved in cell-cycle regulation of *S.cerevisiae*. The same set of genes has been used by Chen *et al.* (2000)[13], who presented a complete model of the cell-cycle events. We applied the MB-CLM to learn the models from the data, for each gene in the dataset, considering all other genes in the dataset as candidate regulators. The MB of genes is obtained by Gibbs variable selection procedure, and then the model evaluation is performed using the proposed algorithm. We investigated the effect of CLM algorithm on the correctness of reconstruction. For small models 1 and 2 (i.e. models with branching factor <=2), CLM algorithm did not make any significant impact. However, for the large model 4, which has a large condition set, the incorporation of CLM algorithm helped discover *new* regulatory relations for some genes which were undetected when only MB algorithm was used. The available biological knowledge also validated the existence of these new regulatory interactions learned from the model. Highly accurate *regulatory* interactions were also discovered for the seven genes CLN1, CLN2, CLB1, CLB2, CLB5, SWI5 and SWI4. These results are observed to be consistent with the available biological knowledge. These inferred genetic interactions as well as the *activatory* connections amongst the genes CLN1, CLN2, CLB5 and CLB6 can be seen in Fig.6.

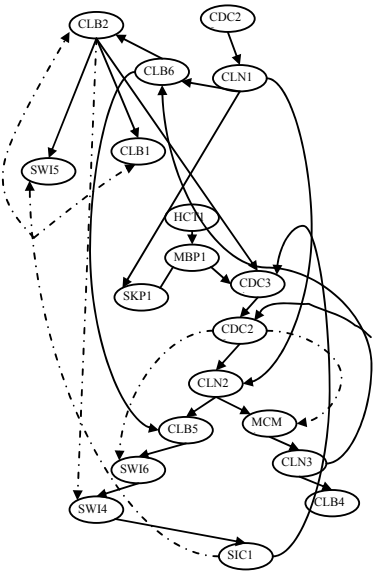


Fig. 6. GRN reconstruction

The *time delay* learning when CLM algorithm was used revealed the activatory influences CLN1→CLN2, CLB6→CLB5, CLN1→CLB6 and CLN3→CLB6 (CLN3 is also the G1-specific cyclin). These find support in the literature [14-19]. The *time delay* learning was also able to infer the inhibitory influence of SIC1 on the genes CLB1, CLB2 and SWI5 [18]. The *plus/ minus* learning using the CLM algorithm showed a positive relation between the genes SIC1, CDC20 and CDC34. The gene CDC20 is required for proteolytic degradation of G1 regulators which explains the negative connections discovered by CLM from gene CDC20 to SWI6 and MCM1, both of which are encoding transcription factors. The gene CDC20 is transcribed in the late S/G2 phase, whereas the genes CLN2 and CLB5 are expressed in G1 phase, supporting the negative connection between CDC20 and these genes.

6 Conclusion

In this paper, a Markov blanket (MB) based constraint minimization algorithm (MB-CLM algorithm) for efficiently learning the GRN is presented. The CLM algorithm initially uses the original Learn_MB algorithm to convert a putative MB structure into a set of constraints which are then tested against the given data. The MB-CLM heuristically minimizes this constraint set using K-map logic minimization technique to improve MB inference resulting in a superior GRN reconstruction. The performance of MB-CLM algorithm is investigated using both the synthetic data and real data (yeast). Experiments with synthetic data show that the number of nodes incorrectly included (or excluded) with only MB algorithm reduces significantly when CLM algorithm is incorporated. Simulations studies with yeast data discovered new regulatory relations when CLM algorithm was used. Using MB-CLM algorithm, both the *time delay learning* algorithm and the *plus/minus learning* algorithm revealed interactions which were not reconstructed with MB algorithm. These newly discovered relations were validated to be correct by the biological knowledge. Thus, in the MB-CLM algorithm improves the overall process of GRN reconstruction.

References

1. Sprites, P., Glymour, C., Scheines, R.: Causation, Prediction, and Search: Adaptive Computation and Machine Learning, 2nd edn. MIT Press, Cambridge (2000)
2. Pearl, J.: Causality: Models, Reasoning and Inference. Cambridge University Press, Cambridge (2000)
3. Verma., T.S., Pearl., J.: 'A theory of inferred causation', Principles of Knowledge Representation and Reasoning, pp. 441–452 (1991)
4. Cheng, J., Bell, D.A., Liu, W.: An algorithm for Bayesian belief network construction from data. In: Book An algorithm for Bayesian belief network construction from data, pp. 83–90 (1997)
5. Ram, R., Chetty, M., Dix, T.I.: Causal Modeling of Gene Regulatory Network. In: Book Causal Modeling of Gene Regulatory Network, pp. 1–8 (2006)
6. Ram, R., Chetty, M.: Learning Structure of Gene Regulatory Networks. In: Book Learning Structure of Gene Regulatory Networks, pp. 525–531 (2007)
7. Mano., M.M.: Digital Design, 3rd edn. Prentice Hall, Inc., Englewood Cliffs (2002)

8. Spellman, P.T., Sherlock, G., Zhang, M.Q., Iyer, V.R., Anders, K., Eisen, M.B., Brown, P.O., Botstein, D., Futcher, B.: Comprehensive identification of cell cycle-regulated genes of the yeast Saccharomyces cerevisiae by microarray hybridization. Mol. Biol. Cell. 9, 3273–3297 (1998)
9. Aliferis, C.F., Tsamardinos, I.: Algorithms for Large-Scale Local Causal Discovery in the Presence of Small Sample or Large Causal Neighborhoods. In: Book Algorithms for Large-Scale Local Causal Discovery in the Presence of Small Sample or Large Causal Neighborhoods, Vanderbilt University (2002)
10. Aliferis, C.F., Statnikov, I.T.A.: HITON, A Novel Markov Blanket Algorithm for Optimal Variable Selection. In: Book HITON, A Novel Markov Blanket Algorithm for Optimal Variable Selection, Vanderbilt University (2003)
11. Chickering, D.M.: Learning Equivalence Classes of Bayesian-Network Structures. Journal of Machine Learning Research 2, 445–498 (2002)
12. Ram, R., Chetty, M.: Framework for path analysis for learning Gene regulatory network. In: Book Framework for path analysis for learning Gene regulatory network, pp. 264–273. Springer, Heidelberg (2007)
13. Chen, K.C., Calzone, L., Csikasz-Nagy, A., Cross, F.R., Novak, B., Tyson, J.J.: Integrative analysis of cell cycle control in budding yeast. Mol. Biol. Cell. 15, 3841–3862 (2004)
14. Futcher, B.: Transcriptional Regulatory Networks and the Yeast Cell Cycle. Current Opinion in Cell Biology 14, 676–683 (2002)
15. Gardner, T.S., et al.: Inferring genetic networks and indentifying compoud mode of action via expression profiling. Science 301, 102–105 (2003)
16. Güldener, U., Münsterkötter, M., Kastenmüller, G., et al.: CYGD: the Comprehensive Yeast Genome Database. Nucleic Acids Research 33, D364–D368 (2005)
17. Shinohara, A., Iida, K., Takeda, M., Maruyama, O., Miyano, S., Kuhara, S.: Finding Sparse Gene Networks. Genome Inf. 11, 249–250 (2000)
18. Stucki, J.W.: Stability analysis of biochemical systems. A practical guide. Progress Biophys. Mol. Biol. 33, 99–187 (1978)

Fusion of Gene Regulatory and Protein Interaction Networks Using Skip-Chain Models

Iti Chaturvedi[1] and Jagath C. Rajapakse[1,2,3]

[1] Bioinformatics Research Center, School of Computer Engineering, Nanyang Technological University, Singapore
[2] Singapore-MIT Alliance, Singapore
[3] Department of Biological Engineering, Massachusetts Institute of Technology, USA

Abstract. Inference of Gene Regulatory Networks (GRN) is important in understanding signal transduction pathways. This involves predicting the correct sequence of interactions and identifying all interacting genes. Using only gene expression data is insufficient, so additional sources of data like protein-protein interaction network (PPIN) are required. In this paper, we model time delayed interactions using a skip-chain model which finds missing edges between non-consecutive time points based on PPIN. Highest Viterbi approximation is used to select skip-edges. The k-skip validation model checks for k missing genes between a predicted interaction, giving us advantages of validation as well as expansion of GRN. The method is demonstrated on a cell-division cycle data of *S.cerevisiae* (yeast). Comparison of the present method, with a previous approach of modeling PPIN by using a Gibbs prior are given.

Keywords: Dynamic Bayesian networks, Gene Regulatory networks, Higher-order Markov chains, Protein-Protein interactions, Viterbi algorithm.

1 Introduction

Most processes of signal transduction involve ordered sequences of biochemical reactions inside the cell, which are carried out by an ensemble of enzymes activated by secondary messengers, resulting in signal transduction pathways. The DNA in a cell contains genes which are converted to mRNA(expressed genes) through transcription and then translated into proteins. Consequently, signal transduction pathways are often interpreted in terms of gene regulatory networks (GRN) and protein-protein interaction networks (PPIN). High throughput techniques allow generation of both gene and protein interaction data simultaneously. Studies that use both gene and protein expressions have been mostly devoted to a single type of data while the other type of data is restricted to validation [1], [2], [3]. Using a single data source of interactions has its own limitations and could create errors in the analysis of the interactome. This is due to two main reasons: firstly, both microarray and PPI data have a lot of noise due to measurement errors, varied transcriptional response in the cell and inter-functional phenomena. Secondly, complex formation and other critical interactions that regulate biological processes take place

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 214–224, 2008.

at the protein level. A protein being a product of a gene, a joint mining of gene regulatory networks and protein interaction networks could reveal genes that are co-expressed and their proteins also interact. Clusters and interactions found by joint mining will be more reliable than those found by using only one type of data. We may thereby have high confidence of finding gene clusters that are regulated by the same mechanism and belong to the same biological process.

Fusion of GRN and PPIN has been attempted by using Naive Bayes where genes are partitioned into different pathways [1]. Likelihood of the data becomes higher when genes in the same pathway have similar expression profiles and are interacting. The unified model is learned using Expectation Maximization algorithm. This method however requires the user to specify the number of pathways which is often unknown. Cross-graph mining [2] can achieve integration by looking for partial cliques in both GRN and PPIN simultaneously. A weighted score of SVM classification of gene expression and functional classification of PPI was suggested in [3]. The weights are determined by simulated annealing. Prior modeling of PPIN into Bayesian learning has been done using Gibbs distribution [4]. A Gibbs random field equivalent to a first-order Markov random field is used to represent the prior graph. This approach is insufficient because many time-delayed interactions are known to exist.

In this paper, we extend skip-chain sequence models [5] and use Viterbi approximation of dynamic Bayesian networks (DBN) to include time delayed interaction edges based on PPIN data. The method is demonstrated by using yeast cell cycle data, where genes are differentially expressed in each phase. Genes in one phase regulate by activation/inhibition genes in the next phase resulting in a cycle. A comparison is done with prior modeling of PPIN using Gibbs distribution. The method almost doubles the sensitivity and is robust to the increase in number of genes. The paper is organized as follows. Section 2 discusses Bayesian networks (BN), dynamic Bayesian networks(DBN) and their extension to higher-order. Section 3 explains the different models: we first discuss prediction of GRN using skip-chain models and our extension to fuse prior knowledge of PPIN. Next we describe the k-skip validation model of GRN based on PPIN. In section 4, we demonstrate our approach on 5 different datasets from yeast cell cycle. A comparison is done with prior modeling of PPIN using Gibbs distribution. Lastly, we make our conclusions in section 5.

2 Modeling GRN with Higher-Order Bayesian Networks

Microarray experiments simultaneously measure expression patterns of thousands of genes over different experimental conditions or over time. Let us consider a set of n such genes $G = \{g_1, g_2, ..., g_n\}$, and time-series gene expression data of length m for all the genes. Let the microarray data matrix $X = [x_1, x_2...x_n]^T$ in which row vector $x_i = (x_{i,1}, x_{i,2}, ..., x_{i,m})$ corresponds to gene expression time series of gene g_i where $x_{i,t}$ denotes the expression level at time t. Let the set of parents (or genes regulating) of gene i be denoted as a_i and the number of states the nodes in a_i take be q_i.

2.1 Bayesian Networks (BN)

A BN is a graphical model that can represent a joint multivariate probability distribution by capturing the properties of conditional independencies among the variables. It is a directed acyclic graph (DAG) having a structure S and a set of conditional distributions. BN can appropriately model the genes as nodes in the network and edge as causal interaction's between them.

The Bayesian network decomposes the joint probability of genes into the product of conditional probabilities by using the chain rule and independence of non-descendant genes, given their parents

$$p(x_1, x_2, ..., x_n) = \prod_{i=1}^{n} p(x_i|a_i, \theta_i) \tag{1}$$

where $p(x_i|a_i, \theta_i)$ is the conditional probability of gene expression x_i given its parents, and θ_i denotes the parameters of the conditional probabilities.

The optimal structure is obtained by maximizing the posterior probability for S. From Bayes theorem,

$$\max_S p(S|X) = \max_S p(S)p(X|S) \tag{2}$$

where $p(S)$ is the prior probability of the network structure. Given the set of conditional distributions with parameters $\theta = \{\theta_i | i = 1, 2, \ldots n\}$, the likelihood can be written as

$$p(X|S, \theta) = \int p(X|S, \theta)p(\theta|S)d\theta \tag{3}$$

Let us assume that gene expressions carry discrete levels of gene expression : $x_{i,t} = k$ where $k \in \{1, 2, ..., d\}$ and d denotes the maximum level of expression of any gene. Let $\theta_{ijk} = p(x_{i,t} = k|a_i = j)$ and N_{ijk} be the number of instances of θ_{ijk} that occur in the training data. Using the property of decomposability,

$$p(X|S, \theta) = \prod_{i=1}^{n} \prod_{j=1}^{q_i} \prod_{k=1}^{d} \theta_{ijk}^{N_{ijk}} \tag{4}$$

Assuming global and local parameter independence,

$$p(\theta|S) = \prod_{i=1}^{n} p(\theta_i|S) = \prod_{i=1}^{n} \prod_{j=1}^{q_i} p(\theta_{ij}|S) = \prod_{i=1}^{n} \prod_{j=1}^{q_i} \prod_{k=1}^{d} p(\theta_{ijk}) \tag{5}$$

Substituting Eq. (4) and Eq. (5) into Eq. (3) gives

$$p(X|S) = \prod_{i=1}^{n} \prod_{j=1}^{q_i} \int \prod_{k=1}^{d} \theta_{ijk}^{N_{ijk}} p(\theta_{ijk}) d\theta_{ijk} \tag{6}$$

We can approximate the integral by using maximum likelihood estimate known as Bayesian Information criterion (BIC)[6]. We can estimate θ_{ijk} as

$$\theta_{ijk} = \frac{N_{ijk}}{\sum_{k=1}^{d_i} N_{ijk}} \tag{7}$$

Taking the log-likelihood, gives us the following expression:

$$BIC = \log p(X|S,\theta) = \sum_{i=1}^{n}\sum_{j=1}^{q_i}\sum_{k=1}^{d} N_{ijk} \log \frac{N_{ijk}}{\sum_{k=1}^{d_i} N_{ijk}} \tag{8}$$

Hence the likelihood approximation of the score needs no prior over parameters as the posterior probability captures information of the prior. The likelihood approximation is known to be good when using large amounts of data. However, if the dataset is small it will over-penalize.

The acyclic condition in BN does not allow self regulation and feedback, which are characteristic of GRN. To overcome this, dynamic Bayesian networks (DBN) are used in which a transition network from one time point to the next characterizes the GRN.

2.2 Dynamic Bayesian Networks (DBN)

A first-order dynamic Bayesian network (DBN) is defined by a pair of structures (S_t, S_{t+1}) corresponding to time instances t and $t+1$ and a transition network of interactions between the two networks. The DBN structure is obtained by unrolling the transition network over time (Figure 1). In slice t, the parents of $x_{i,t}$ are those specified in the initial network S_t, and in slice $t+1$, the parents of $x_{i,t}$ are those genes in slice t corresponding to parents of $x_{i,t}$ in S_t. The transition network of interactions between time instances t and $t+1$ is given by

$$p(x_{i,1}, ..., x_{i,m}) = \prod_{t=1}^{m} p(x_{i,t}|x_{i,t-1}, \theta_i) \tag{9}$$

where $t = 0$ corresponds to the dummy initial state.

The metric for a DBN can hence be defined as

$$p(X|S,\theta) = \prod_{t=1}^{m}\prod_{i=1}^{n}\prod_{j=1}^{q_i}\prod_{k=1}^{d} \theta_{ijk}^{N_{ijk}^{(t,t+1)}} \quad \forall t \tag{10}$$

where $N_{ijk}^{(t,t+1)}$ correspond to the transition network (S_t, S_{t+1}). The first-order DBN has $2n$ nodes.

The first-order Markov DBN can be extended to a higher-order to allow higher-order interactions among variables. For an r-order Markov DBN, given a node x_i, its parents are chosen from the set of variables $X[t-r] \bigcup X[t-r-1]....X[t-1]$, where $X[t]$ is a column of gene expression matrix at time t. We assume r-order stationary Markov chain. With this assumption an r-order transition network has $(r+1) \times n$ nodes (Figure 1), where n is number of genes.

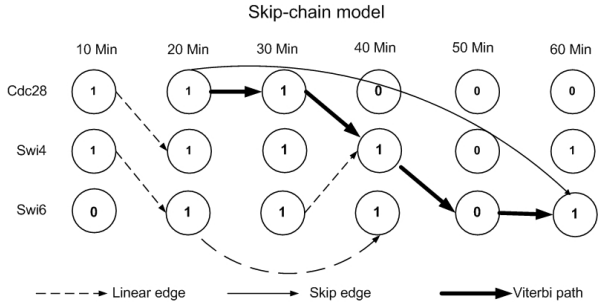


Fig. 1. Illustration of six time points and three genes of G1-S phase of cell-cycle in a dynamic Bayesian network. The dashed edges are linear $r = 1, 2$ order edges found by linear features. The solid directed edge is an example of skip-edge over four time points which models a long-distant dependency. The bold directed line shows a skip path computed by Viterbi algorithm.

Now N_{ijk} can include cases from any of the previous r time slices. As r increases the search space becomes extremely large. In order to address this problem we propose skip-chain sequence models.

3 Fusion of GRN and PPIN

3.1 Skip-Chain Sequence Model

A higher-order DBN is unable to accomodate long-distance dependencies because the number of parameters increases with the order. For example, if order is r, for binary gene expressions with a maximum q parents, there will be r^q parameters for each gene. To overcome the explosion of parameters, a skip-chain sequence model [7] augments a linear chain with skip-features that represent long range dependencies. It then simply factorizes the prediction probability into linear and skip features. The number of skip-features can be implemented based on prior knowledge as given in PPIN.

Linear-chain feature functions $f_u(x_i, a_{i(t-r:t)}, t)$ represent local dependencies that are consistent with an r-order Markov assumption of gene expressions. These represent dependencies between nearby time points and cannot represent higher-order dependencies like activation, inhibition or feedback, which occur throughout the time-series of the pathway. We relax the above assumption by using skip-chain feature functions $g_{u'}(x_i, a_i, s_t, t)$ which exploit dependencies between genes that are arbitrarily distant at time instances s_t and t respectively. Such a skip-feature models variable length Markov chain upto $m-1$ order where m is number of time points. The score of the structure is a weighted sum of linear and skip-edge scores:

$$\log p(x_i|a_i) \propto \sum_{u=1}^{U} \lambda_u f_u(x_i, a_{i(t-r:t)}, t) + \sum_{u'=1}^{U'} \mu_{u'} g_{u'}(x_i, a_i, s_t, t) \qquad (11)$$

where λ_u and $\mu_{u'}$ are weights for corresponding features.

Inspired by a previous application of skip-chain models to summarise group meetings [5], we achieve fusion by designing an interaction skip feature (g_i, g_j, s_t, t) between genes g_i and g_j, having similar expressions at two different time points s_t and t where $s_t \in \{1, 2, 3, .., t-2\}$. We further constrain that these genes must be interacting as indicated by the PPIN. In the next section, we discuss how a Viterbi approximation is used to determine the optimal skip features.

3.2 Skip-Edge Determination

In the skip-chain models, higher-order interactions are represented by skip-edges. We used Viterbi approximation of a DBN [8] to determine the optimal skip-features, as the maximum a posterior forward path through the trellis. This lets us compare different time-delayed skip-features between two genes. An example of such a path to model a fourth order interaction is shown in Figure 1. The Viterbi algorithm first calculates log transition probabilities from the data. The transition probability from gene $g_j, t-1$ to g_i, t at is defined as

$$p(x_{i,t}|x_{j,t-1}) = \frac{n_{i,j,t}}{\sum_{j'=1}^{n} n_{i,j',t}} \tag{12}$$

where $n_{i,j,t}$ denotes number of occurrences where $x_{i,t} = x_{j,t-1} = 1$ in the discretized gene expression data.

Then, forward state transitions l^t that give minimum transition probability at each time point is chosen:

$$l^t = \arg\min_j p(x_{i,t}|x_{j,t-1}) \tag{13}$$

And the skip-edge score is the negative of the total transition probabilities at the last time point:

$$g(x_i, a_i, s_t, t) = -\sum_{t'=s_t}^{t} \log l^t \tag{14}$$

where the parent set a_i has only one gene at time point s_t. As there can be many time-delayed skip features between two genes, we choose the time delay which has the highest normalized a posterior probability. Normalization is done by dividing the total probability by length of skip-edge.

The linear feature model is obtained from the Bayesian network:

$$f(x_i, a_{i(t-r:t)}, t) = \log p(x_i|a_i, \theta_i) \tag{15}$$

where a_i is parent set from any of previous r time points. The score of the structure now becomes

$$\log p(x_i|a_i) \propto f(x_i, a_{i(t-r:t)}, t) + \mu g(x_i, a_i, s_t, t) \tag{16}$$

Since we have a single parameter μ, it can be simply determined by repeated trial.

In Figure 1 the edges of three genes in the first six time points can be viewed. The dashed edges are linear $r = 1, 2$ order edges. The solid directed edge is an example of skip-edge over three time points which models a long-distant dependency.

3.3 *k*-Skip Validation

Validation of GRN is often done with the use of experimentally verified interactions listed in corresponding PPIN. It is presumed that probabilistic influence flows in the graph and genes g_i and g_j maybe connected if there is an unblocked path $g_i \rightarrow g_1 \rightarrow g_2 \rightarrow .... \rightarrow g_k \rightarrow g_j$ in the PPIN. Since PPI data available is undirectional, the true structure of S is predicted by using a k-skip validation model [9]. The k-skip validation model looks for one or more genes that are skipped when predicting an interaction.

Consider two genes, g_i and g_j, interacting in the GRN, The k-skip validation model checks for a cascade of k genes $(g_1, g_2, ..., g_k)$ such that any gene g_c in the cascade interacts with the next gene g_{c+1} where $c \in \{1, 2, .., k-1\}$. Lastly g_i must interact with g_1 and g_j to interact with g_k. Some examples are shown in Figure 2: Interaction Cln3-Cdc28 is predicted using microarray data and is also found in PPIN; Mbp1-Cln3 is not found in PPIN, however a 1-skip form Mbp1-Swi4-Cln3 exists in PPIN; Similarly a 2-skip form of Exg2-Htb2 will be Exg2-Hsp82-Spt15-Htb2.

A DBN chooses parent-child relations with the highest likelihood based on the time-series microarray data and assumes a first-order Markov chain. However, many time-delayed interactions are known to exist and this causes skipping of genes. The k-skip validation could expand a predicted GRN as well as correct using information from PPIN. It has been previously reported that 1-skip and 2-skip forms are common in predicted GRN.

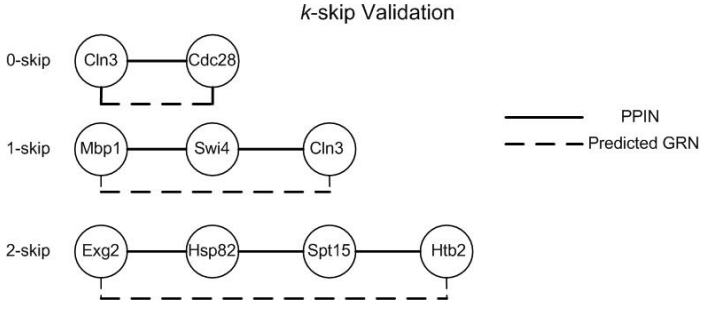


Fig. 2. k-skip validation model looks for one or more genes in the PPIN that were skipped while predicting gene interaction. Dashed lines are predicted false positives of GRN and solid lines are PPIN.

4 Results and Discussion

We evaluated our method on a dataset acquired in cell-cycle regulation of yeast [10]. The microarray dataset was taken from Spellman et al. [10] which has mRNA measurements of 6,178 genes under different experimental settings. The expression values range from -2 to +2, which were normalized and discretized into levels 0(under-expression) and 1(over-expression), respectively. We chose 24 time points of cdc-15 cell cycle arrest ranging from 10 to 290 min. Yeast cell-division cycle consists of four main phases: genome duplication (S phase), and nuclear division (M phase), separated by two gap phases (G1 and G2). The S-G1-M-G2-S form a cycle for cell duplication. Phase specific gene expression profiles were extracted based on the list given in [10]. The PPIN dataset for validation was downloaded from BioGRID (Biological General Repository for Interaction Datasets) [11]. It has over $50,000$ experimental as well as literature-derived interactions for the genome. Phase specific prior PPIN for the pathway was derived from the validation set. Using phase specific PPIN network and gene expression profiles, we looked for skip-edges.

For expression data discretized into two levels 0(under-expressed) and 1(over-expressed), we consider 7 time points (9 to 16) of peak activity for 118 genes in G1 phase of cell-cycle. Table 1 shows that a skip feature captures the correlation among non-consecutive time points. A chi-square test shows that consecutive time points t and $t-1$ were not significantly correlated with a p-value 0.77, while noncosecutive time points s_t and t where $s_t \in \{1, 2, 3, ..., t-2\}$ are correlated with a p-value less than 0.001. It can be seen that there are 9146 skip-edges.

A genetic algorithm (GA) was implemented to find the optimal structure of the instantaneous network of GRN [12]. A solution individual $C = \{c_{ij}\}_{n \times n}$ where $c_{ij} \in \{0, 1, ..., r\}$ denotes the strength of the interaction between genes i and j, and 0 means no interaction. We used highest time-delay as the order of an interaction in PPIN and a random order for unknown interactions. Each individual in the GA allowed upto three parents for a gene. The GA chooses the network with best combination of skip and linear edges (Eq. 16). Simulation was done at different numbers of individuals (N) and generations (G) (N=200/300/400 and

Table 1. Contingency tables for 118 genes in G1 phase, at peak activity (time points 9 to 16) for yeast cell cycle data. Chi-square test shows that the adjacent expressions $x_{i,t}$ and $x_{j,t-1}$ are not correlated with p-value = 0.77, however expressions at far away time points $x_{i,t}$ and x_{j,s_t} where $s_t \in \{1, 2, 3, ..., t-2\}$ clearly influence each other with a p-value ≤ 0.001.

	$x_{i,t} = 0$	$x_{i,t} = 1$	Chi-square	P-value
Linear edge				
$x_{j,t-1} = 1$	518	1954	0.08	$p = 0.77$
$x_{j,t-1} = 0$	152	554		
Skip edge			96.46	$p \leq 0.001$
$x_{j,s_t} = 1$	2848	9146		
$x_{j,s_t} = 0$	2512	10918		

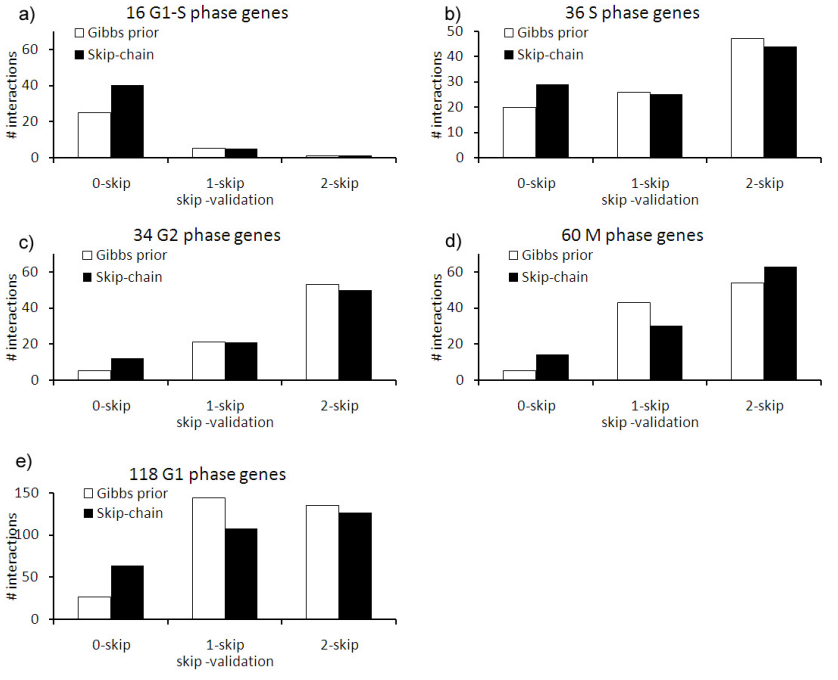


Fig. 3. k-skip validation to compare Gibbs prior and skip-chain models on five datasets in yeast cell-cycle: (a) 16 genes from G1-S phase (b) 36 genes from S phase (c) 34 genes from G2 phase (d) 60 genes from M phase and (e) 118 genes from G1 phase. Number of correct predictions by skip-chain is almost twice those predicted by Gibbs prior.

G=300/400/500). The GA stops when the maximum number of generations is reached or if the score does not change for 20 consecutive generations. The best prediction among all five runs was considered.

Previously, PPIN have been fused with GRN using Markov random field as the prior network [4]. The interaction potential for a PPIN validated interaction is defined by the user. The entropy $P(S)$ then becomes a sum of interaction potentials for a predicted GRN. Appropriate weights of the PPI edge in Gibbs random field and skip-chain DBN which gave best prediction were got by repeated trial. The appropriate weight for 16 genes with 4678 skip-edges was found to be 40, as further increase in weight did not improve prediction. We approximately used 80 for 36 genes, 160 for 60 genes and so on.

The above procedure was carried out for five sets of genes (i) 16 genes in G1-S phase (ii) 36 genes in S phase (iii) 34 genes in G2 phase (iv) 60 genes in M phase and (v) 118 genes in G1 phase (Figure 3). For all five datasets our method outperformed the previous approach of fusing PPIN as a Gibbs prior. The sensitivity approximately doubles in all cases while the specificity remains high (Table 2). We conclude that the method is robust to the increase in the number of genes. Unlike Gibbs prior, the skip-chain model consider's

Table 2. Comparison of fusion of GRN and PPIN for Yeast cell cycle data by using a Gibbs prior and skip-chain model

	Gibbs prior		Skip-chain model	
Number of Genes	Sensitivity	Specificity	Sensitivity	Specificity
16 G1-S	0.35	0.73	0.65	0.93
36 S	0.59	0.86	0.74	0.88
34 G2	0.34	0.88	0.99	0.99
60 M	0.1	0.9	0.3	0.9
118 G1	0.1	0.95	0.21	0.96

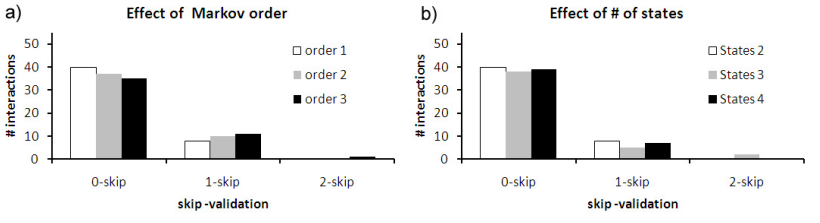


Fig. 4. k-skip validation of a predicted GRN for 16 G1-S phase genes of Yeast cell cycle (a) Increasing the order of a linear edge decreases the number of correct predictions. (b) Increasing the number of discrete states does not show any significant change in prediction performance.

the time delayed interaction between two genes. It is also able to distinguish between different interactions while Gibbs prior assigns equal weights to all PPIN interactions. We only found 1-skip or 2-skip cascades and no 3-skip cascades were found in all the datasets. The number of 2-skip was also higher in the larger networks (see Figure 3).

A few interactions were seen outside the prior network, suggesting missing members of a pathway. Increasing the Markov order of the DBN causes overfitting, this could be because of redundant effects of skip-chain and the higher-order DBN(Figure 4a). Here we have considered binary states $\{0, 1\}$ to represent under expression and over expression of a gene. Increasing the number of states did not give a significant increase in performance (Figure 4b). It might however be useful when applied to other datasets with higher noise.

5 Conclusion

Higher-order dependencies are significant for time-series gene expression data analysis when deriving gene regulatory networks. We propose a method for effective fusion of GRN and PPIN by introducing skip-edges found on PPIN into GRN predicted by first-order DBN modeling. This almost doubles the sensitivity of GRN, compared to the earlier modeling using Gibbs distribution while the number of false positives remains same.

Feature functions are a good way to include prior knowledge into a gene-regulatory network. The method is found to be robust when applied to larger networks. The approach is computationally efficient. Here, we have only considered interactions where both genes have similar expression at different time points. Similar modeling can be done for activation, inhibition and feedback events of causal networks of genes.

References

1. Segal, E., Wang, H., Koller, D.: Discovering molecular pathways from protein interaction and gene expression data. Bioinformatics 19(suppl. 1), i264-i272 (2003)
2. Pei, J., Jiang, D., Zhang, A.: Mining Cross-graph Quasi-cliques in Gene Expression and Protein Interaction Data. In: Proceedings of the 21st International Conference on Data Engineering (ICDE 2005), vol. 5(8), pp. 353–356 (2005)
3. Tu, K., Yu, H., Li, Y.-X.: Combining gene expression profiles and protein-protein interaction data to infer gene functions. Journal of Biotechnology 124, 475–485 (2006)
4. Nariai, N., Kim, S., Imoto, S., Miyano, S.: Using protein-protein interactions for refining gene networks estimated from microarray data by Bayesian networks. PSB. 9, 336–347 (2004)
5. Galley., M.: A Skip-Chain Conditional Random Field for Ranking Meeting Utterances by Importance. In: Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP 2006), pp. 364–372 (2006)
6. Friedman, N., Murphy, K., Russell, S.: Learning the structure of dynamic probabilistic networks. In: Fourteenth Conf. on Uncertainty in Artificial Intelligence (UAI), pp. 139–147 (1998)
7. Sutton, C., McCallum, A.: Collective segmentation and labeling of distant entities in information extraction. In: Presented at ICML Workshop on Statistical Relational Learning and Its Connections to Other Fields, University of Massachusetts (2004)
8. Tang, H., Huang, T.S.: Improved graphical model for audiovisual object tracking. In: IEEE International conference on Multimedia and Expo. pp. 997–1000 (2006)
9. Chaturvedi, I., Sakharkar, M.K., Rajapakse, J.C.: Validation of Gene Regulatory Networks from Protein-Protein Interaction Data: Application to Cell-cycle Regulation. In: Rajapakse, J.C., Schmidt, B., Volkert, L.G. (eds.) PRIB 2007. LNCS (LNBI), vol. 4774, pp. 300–310. Springer, Heidelberg (2007)
10. Spellman, P.T., Sherlock, G., Zhang, M.Q., et al.: Comprehensive Identification of Cell Cycle-regulated Genes of the Yeast Saccharomyces cerevisiae by Microarray Hybridization. Molecular Biology of the Cell 9, 3273–3297 (1998)
11. Stark, C., Breitkreutz, B.J., Reguly, T., et al.: BioGRID: a general repository for interaction datasets. Nucleic Acids Res. 34(Database issue), D535–539 (2006)
12. Xing, Z., Wu, D.: Modeling Multiple Time Units Delayed Gene Regulatory Network Using Dynamic Bayesian Network. In: Sixth IEEE International Conf. on Data Mining (ICDMW 2006), pp. 190–195 (2006)

TopEVM: Using Co-occurrence and Topology Patterns of Enzymes in Metabolic Networks to Construct Phylogenetic Trees

Tingting Zhou[1,2], Keith C.C. Chan[2], and Zhenghua Wang[1]

[1] National Laboratory for Paralleling and Distributed Processing,
National University of Defense Technology, Changsha, Hunan, 410073, P.R. China
Tingting.Zhou@live.com, zhhwang188@sina.com
[2] Department of computing, The Hong Kong Polytechnic University, Hong Kong, China
cskcchan@inet.polyu.edu.hk

Abstract. Network-based phylogenetic analysis typically involves representing metabolic networks as graphs and analyzing the characteristics of vertex sets using set theoretic measures. Such approaches, however, fail to take into account the structural characteristics of graphs. In this paper we propose a new pattern recognition technique, *TopEVM*, to help representing metabolic networks as weighted vectors. We assign weights according to co-occurrence patterns and topology patterns of enzymes, where the former are determined in a manner similar to the *Tf-Idf* approach used in document clustering, and the latter are determined using the degree centrality of enzymes. By comparing the weighted vectors of organisms, we determine the evolutionary distances and construct the phylogenetic trees. The resulting *TopEVM* trees are compared to the previous *NCE* trees with the NCBI Taxonomy trees as reference. It shows that *TopEVM* can construct trees much closer to the NCBI Taxonomy trees than the previous *NCE* methods.

Keywords: *TopEVM*, phylogenetic analysis, metabolic network, co-occurrence pattern, document clustering, topology pattern, degree centrality, evolutionary distance.

1 Introduction

The objective of phylogenetic analysis is to reconstruct the evolutionary relationship among different species and to display them in a tree-structured model called a *phylogenetic tree* [1]. Applications include the design of new drugs and the reconstruction of the history of infectious diseases [2]. Most previous research [3] in this area has been based on sequence alignment but these sequence-based approaches are easily influenced by horizontal gene transfer (HGT) [4, 5]. An alternative to this is network-based phylogenetics analysis, which compares the homogeneous biological networks of organisms. They often make use of metabolic networks and take the quantified difference between these networks as the evolutionary distance.

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 225–236, 2008.

A metabolic network is a hierarchical, graph-represented abstract of an actual metabolism. Composed of thousands of metabolites, enzymes, reactions and the relationships among them, global metabolic networks are too large and complicated to be compared element by element. So, for comparison purposes, the vertex sets of graphs, rather than the entire graph, is common to use. In such cases, the evolutionary distances are determined by applying set theoretic measures [6-10]. For example, Aguilar et al [11] treat the organisms as enzyme sets from the view of metabolism, building a binary vector for each organism according to the presence or absence of the enzymatic functions. Using the *NCE* (Number of Common Enzymes) method and a normalized *Hamming distance*, they construct phylogenetic trees by creating a distance matrix for each metabolic class. Forst et al [8] construct 'clean' metabolite-reaction bipartite graphs to represent metabolic networks. Using the *Jaccard distance* as the evolutionary distance measure, they construct the distance matrix by taking organisms as reaction sets. Tohsato [7] consider metabolic networks as enzymatic reaction sets. Also using the *Jaccard distance*, she determines the evolutionary distance matrix and constructed phylogenetic trees. One drawback of such set-theoretic methods is that they do not usually take into account the edge information, and therefore they do not have enough topological characteristics for the network comparison, especially the topological importance of vertices [9, 12, 13].

In this paper we propose a new pattern recognition technique, *TopEVM*, for use in phylogenetic analysis. *TopEVM* avoids a common drawback of set-theoretic methods in that it takes account of the structural characteristics of graphs by representing metabolic networks as weighted vectors. We assign the weights based on the co-occurrence and topology patterns of enzymes in organisms, where the co-occurrence patterns are determined using a method similar to the *Tf-Idf* approach in the document clustering and the enzyme topology patterns are determined according to the degree of centrality of enzymes. By comparing the weighted vectors of organisms, we determine the evolutionary distance matrices for the construction of phylogenetic trees. Comparing to the previous set-theoretic methods, *TopEVM* can produce phylogenetic trees closer to the taxonomy trees of NCBI.

The remainder of this paper is organized as follows. Section 2 elaborates the *TopEVM* approach. Section 3 describes our experiments and results. Section 4 provides conclusion and outlines directions for the future work.

2 *TopEVM*: Constructing Phylogenetic Trees by Using Enzyme Co-occurrence and Topology Patterns

In this section we describe the operation of the *TopEVM* approach, which proposes the use of a frequency weighting scheme and a topological vector. This approach proceeds from the observation that it is possible to regard the construction of species trees in phylogeny as similar to the process of distance-based clustering of organisms which may in turn be seen as analogous to document clustering, with an organism as a document and an enzyme as a term. This allows us to apply feature extraction approaches and the hierarchical clustering methods to the construction of phylogenetic trees.

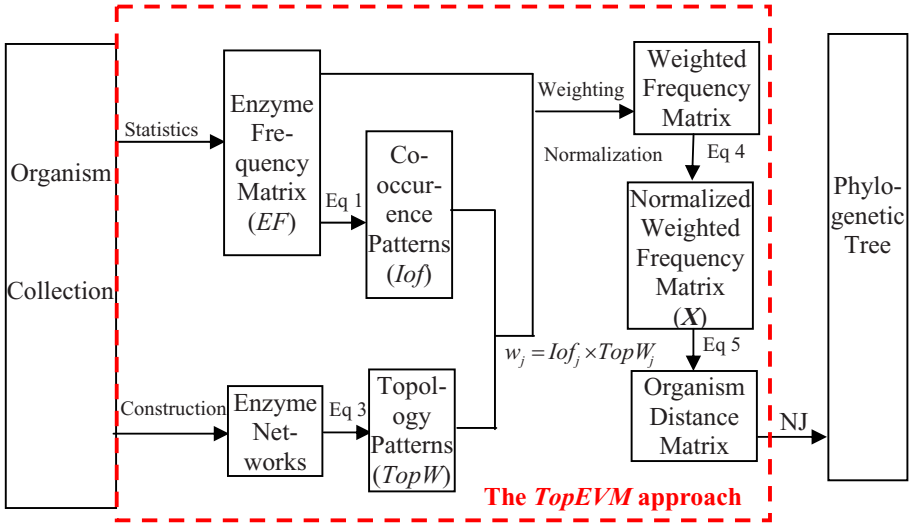


Fig. 1. The *TopEVM* approach

The first step in the *TopEVM* approach is to set up a matrix to record how frequently an enzyme occurs in a given collection of organisms, and extract enzyme co-occurrence patterns according to this matrix. By analogy with the *Tf-Idf* weighting scheme used in document clustering, we define *Inverse organism frequency* (*Iof*), a weight vector, to extract enzyme co-occurrence patterns. In the second step *TopEVM* uses a topological weight vector, *TopW*, to extract topologically important enzymes, the enzyme topology patterns, for use as features. This is done by representing metabolic networks as enzyme graphs and then counting the degree centrality, one measure of topological importance, of enzymes. The next step is to normalize the enzyme co-occurrence and topology weighting schemes. These are then used to convert the original frequency matrix into a new matrix in which rows denote the final Topology-weighted Enzyme Vector Model (*TopEVM*) of organisms. Finally a distance matrix is established by comparing the *TopEVM* of organisms with *Soergel Distance* as the distance measure. The distance matrix is used to construct the phylogenetic trees by use of some distance-based clustering approach, e.g., Neighbor Joining (NJ) method. Figure 1 shows the flow of the entire procedure.

2.1 Inverse Organism Frequency: Extracting Enzyme Co-occurrence Patterns

The first step to extract the co-occurrence patterns of enzymes is to set up a matrix to record how frequently an enzyme occurs in a given collection of organisms. For this purpose, we define *Enzyme Vector Space* to denote the organism-enzyme frequency matrix and *Enzyme Vector Model* to denote the organisms as enzyme frequency vectors. It should be noted that we make two assumptions in the definitions. First, we assume that

enzymes are arranged in the ascending order of their EC numbers[1] and the order is kept constant. Second, we assume that organisms in the collection are arranged in an arbitrary but constant order.

Enzyme Vector Space. Let O be a collection of m organisms o_i, $i = 1..m$. Let E be all the n enzymes e_j, $j = 1..n$, of at least one organism in O. The *Enzyme Vector Space* O^{mn} is defined through the organism-enzyme frequency matrix, EF, where ef_{ij} is the frequency of the j^{th} enzyme, $j = 1..n$, in the i^{th} organism.

Enzyme Vector Model. Given an enzyme vector space O^{mn} and its organism-enzyme frequency matrix, EF, the enzyme frequency vector for the i^{th} organism o_i is defined as the i^{th} row of EF. We call this representation of organisms as *Enzyme Vector Model*.

Document clustering generally assumes that the total term frequency is not always indicative of a term's information content. To account for this disparity, the *Inverse document frequency* (*Idf*) weighting scheme is often applied [14]. We find a similar situation when we compare the enzyme frequency vectors of organisms. That is to say, the frequency of an enzyme appeared in all the organisms cannot be assumed to indicate its information content. To deal with this, we apply a weighting scheme in this study, which is similar to the *Idf* weighting scheme. We call this the *Inverse organism frequency* (*Iof*) weighting scheme and define it as follows.

Organism Frequency. Given an enzyme vector space O^{mn} and its organism-enzyme frequency matrix EF, the *Organism Frequency* (of_j) of a given enzyme e_j, $j = 1..n$, is defined as the number of organisms that contain the enzyme e_j.

Inverse Organism Frequency. Given a enzyme vector space O^{mn} and its organism-enzyme frequency matrix EF, the *Inverse organism frequency* (Iof_j) of a given enzyme e_j, $j = 1..n$, is defined as the logarithm of the quotient of dividing the total organism number by its organism frequency (of_j). That is,

$$Iof_j = \log\left(m \Big/ \sum_i I_{(ef_{ij}>0)} \right) \tag{1}$$

where $I_{(\cdot)}$ denotes the indicator function, $I_{(cond)} = \begin{cases} 1 & if.cond.is.fulfilled \\ 0 & otherwise \end{cases}$, and $\sum_i I_{(ef_{ij}>0)}$ is the organism frequency of e_j, namely of_j.

Once Iof_j has been assigned to enzyme e_j, the original frequency of enzyme e_j in organism o_i, namely ef_{ij}, can be transformed into a new weighted frequency ef_{ij}',

$$ef_{ij}' = Iof_j \cdot ef_{ij} \tag{2}$$

Since the *Iof* weighting scheme gives lower weights to the enzymes found in a large number of organisms and higher weights to those found in fewer organisms, the *Iof* weights emphasize organism-specific enzymes.

[1] The EC (Enzyme Commission) number is a numerical classification scheme for enzymes.

2.2 Topology Weight: Extracting Enzyme Topology Patterns

Some enzymes in a metabolic network will have a higher average connectivity than others [15]. On the assumption that this higher average connectivity represents topological important information, we define *Topology Vector Space* to denote the organism-enzyme topology matrix and *Topology weight* (*TopW*) weighting scheme to select more strongly connected enzymes.

Topology Vector Space. Let $\boldsymbol{R}$ be a collection of m metabolic networks. r_i is constructed for the i^{th} organism o_i, $i = 1..m$. Let $\boldsymbol{E}$ be the collection of all the n enzymes e_j, $j = 1..n$, which are contained by at least one metabolic network in $\boldsymbol{R}$. The Topology Vector Space $\boldsymbol{R}^{mn}$ is defined through the organism-enzyme topology matrix, $\boldsymbol{T}$, where t_{ij} is the topological importance of the j^{th} enzyme in the metabolic network of the i^{th} organism, $j = 1..n$.

In this study, the degree centrality, the number of direct neighbors of a node [16], is regarded as the measure of the node's topological importance. In order to distinguish the absent enzymes from the present enzymes with degree as '0', we assign the degree centrality of the absent enzymes as '-1' in the topology matrix $\boldsymbol{T}$.

Topology Weight. Given a topology vector space R^{mn} and its organism-enzyme topology matrix T, the *Topology weight* ($TopW_j$) of the given enzyme e_j, $j = 1..n$, is defined as:

$$TopW_j = \sum_i \left(t_{ij} \cdot I_{(t_{ij} \geq 0)}\right) \Big/ \sum_i I_{(t_{ij} \geq 0)} \tag{2}$$

where $I_{(\cdot)}$ is an indicator function defined as in Eq 1, and $\sum_i I_{(t_{ij}>0)}$ is the total number of metabolic networks in $\boldsymbol{R}$ containing enzyme e_j.

The *TopW* weighting scheme gives higher weights to the enzymes with higher average degree, which strengthens the importance of more highly-connected enzymes.

2.3 Normalization: Eliminating the Influence of Vector Length on Distance

The difference of the vector length can influence the calculation of the distance between vectors. *Iof* and *TopW* weighting schemes help select the 'important' enzymes as the features of organisms, but result in the organism vectors with different lengths. Therefore, it is necessary to normalize the weighted organism vectors before calculating the distance between them.

Let $\boldsymbol{X}$ denotes the weighted and normalized organism-enzyme frequency matrix, where the element x_{ij}, $i = 1..m$, $j = 1..n$, is given by

$$x_{ij} = \frac{w_j \times f_{ij}}{\sqrt{\sum_k \left(w_k \times f_{ik}\right)^2}}, \quad w_j = Iof_j \times TopW_j \tag{3}$$

The rows in $\boldsymbol{X}$ are the final representative of organism vectors.

2.4 Soergel Distance: Calculating the Evolutionary Distance

Soergel distance is one of the distance measures which are commonly used to calculate the evolutionary distance, a crucial measure of the similarity of organism vectors. It has the advantages that its range is limited to 0~1 and it obeys the triangular inequality [17].

Suppose $\boldsymbol{X_A}$ and $\boldsymbol{X_B}$ are two vectors of equal length n, the Soergel Distance between them is defined as:

$$D_{A,B} = \frac{\sum_{j=1}^{n} \left| x_{jA} - x_{jB} \right|}{\sum_{j=1}^{n} \max\left(x_{jA}, x_{jB} \right)} \tag{4}$$

The distance matrix can be established by calculating the Soergel distance between organism vectors pair-wisely. It can be used to construct the phylogenetic trees using a suitable distance-based clustering algorithm, e.g. Neighbor-joining (*NJ*) method.

3 Experiments and Results

In this study, enzyme and reaction data are obtained from the database created by Ma and Zeng [18]. The Ma and Zeng database consists of five tables: *reaction*, *enzyme*, *react*, *connect* and *organism*, and contains 3663 enzymes and 107 organisms (8 Eukaryotes, 83 Bacteria and 16 Archaea) in total. We acquire enzyme frequency information from *enzyme*, and construct the enzyme graphs for each organism from *enzyme* and *reaction*.

Although the *TopEVM* approach is capable of dealing with large collections of organisms, for the sake of concision, in this explanation we select only eight organisms: *rno*, *mmu*, *afu*, *mja*, *nme*, *hin*, *lin* and *bsu*.

Table 1 lists the details of these 8 organisms: their ID in KEGG database (*KEGG ID*), their full name(*Organism*), the Kingdom they belong to (*Kingdom*)□their ID in NCBI Taxonomy [19] (*NCBI Tax Id*), and the number of enzymes they contain (N_E).

Table 1. The details of the 8 organisms

KEGG ID	Organism	Kingdom	NCBI Tax ID	N_E
rno	Rattus norvegicus	Eukaryota	487	416
mmu	Mus musculus	Eukaryota	727	470
afu	Archaeoglobus fulgidus	Archaea	1423	277
mja	Methanococcus jannaschii	Archaea	1642	244
nme	Neisseria meningitides	ProteoBacteria	2190	369
hin	Haemophilus influenzae	ProteoBacteria	2234	386
lin	Listeria innocua	Bacteria Firmicute	10090	388
bsu	Bacillus subtilis	Bacteria Firmicute	10116	504

3.1 Distribution of the *Iof* Weights

According to the definitions in Section 2.1, omitting the absent enzymes in all of the 8 organisms, we calculate the *Iof* weight vector of length 1063. The *Iof* weights values distribute over 8 points. Table 2 displays the value of N_E along the *Iof* Weights. Nearly 30% enzymes have the highest *Iof* weights. 2.1% enzymes will be neglected for their *Iof* importance are 0. Moreover, there are around 60% enzymes with the *Iof* value over 1, and 40% between 0 and 1. Since the *Iof* weighting scheme gives lower weights to the enzymes occurring in a large number of organisms, it comes that the lower the *Iof* value is, the more organisms the enzyme spreads in. This observation also confirms the conclusion of Liu et al [13]. That is, most of the enzymes occur in several organisms they prefer, while only few enzymes occur in most of the studied organisms.

Table 2. The statistics of the number of enzymes along the *Iof* weights

***Iof* Weight**	**2.08**	**1.39**	**0.98**	**0.69**	**0.47**	**0.29**	**0.13**	**0**
Num Enzymes	318	310	102	118	71	85	36	23
Percentage(%)	30	29.1	9.6	11.1	6.7	8.0	3.4	2.1

3.2 Distribution of the *TopW* Weights

In order to calculate the *TopW* weights, we represent the enzyme networks upon the following principles: vertices denote individual enzymes and arcs denote the relationships between them; if one enzyme's product is the substrate of another enzyme, then there's an arc directed from the former enzyme to the latter. The bidirectional arc is replaced by two individual arcs with opposite direction.

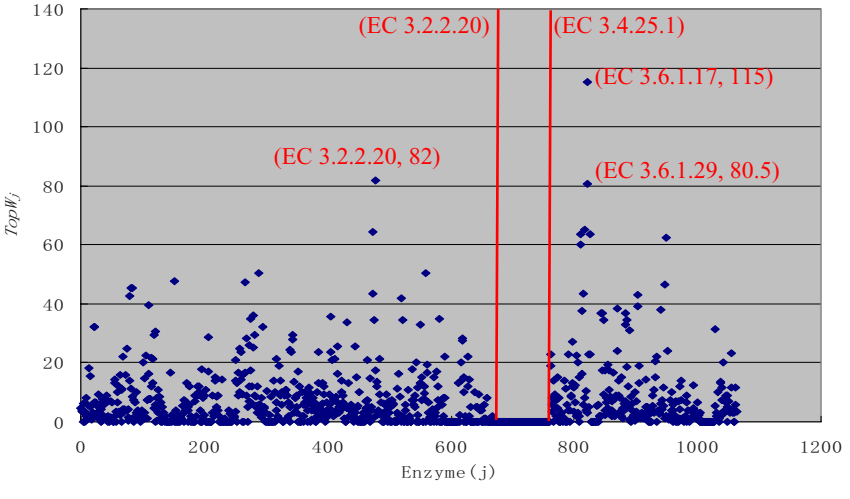


Fig. 2. Distribution of the *TopW* weights. The x-coordinates denote the ordered enzymes. The y-coordinates denote the corresponding *TopW* weight value of the enzyme. The coordinates of the top 3 enzymes are marked on the figure. The continuous range between two vertical dash lines denotes the range of enzymes whose *TopW* weight is 0.

Table 3. The enzymes with the top 8 *TopW* value

Order	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Enzyme	3.6.1.17	2.7.4.10	3.6.1.29	3.6.1.15	2.7.4.6	3.6.1.3	3.6.3.1	4.6.1.1
TopW	115	82	80.5	65	64.25	63.5	63.5	62.3

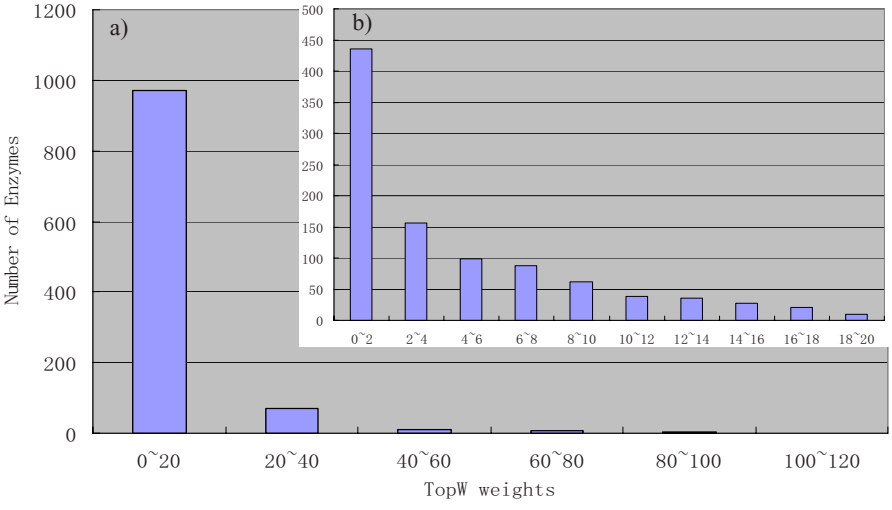


Fig. 3. Distribution of the number of enzymes along the *TopW* weights It contains two diagrams. Fig 3a) shows the distribution of enzymes all over the range of the *TopW* weights. Fig 3b) expands the *TopW* weight range of 0~20.

We construct the *TopW* weight vector for the ordered enzyme array on the basis of the definitions in Section 2.2. Fig.2 shows the distribution of the *TopW* weight values. Most of the *TopW* weights are small but several are very big. For example, *dinucleoside tetraphosphatase* (EC 3.6.1.17)'s *TopW* weight is 115, *AMP phosphotransferase* (EC 2.7.4.10)'s is 82, and *bis(5'-adenosyl)-triphosphatase* (EC 3.6.1.29)'s is 80.5. This observation indicates that as a topology pattern, the *TopW* weights of few enzymes are high, while that of most enzymes are low. We also notice that from *EC 3.2.2.20* to *EC 3.4.25.1*, there is a gap in which 110 enzymes have *TopW* values as 0. They are part of glycosyl hydrolases (*EC 3.2.*-.-), and all of the hydrolases acting on ether bonds (*EC 3.3.*-.-) as well as peptide bonds (*EC 3.4.*-.-). It is mostly due to either the large absence of the enzymes or their possible isolation.

Table 3 displays the enzymes with the top 8 *TopW* weights. It shows that the *hydrolases acting on acid anhydrides* (*EC 3.6.*-.-) have more connection, which means *hydrolases* may be more topologically important than the enzymes with other function.

Fig 3 shows the distribution of the number of enzymes along the *TopW* range. It can be seen in Fig 3a) that more than 90% of enzymes are found within the *TopW* range of

0~20 (the first tallest bar). Fig 3b) expands the first bar of Fig 3a), and displays its details of the distribution. As is shown in Fig 3b), there are 436 enzymes in the *TopW* range of 0~2. The number is nearly 41% of the total number of the enzymes. It indicates that most of nodes have very low connectivity but a handful of nodes (hubs) have much higher connectivity in the constructed enzyme networks. This result is in accordance with the scale free property of metabolic networks, and shows that enzymes become topologically different during evolution [15].

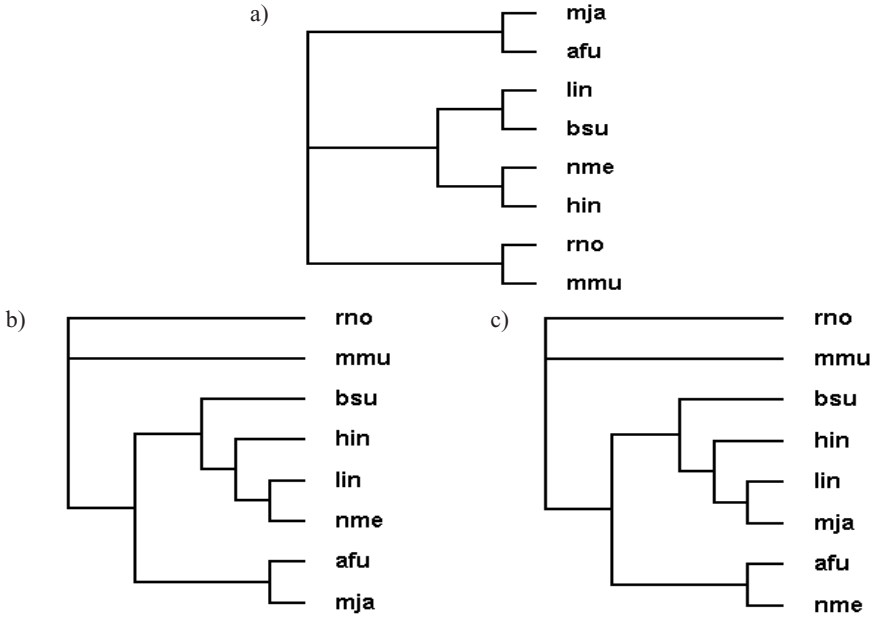


Fig. 4. The comparison of a) the NCBI tree, b) the *TopEVM* tree, and c) the *NCE* tree

3.3 Construction and Evaluation of *TopEVM* Phylogenetic Tree

We calculate the distance for each organism pair by use of Soergel distance, and obtain the distance matrix (Table 4) for constructing the phylogenetic tree. With the help of the Phylip [20] package, we use *NJ* (Neighbor Joining) method to do the construction. The resulting tree is rootless, which is displayed as Fig 5b) by use of TreeView [21]. We also obtain the phylogenetic tree from NCBI Taxonomy (Fig 5a) as the reference, and construct trees using the *NCE* method (Fig 5c) for evaluation.

As is shown in the *TopEVM* tree, the two Archaea *afu* and *mja* are grouped together undoubtedly, which is in line with the taxonomy from NCBI, and so do the two Eukaryotes *rno* and *mmu*. In the *NCE* tree, although *rno* and *mmu* are grouped together, the 4 Bacteria and 2 Archaea are mixed up.

We use *TOPD/FMTS* [22] to evaluate the similarities of trees. This software is complemented with a randomization analysis to test the null hypothesis that the similarity

Table 4. The resulting distance matrix of the 8 organisms upon *TopEVM*

	mmu	*rno*	*afu*	*nme*	*mja*	*hin*	*lin*	*bsu*
mmu	0.0000	0.1079	0.7201	0.7224	0.6754	0.6905	0.7468	0.6790
rno	0.1709	0.0000	0.7613	0.7793	0.7520	0.7613	0.8020	0.7073
afu	0.7201	0.7613	0.0000	0.2737	0.4409	0.4911	0.6667	0.5287
nme	0.7224	0.7793	0.2737	0.0000	0.4471	0.3951	0.6266	0.5617
mja	0.6754	0.7520	0.4409	0.4471	0.0000	0.2395	0.5382	0.3533
hin	0.6905	0.7613	0.4911	0.3951	0.2395	0.0000	0.4438	0.3129
lin	0.7468	0.8020	0.6667	0.6266	0.5382	0.4438	0.0000	0.2486
bsu	0.6790	0.7073	0.5287	0.5617	0.3533	0.3129	0.2486	0.0000

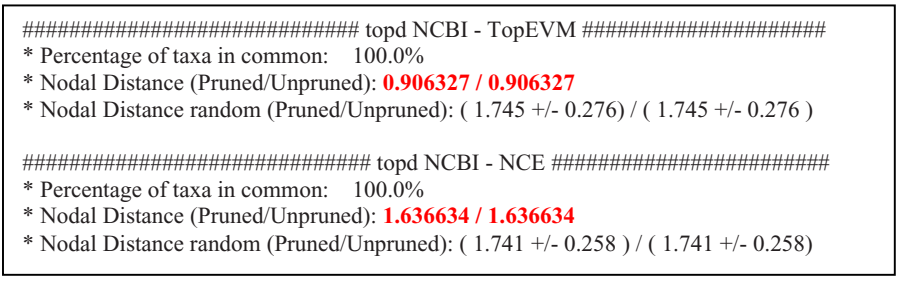


Fig. 5. The comparison result of the *TopEVM* tree and the *NCE* tree with the NCBI tree as reference

between two trees is not better than chance. With the NCBI tree as the reference, the comparison result of the *TopEVM* tree and the *NCE* tree is showed in Fig 6, which shows the *TopEVM* tree is closer to the NCBI tree with a less Nodal Distance [23] as 0.9.

4 Conclusion and Future Work

This paper proposes a new pattern recognition technique, *TopEVM*, which represents the metabolic networks as weighted vectors. By calculating the distances among these weighted vectors, evolutionary distance matrices are determined for the construction of phylogenetic trees. Comparing to the previous set-theoretic methods, our *TopEVM* method results a phylogenetic tree closer to the taxonomy tree of NCBI, which shows *TopEVM* can be a very useful approach for the network-based phylogenetic analysis.

Nevertheless, our experiments so far have considered only the *Tf-Idf* weighting scheme to integrate enzyme's frequency content. It is hard to say that there is no other weighting scheme which is more suitable. Besides, among the extensive topological indices, we only considered the degree centrality; we would also like to consider more topological information for improving our model further.

Acknowledgments. We thank Dr. Hong-Wu Ma and Prof. Dr. An-Ping Zeng for sharing their revised metabolic database. We also thank Dr. Puigbò for the direction on the use of TOPD/FMTS. This work is supported by the National Natural Science Foundation of China (6077 3021).

References

1. Husmeier, D.: Introduction to Statistical Phylogenetics. In: Husmeier, D., Richard Dybowski, R., Roberts, S. (eds.) Probabilistic Modeling in Bioinformatics and Medical Informatics. Springer, Heidelberg (2006)
2. Holmes, E.C., et al.: Using Phylogenetic Trees to Reconstruct the History of Infectious Disease Epidemics. In: Harvey, P. (ed.) New Uses for New Phylogenies. Oxford University Press, Oxford (1996)
3. Ludwig, W., Schleifer, K.: Bacterial Phylogeny Based on 16S and 23S rRNA Sequence Analysis. FEMS Microbiol Rev. 15(2-3), 155–173 (1994)
4. Wolf, Y.I., et al.: Genome Trees and the Tree of Life. Trends in Genetics 18(9), 472–479 (2002)
5. Pal, C., Papp, B., Lercher, M.J.: Adaptive Evolution of Bacterial Metabolic Networks by Horizontal Gene Transfer. Nature Genetics 37(12), 1372–1375 (2005)
6. Ebenhöh, O., Handorf, T., Heinrich, R.: A Cross Species Comparison of Metabolic Network Functions. Genome Informatics 16(1), 203–213 (2005)
7. Tohsato, Y.: A Method for Species Comparison of Metabolic Networks Using Reaction Profile. IPSJ Digital Courier 2(0), 685–690 (2006)
8. Forst, C.V., Flamm, C., Hofacker, I.L., Stadler, P.F.: Algebraic Comparison of Metabolic Networks, Phylogenetic Inference, and Metabolic Innovation. BMC Bioinformatics 7(1), 67–78 (2006)
9. Zhou, T., Chan, C., Pan, Y., Wang, Z.: An Approach for Determining Evolutionary Distance in Network-Based Phylogenetic Analysis. In: Măndoiu, I., Sunderraman, R., Zelikovsky, A. (eds.) ISBRA 2008. LNCS (LNBI), vol. 4983. Springer, Heidelberg (2008)
10. Ma, H.W., Zeng, A.P.: Phylogenetic Comparison of Metabolic Capacities of Organisms at Genome Level. Molecular Phylogenetics and Evolution 31(1), 204–213 (2004)
11. Aguilar, D., et al.: Analysis of Phenetic Trees Based on Metabolic Capabilities Across the Three Domains of Life. Journal of Molecular Biology 340(3), 491–512 (2004)
12. Zhu, D., Qin, Z.S.: Structural Comparison of Metabolic Networks in Selected Single Cell Organisms. BMC Bioinformatics 6(8) (2005)
13. Liu, W., Lin, W., Davis, A., Jordan, F., Yang, H., Hwang, M.: A Network Perspective on the Topological Importance of Enzymes and Their Phylogenetic Conservation. BMC Bioinformatics 8(121) (2007)
14. Aizawa, A.: An Information-theoretic Perspective of Tf-idf Measures. Information Processing and Management 39(1), 45–65 (2003)
15. Light, S., Kraulis, P., Elofsson, A.: Preferential Attachment in the Evolution of Metabolic networks. BMC Genomics 6(1), 159 (2005)
16. Aittokallio, T., Schwikowski, B.: Graph-based Methods for Analyzing Networks in Cell Biology. Briefings in Bioinformatics 7(3), 243 (2006)
17. Willett, P., Barnard, J.M., Downs, G.M.: Chemical Similarity Searching. J. Chem. Inf. Comput. Sci. 38(6), 983–996 (1998)

18. Ma, H.W., Zeng, A.P.: Reconstruction of Metabolic Networks from Genome Data and Analysis of Their Global Structure for Various Organisms. Bioinformatics 19(2), 270–277 (2003)
19. NCBI Taxonomy, http://www.ncbi.nlm.nih.gov/Taxonomy/
20. Phylip, http://evolution.genetics.washington.edu/phylip.html
21. TreeView, http://taxonomy.zoology.gla.ac.uk/rod/treeview.html
22. Puigbo, P., Garcia-Vallve, S., McInerney, J.O.: TOPD/FMTS: A New Software to Compare Phylogenetic Trees. Bioinformatics 23(12), 1556 (2007)
23. Bluis, J., Shin, D.G.: Nodal Distance Algorithm: Calculating a Phylogenetic Tree Comparison Metric. In: Proceedings of third IEEE Symposium on Bioinformatics and Bioengineering (2003)

Generating Synthetic Gene Regulatory Networks

Ramesh Ram and Madhu Chetty

Gippsland School of IT, Monash University,
Churchill, Victoria 3842, Australia
{Ramesh.ram,Madhu.chetty}@infotech.monash.edu.au

Abstract. Reconstructing GRN from microarray dataset is a very challenging problem as these datasets typically have large number of genes and less number of samples. Moreover, the reconstruction task becomes further complicated as there are no suitable synthetic datasets available for validation and evaluation of GRN reconstruction techniques. Synthetic datasets allow validating new techniques and approaches since the underlying mechanisms of the GRNs, generated from these datasets, are completely known. In this paper, we present an approach for synthetically generating gene networks using causal relationships. The synthetic networks can have varying topologies such as small world, random, scale free, or hierarchical topologies based on the well-defined GRN properties. These artificial but realistic GRN networks provide a simulation environment similar to a real-life laboratory microarray experiment. These networks also provide a mechanism for studying the robustness of reconstruction methods to individual and combination of parametric changes such as topology, noise (background and experimental noise) and time delays. Studies involving complicated interactions such as feedback loops, oscillations, bi-stability, dynamic behavior, vertex in-degree changes and number of samples can also be carried out by the proposed synthetic GRN networks.

Keywords: Causal model, synthetic gene regulatory networks, microarrays.

1 Introduction

The reconstruction of gene regulatory networks (GRNs) using microarray datasets is amongst the major challenges currently being investigated in the field of molecular biology research. Reconstruction of GRN provides researchers an opportunity to form new hypotheses related to the behavior of biological systems a-priori to the experiments to be carried out, which in turn prevents performing expensive and lengthy biological experiments thereby expediting the discovery process. In domains other than bioinformatics, datasets studied typically have very few features and large number of samples. However, in case of microarray dataset, they invariably have large number of genes with very few samples. Consequently, traditional statistical approaches for analyzing these microarrays become inadequate and the need for applying other techniques becomes necessary. A plethora of modeling and inference techniques, starting from standard multivariate statistics to machine-learning and

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 237–249, 2008.

heuristics [1-5], are available to reconstruct GRN from large-scale gene expression data sets. While using these techniques, the validation and evaluation of methods using real life datasets is limited since such datasets may not always be properly documented. In other words, the validation is limited by the information that was previously established independently by other approaches.

A synthetically reconstructed GRN, while preserving the characteristics of the underlying data generation system, allows different experiments to be performed to investigate the effect of parametric variations. A synthetic dataset permitting large variety of parametric variations is a possible solution which will allow more rigorous testing and evaluation of methods for reconstructing GRNs. Although limited, efforts for generating synthetic data for GRN reconstruction are available. Mendez *et al* [6] proposed a method based on differential equations for generating synthetic microarray data. However, the methods allows variation of only noise and topology parameters and does not include the flexibility of varying single or combination of parameters for validating individual features of the GRN methods. Eisen *et al* [7] generated synthetic dataset and applied for studying hierarchical clustering for gene expression data. As the method suffered from the lack of knowledge about the GRN under study, any conclusion vis-à-vis the underlying biology became uncertain. Further, because the data sets were different in each of the studies carried out, it was not possible to make any comparisons amongst studies that employed this approach. Friedman *et al* [1] generated a Boolean synthetic data to validate the robustness of their Bayesian methods. Although useful for generating synthetic datasets, none of these techniques were suitable to examine model specific features such as time-delays, feedback loops, dynamic behavior, etc. Furthermore, all these techniques were limited in their ability to generate a variety of synthetic networks at different stages of refinement of GRN reconstruction methods.

In this paper, we present a novel causal modeling method for synthetically generating GRN which includes all GRN related features that are commonly modeled in reconstruction algorithms. The variation of these features, in a controlled way, determines the desired level of complexity of the synthetically generated gene expression data. The proposed synthetic generation of networks is along the lines of our ongoing work on causal modeling for reconstruction of real-life GRN [8-12] wherein we have investigated the application of causal modeling technique to *Saccharomyces cerevisiae* (yeast) [15] microarray dataset. The obtained results are in close agreement with known biological findings thus validating the modeling process. Briefly, the causal GRN reconstruction from microarray begins with the application of a network structure construction algorithm resulting in a large number of possible GRN structures. A continual evaluation and evolution of these structures results in a structure that best fits the microarray data results and is considered as the desired GRN model. The complexity of reconstruction is increased gradually at each stage of refinement of the model. The rest of the paper is structured as follows: Section 2, a brief overview of causal modeling is given. Section 3 elaborates on the system and methods used to realistically generate the synthetic data. Section 4 provides experiments and results. Finally, section 5 provides concluding remarks on the paper and some future work.

2 The Causal Modeling Approach

A causal GRN structure is represented by a directed graph whose nodes represent the genes and directed edges between nodes indicate their causal relationship. Pioneering work in causal modeling was reported by Pearl *et al*, and Sprites *et al* [13, 14] who proposed algorithms to infer a causal structure from experimental data by using partial correlations if the underlying causal structure was a directed, acyclic graph (DAG). In our approach for causal GRN reconstruction, the central step of determining the fitness of the data given the whole network is decomposed into determining the set of scores of local models that includes fitness of structure, direction of causality and sign (positive/ negative) of regulation. The task of network reconstruction is cast into a search for candidate gene networks whose scores are high. To implement a heuristic search method, we apply a genetic algorithm (GA), whereby creating and evolving different networks to eventually obtain a network that best fits the microarray data. Due to the stochastic nature of the GA, the GA is repeated few times and the resulting network structures are combined in a predefined manner to reconstruct the final gene network. While evaluating the fitness, the putative network is actually decomposed into Markov Blankets (MB) and conditional independence tests are applied in order to detect whether or not connections are direct or indirect. The direction and sign of regulation are recovered by estimating the time delay and correlation between expression profiles of pairs of genes.

3 Methodology for Generation of Synthetic Data

The proposed method for generation of synthetic networks allows for various parametric variations, such as, network topology, varying levels of complexity of interaction, time delays, number of samples and amount of noise in the data.

Figure 1 shows the flow chart of the mechanism of proposed system for synthetic network generation. The synthetic network generator, written in MATLAB, offers an option for choice of topologies that determines the structure of the network and specifies interactions between the genes. With this option, we can generate any number of networks having different topologies. In the next step, by choosing interactions and setting equation parameters, the full dynamics of the gene network (such as feedback loops, oscillations and so on) is described and can be implemented in specified pre-defined ways to produce a required level of complexity of gene interactions. Next, for generating discrete samples, the continuous responses of the genes in the synthetic network are sampled at different time instants to produce a noiseless time course data. Next, to make the sampled data realistic, time delays are added to the samples in a specified manner. Following this, noise is added to the data according to the Gaussian or gamma distributions. Finally, gene expression ratios are calculated which realistically represent the real-life microarray data set.

In the following section, the entire process of generating the network topology and corresponding gene interactions is described in detail.

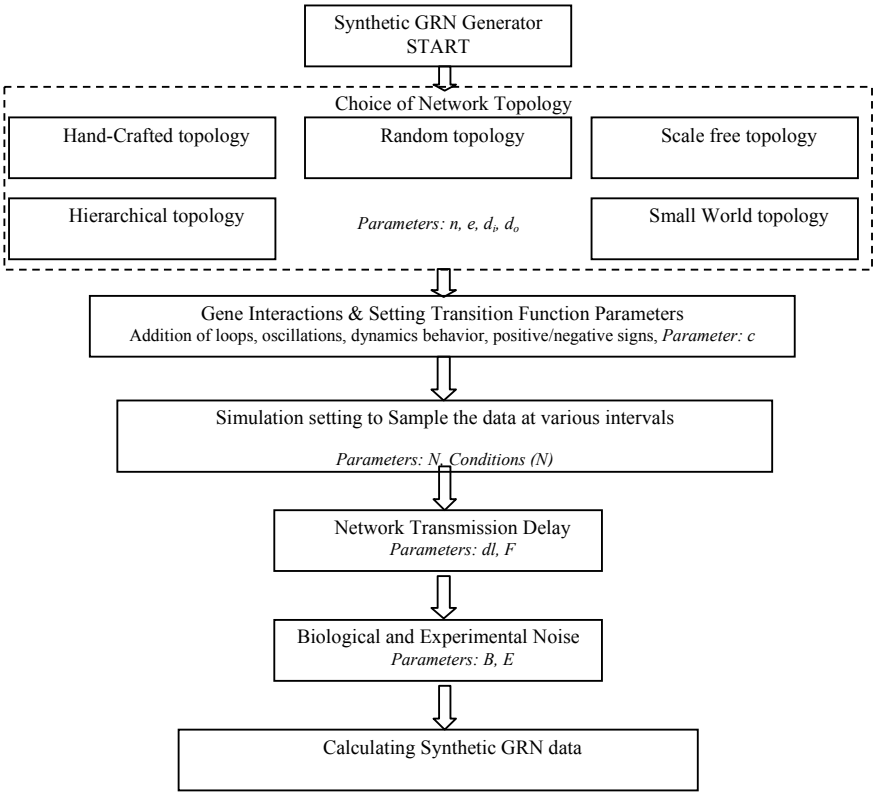


Fig. 1. Proposed methodology of synthetic gene expression data generation, The symbols used are: n - number of genes, e – number of edges, di – incoming degree distribution, do – outgoing degree distribution, c – percentage of complex interactions, N – number of samples, Condition (N) – specifies experimental conditions for each sample as in real each sample is an experiment, dl – delay levels, F – probability distribution of delays, B – percentage of biological noise in terms of hidden nodes, E – percentage of experimental noise.

3.1 Network Topology

As mentioned earlier, the first step of synthetic data generation is to define a network topology. A topology is chosen by setting following three parameters:

i) Total number of genes in the network,
ii) Distribution of the incoming degree of connectivity (i.e. the distribution of the number of parents per gene) and
iii) Distribution of the outgoing degree of connectivity (i.e. the distribution of the number of children per gene).

Based on the incoming and outgoing degree distribution parameters mentioned above, four different topologies are available for selection (with corresponding distribution provided in parenthesis):

- Random topology (Poisson distribution)
- Scale Free topology (power law distribution)
- Small World topology (power law distribution with small average distance between genes)
- Hierarchical topology (power law distribution with inherent modular structure)

In random topology (RND), the connectivity degree follows a Poisson distribution. The nodes that deviate from the average are rare and decrease exponentially and the clustering coefficient is independent of a node's degree of connectivity [16]. In Scale Free (SF) topology [17], the connectivity degree follows a power law distribution, i.e. the behavior of a network system is controlled by few important nodes. Majority of nodes have only a few connections, while some special nodes connect with many other nodes forming a hub, i.e., most nodes are poorly connected, while a few are highly connected (Hubs). In a Small World networks (SW) [18], the mean shortest path is $l \sim \log(N)$ indicating that most nodes are connected by a short path. SW networks are characterized by large Clustering Coefficient and small Average Path Length. The Hierarchical network (HR) [19] integrates a scale-free topology with an inherent modular structure by generating a network that has a power-law degree distribution with degree exponent $\gamma = 1 + \ln 4/\ln 3 = 2.26$.

In cases where the aforementioned topological types are not appropriate due to the uncertainty of GRN topology, we propose another topology, which we will refer henceforth as, 'handcrafted topology'(HC). The choice of any of the network topology is user-definable and can be used for checking robustness of algorithm against topology. To generate a network topology close to real life GRN, network structures previously described in biological literature such as E. coli [20] and S. cerevisiae [21] were taken into account. These networks are partially random and partially scale free i.e. the distribution of the incoming degree of connectivity follows a Poisson distribution (random topology) while the distribution of the outgoing degree of connectivity follows a power-law (scale free topology). A single topology or combinations of two or more topologies to generate the gene network structure is user definable.

At this stage, the network structure is without any complex interactions, such as self loops, oscillations and dynamic behaviour. In the next section, we present the inclusion of these features to the network topology.

3.2 Gene Interactions and Transition Function Parameters

After generating the topology, transition functions representing the regulatory interactions between the genes are assigned to the edges in the network as follows:

i) Choosing the regulatory interactions
ii) Setting the transition function parameters

The entire synthetic modeling of gene networks essentially considers a causal interaction of genetic regulation. It considers each gene to be directly affected by number of other genes and represents the interaction as directed edges. A transition function defines the relationship between gene and its parent genes. The genes are

represented as continuous variables rather than discrete variables, i.e. synthetic gene expression values are continuous rather than 0 or 1. First, while choosing the regulatory interactions, the genes are represented as activators or repressors. Our proposed method of network modeling allows for this positive or negative linear causal relationship between the input (i.e. parent) genes and the gene under consideration. Mathematically, these network models are based on set of linear causal equations. Each equation corresponds to gene expression which is a function of a positive (activation) and negative (repression) terms. When a given gene interacts with more than one regulator, different regulators can either act independently or in a more complex manner (such as complex combinational, short term co-activation, co-repression or a combination) on the target genes resulting in different interactions such as feedback loops, oscillations and dynamic behavior.

To incorporate such complexities, for each combination of a gene and its regulators, appropriate equation is selected, depending on the number of activators and repressors and on the user-defined settings that control the fraction of complex interactions. For genes involved in *cycles*, it is possible that not all inputs of their transition function are known during loop propagation. To model these loops, an approximation compatible with the steady-state transition functions is chosen. This approximation is represented by a parameter to represent complex interactions. It is an extremely useful parameter because it allows initial performance evaluation of a method to be done on relatively easy problems (e.g. small noiseless networks without complex interactions between regulators). Increasingly difficult data sets can subsequently be generated as the GRN inference method is improved or refined. Again, setting transition function parameters involves choosing appropriate correlation parameter settings of the transition function equations. The strength of correlation is an important parameter and is chosen from a distribution that allows a large variation of interaction that are likely to occur in true networks (including linear activation functions, sigmoid functions, sinusoidal functions, etc.), while avoiding very steep transition functions. To explain a simple chain interaction in the network considers, for example, that x causes y and y causes z. That is, $x \rightarrow y \rightarrow z$

$$x(t) = A\sin(Bt) \; ; \; y(t) = x(t) \; ; \; z(t) = y(t) \tag{1}$$

The expression x(t) is a sinusoid with amplitude A, time period $2\pi/B$ where B is angular frequency. In this case, the strength of correlation between x and y is 1, so the signals are equal, but varied based on parametric specification.

3.3 Data Samples

Using the continuous gene expression output (resulting from the equations written for each node of the synthetic network), data is sampled at either fixed or irregular time spacing between gene expressions. The number of samples and the time step for sampling can be chosen either randomly or it can also be user defined. The sampled data represents the temporal state of synthetic network under different experimental conditions. This is similar to real microarray experiments where each sample of the dataset is an experiment that is repeated at fixed or irregular intervals of time. At this stage, various settings needed for simulation of the network per each sample (simulating a real experiment setup) for N sample is complete. However, note that the

data representing real life conditions is not yet generated as time delay and noise component are yet to be added.

3.4 Network Transmission Delays

A delay in transmission of signals emitted by genes, being an important characteristic of all gene networks; it is important to realistically implement this feature in synthetic datasets. In the proposed modeling approach, we implement the delay levels as a user defined parameter which is nothing but the maximum number of samples on which the delay can be experienced. Further, to make the modeling more realistic, we have also made it possible to specify the fraction of interactions which have delays. Based on the choice of this parameter, a delay distribution is obtained for the links between the genes. Delays are implemented by simply reassigning a new simulation setting for a particular sample explained in section 3.2 based on the delays assigned. This simulates the delay in the real microarray dataset. The fraction of links involved in time delay is determined using a known probability density in case it is not user defined. Investigations involving time delay parameter variation can thus be carried out on the datasets by incorporating/eliminating time delays.

3.5 Biological and Experimental Noise

A real life microarray data contains two types of noises, namely biological and experimental. The biological noise corresponds to stochastic variations in gene expression, and this noise is unrelated to the applied experimental procedures. It is present due to, for example, environmental conditions such as temperature, pressure, etc. While experimental noise is the noise due to the technique used to extract the data. Both these noises also should be appropriately included in the simulated data.

Briefly, biological noise is added by the presence of hidden background nodes which are either genes or conditions and experimental noise is added as Gaussian white noise. First, the background hidden node (for incorporating biological noise), which is a parameter to choose the amount of background noise, is user defined. The equations of the background noise nodes are generally uncorrelated to the genes on which they are acting. A limited number of input nodes are selected that mimic the external conditions and consider the genes not linked to these input genes act as background nodes. These are now part of the simulation set up while the data is not generated.

As the real microarray data also has experimental noise, three user defined choices for addition of experimental noise are made available: i) Lognormal ii) Gaussian iii) Gamma distributions. All these distributions take a percentage of the amount of noise as input which is then applied to make the final output data noisy. However, this experimental noise is added only after the simulated microarray data is generated. This is explained in section 3.7.

3.6 Synthetic Network Generator Parameters

The entire flow chart for the generation of synthetic data is given in Fig. 1 which also shows the system and the parameters controlling the synthetic data generation at every step of the process. These parameters which are listed below can each be varied

independently either before or during the simulation process for conducting simulated experiments with synthetic data:

1. Choice of source network.
2. Size of the network in number of nodes.
3. Number of background nodes.
4. Number of available experiments and samples for each condition.
5. Level of stochastic and experimental noise.
6. Fraction of complex interactions.

3.7 Calculating Synthetic GRN Data

Using the synthetic network generator described earlier, simulations are next performed to generate the synthetic microarray data. The genes without regulatory inputs are assigned an arbitrary expression level which can be changed during an experiment (sample). The expression levels of the genes in the network are calculated, as specified by their transition functions, starting from the input genes. After these noise-free expression values are computed, noise is then appropriately incorporated in the data to reflect noise present in the real microarray data. These computed noisy expression values can be used for analyzing the noise which a GRN reconstruction method under investigation can handle. This feature of adding noise enables the comparison of level of noise in dataset on the reconstruction algorithms. A gene expression profile experiment for different time t corresponds to a vector $[x_1(t) \ldots x_n(t)]$. For a set of N samples, a n x N matrix is constructed which is the *final synthetically generated microarray dataset.* This dataset can be used for investigation and evaluation of various GRN reconstruction algorithms.

4 Experiments and Results

In order to conduct tests using synthetic data set, several datasets are created by varying network generator parameters (one or two at a time). The group of data sets which have similar variations are categorized into one of the four groups A, B, C or D (see Table 1). Although the experiments involved significantly large number of data sets to test robustness of GRN methods, due to space restriction, only a limited number of important models have been included in the paper and shown in Table 1.

The Group A consists of a set of synthetic network models which are used for investigating methods for their robustness against network topology. With this group, we carry out an initial level of testing since it contains no complex interactions and also because the effect of the noise is kept low. Different sample sizes help determine accuracy of reconstruction as generally most methods require higher sample size data to make accurate estimations.

The Group B networks compare two different network topologies, namely SF and RND. Compared to Group A, these are large sized networks of 500 genes and 500 interactions. Fig.2 (a) shows networks that follow a random topology (RND) while the network shown Fig. 2 (b) is a scale-free (SF) network. From the figure, we can observe the differences resulting due to two differing topologies. The random topology has arbitrary arrangement of links throughout the network while the scale

Table 1. The Synthetic data sets are organized in four groups A, B, C, D. Column 2 gives different network topologies: Scale Free (SF), Small World (SW), Random (RND) and Handcrafted (HC). For each group, column 3 shows the number of repeated models for a given experiment. Column 4 and column 5 respectively give the number of genes and the edges in a given model. Column 6 gives the % fraction of complex interactions. Column 7 gives the network transmission delay. Column 8 gives the number of parents while column 9 gives the %ge noise of each model. Column 10 gives number of samples for each condition.

1	2	3	4	5	6	7	8	9	10
Group	Topology	No. of Models	Genes	Edges	% Complexity	Delay	No. of parents	% Noise	Samples
A	SF	50	100	200	20	0	2	1	20, 50, 100
	SW	50	100	200	20	0	2	1	20, 50, 100
	RND	50	100	200	20	0	2	1	20, 50, 100
	HC	50	100	200	20	0	2	1	20, 50, 100
B	SF	5	500	500	40	2	5	5	50
	RND	5	500	500	40	2	5	5	50
C	SF	50	50	50	20	1	3	5	50
	SF	50	50	100	20	1	4	5	50
	SF	50	50	200	20	1	7	5	50
D	SF	10	100	200	40	1	1	5	20, 50
	SF	10	100	200	30	2	2	1	20, 50
	SF	10	100	200	50	-2	3	5	20, 50
	SF	10	100	200	10	3	4	1	20, 50
	SF	10	100	200	40	-3	4	5	20, 50
	SF	10	100	200	30	4	5	1	20, 50
	SF	10	100	200	50	0	3	10	20, 50
	SF	10	100	200	10	0	2	1	20, 50

free network has hubs with large proportion of links in the top right corner of the figure while lesser number of links in the rest of the figure. Note that the number of genes and gene interactions is the same for the two cases under consideration. Since scalability is an important feature of GRN algorithms, this group enables to justify if the algorithm is robust in terms of size.

In Group C, the number of genes in the networks is kept fixed at 50 and the topology chosen for study is Scale Free. The number of links is varied as 50, 100, 200. This group is useful for checking robustness of methods with respect to density of connectivity (i.e. no. of parents per gene) along with accuracy with respect to number of samples.

The Group D is designed to test the combinational effect of density of connectivity and also to include varying delays and noise intensity parameters resulting in an increasing average number of connections per gene. The D group tests are for advanced level testing of GRN algorithms as the data generated is from a complex complicated network of interactions. Because these gene networks are generated with random connectivity for each of the rows in Table 1, we repeated the generation of models for specified number of times (see column 3) and took the average results

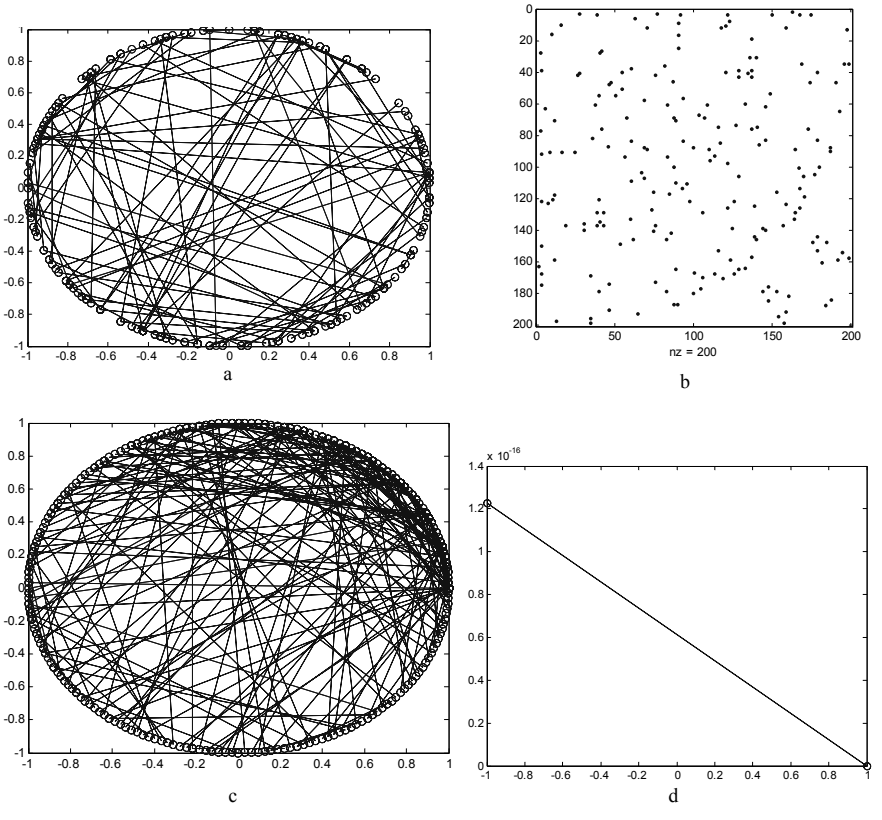


Fig. 2. Two illustrative network topologies of B Group: a- RND Network Topology (n=200, e = 100, di = dp = Poisson pdf), b – Adjacency matrix (n x n) of the random topology, c - Scale Free Network Topology (n=200, di = dp = Power law pdf), d – Power Law function, f(x) on y-axis.

from each row to get a synthetic dataset which is close to real dataset. The simulation results from this synthetic dataset are shown in Figure 3.

For these simulations, an example network given in Table 1 is considered. . The part of the network under consideration (also known as Markov Blanket of gene A is shown in figure 3(a). For investigations involving noise, we consider four different types of noise type's namely a) Lognormal b) Gaussian c) Linear and d) Constant. The variation of these noises as a function of gene expression is shown in Fig 3(b). For purpose of experimentation, the amount of noise is added as a function of the gene expression (i.e. mRNA accumulation). In Figure 3(c), the expression of synthetic gene A is shown for both conditions: with and without noise. These plots show the effect of noise on the synthetic gene expression. Again, Figure 3(d) shows the effect of an input gene (such as gene B) and an output gene (such as gene D) on another gene A in the presence of time delay and regulation (plus or minus). As can be seen

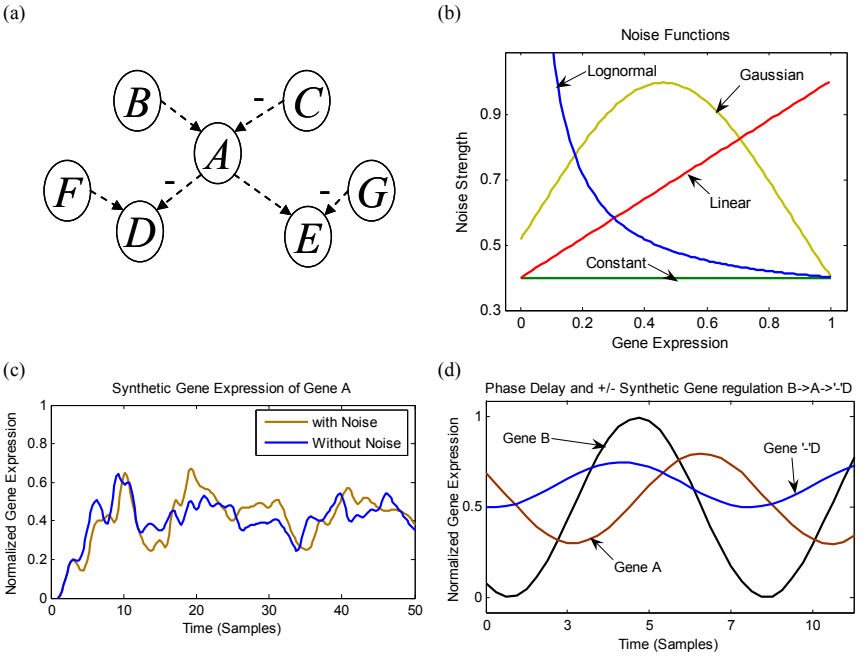


Fig. 3. Simulation results: (a) A subset of genes of the example network (labeled A to G). This sub-network has two input genes and contains repressor gene D. (b) The different noise functions used in the simulation. (c) The expression gene A with and without addition of noise. (d) Shows phase shift (time delay) and plus/ minus regulation between B→A→D.

from Figure 3(a), gene B has a strong effect on gene A, but a pronounced negative effect on the expression level of gene D. This is because gene D has a repressor link and is directly stimulated by gene A and also indirectly by input gene B.

5 Conclusion

The network generator system presented in this paper generates synthetic GRN datasets based on causal modeling approach for GRN [8]. Illustrative investigations using the network generator show the significance of the application of system for synthetic data generation. The proposed system can generate four different network topologies, namely scale free, small world, random and hierarchical. Further, the generated synthetic network is made realistic by incorporating complex network characteristics such as transmission delays, biological and experimental noise. These datasets are generated for evaluation of methodologies based on these synthetic datasets. The system will help other similar methods to computationally determine the robustness and also establish comparisons between the methods. In comparison to other existing methods, the proposed system is useful in carrying out rigorous studies about the GRN methods by particularly varying single and combinational features of the networks such as the topology, interaction types, noise levels, time delay of the

interactions and so on. The complexity of the generated datasets can be easily varied by parametric variation thereby generating a hierarchy of networks in terms of size, scales, samples, etc. In real life GRNs, the products of some specific genes are essential for transcription to take place (e.g. mRNA), and their absence cannot be counteracted by increased expression of other activators (e.g. the transcription factor TF). Although the synthetic gene network models and datasets generated here are simplistic in comparison to what actually happens between genes in the real world biology to produce the microarray dataset, they mimic the characteristics of experimental data which makes it suitable to test the methods used on real datasets. Furthermore, since the ground truth about real world GRN is still unknown to a certain extent and hence to make any significant advances towards understanding gene networks by using artificial synthetic networks and datasets will be highly important and useful for future analysis of very complex GRN models.

Although only additive interactions for different hidden nodes have been included in this paper, it is easily possible to enhance the system further by including non-additive interactions between the activators and repressors. All these improvements in generation of synthetic networks will make the network models more realistic. With a clear understanding gene regulation problem, it would be possible to simulate the process of complex gene regulatory networks. Future developments in this research will include exploring large numbers of diverse synthetic gene networks to search for particular properties similar to real gene networks, and expanding the system to protein-protein interaction networks.

References

[1] Friedman, N., Linial, M., Nachman, I., Pe'er, D.: Using Bayesian networks to analyse expression data. Journal on Computational Biology 7, 601–620 (2000)
[2] Liang, S., Fuhrman, S., Somogyi, R.: REVEAL, a general reverse engineering algorithm for inference of genetic network architecture. In: Pacific Symposium on Biocomputing, vol. 3, pp. 18–29 (1998)
[3] Ando, S., Iba, H.: Inference of gene regulatory model by genetic algorithms. In: Proc. Conference on Evolutionary Computation, pp. 712–719 (2001)
[4] Wahde, M., Hertz, J.: Modeling genetic regulatory dynamics in neural development. Journal on Computational Biology 8, 429–442 (2001)
[5] Mendes, P., Sha, W., Ye, K.: Artificial gene networks for objective comparison of analysis algorithms. Bioinformatics 19, 122–129 (2003)
[6] Ram, R., Chetty, M., Dix, T.I.: Fuzzy Model for Gene Regulatory Networks. In: 2006 IEEE Congress on Evolutionary Computation (CEC) (2006)
[7] Eisen, M.B., Spellman, P.T., Brown, P.O., Botstein, D.: Cluster analysis and display of genome-wide expression patterns. Proc. Natl. Acad. Sci. USA 95, 14863–14868 (1998)
[8] Ram, R., Chetty, M., Dix, T.I.: Causal Modeling of Gene Regulatory Network. In: IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB) (2006)
[9] Ram, R., Chetty, M., Dix, T.I.: Learning Structure of Gene Regulatory Networks. In: 6th IEEE International Conference on Computer and Information Science (ICIS) (2007)
[10] Ram, R., Chetty, M.: A Guided genetic algorithm for Gene Regulatory Network. In: 2007 IEEE Congress on Evolutionary Computation (CEC) (2007)

[11] Ram, R., Chetty, M.: Framework for path analysis for learning Gene regulatory network. In: Pattern Recognition in Bioinformatics. Springer – LNBI publication, Heidelberg (2007)
[12] Ram, R., Chetty, M.: Modelling Gene regulatory networks. In: Applications of Computation Intelligence in biomedicine and Bioinformatics. Springer, Heidelberg (accepted, 2007)
[13] Sprites, P., Glymour, C., Scheines, R.: Causation, Prediction, and Search: Adaptive Computation and Machine Learning, 2nd edn. MIT Press, Cambridge (2000)
[14] Pearl, J.: Causality: Models, Reasoning and Inference. Cambridge University Press, Cambridge (2000)
[15] Spellman, P.T., Sherlock, G., Zhang, M.Q., Iyer, V.R., Anders, K., Eisen, M.B., Brown, P.O., Botstein, D., Futcher, B.: Comprehensive identification of cell cycle-regulated genes of the yeast Saccharomyces cerevisiae by microarray hybridization. Mol. Biol. Cell. 9, 3273–3297 (1998)
[16] Erdös, P., Rényi, A.: On random graphs. Publ. Math. Debrecen 6, 290–297 (1959)
[17] Barabasi, A.L., Albert, R.: Emergence of scaling in random networks. Science 286, 509–512 (1999)
[18] Calvert, K.L., Doar, M.B., Zegura, E.W.: Modeling Internet topology. IEEE Communications Magazine 35, 160–163 (1997)
[19] Rahmel, J.: SplitNet: A Dynamic Hierarchical Network Model. In: AAAI/IAAI, vol. 2, p. 1404 (1996)
[20] Featherstone, D.E., Broadie, K.: Wrestling with pleiotropy: genomic and topological analysis of the yeast gene expression network. Bioessays 24, 267–274 (2002)
[21] Jeong, H., Tombor, B., Albert, R., Oltvai, Z.N., Barabási, A.-L.: The large-scale organization of metabolic networks. Nature 407, 651–654 (2000)

Gene Selection for Microarray Data by a LDA-Based Genetic Algorithm

Edmundo Bonilla Huerta, Béatrice Duval, and Jin-Kao Hao

LERIA, Université d'Angers
2 Boulevard Lavoisier, 49045 Angers, France
{edbonn,bd,hao}@info.univ-angers.fr

Abstract. Gene selection aims at identifying a (small) subset of informative genes from the initial data in order to obtain high predictive accuracy. This paper introduces a new wrapper approach to this difficult task where a Genetic Algorithm (GA) is combined with Fisher's Linear Discriminant Analysis (LDA). This LDA-based GA algorithm has the major characteristic that the GA uses not only a LDA classifier in its fitness function, but also LDA's discriminant coefficients in its dedicated crossover and mutation operators. The proposed algorithm is assessed on a set of seven well-known datasets from the literature and compared with 16 state-of-art algorithms. The results show that our LDA-based GA obtains globally high classification accuracies (81%-100%) with a very small number of genes (2-19).

Keywords: Linear discriminant analysis, genetic algorithm, gene selection, classification, wrapper.

1 Introduction

The DNA Microarray technology permits to monitor and to measure gene expression levels for tens of thousands of genes simultaneously in a cell mixture. Several studies have demonstrated that expression profiles provide valuable information for cancer diagnosis and prognosis [1,2,3,9]. The ability to distinguish a cancer from morphologically similar tissues using their gene expression profiles is important to propose appropriate therapies. Classification of different tumor types is intertwined with the problem of gene selection, which aims to extract from a great number of genes monitored by a Microarray chip, a small subset of discriminant genes. Gene selection is thus of practical and fundamental interest. The identification of relevant biomarkers is necessary for the elaboration of medical diagnostic tests. Knowledge about discriminant gene subsets may confirm the understanding of cancer mechanisms and suggest new ideas to explore.

Two main approaches have been proposed for gene selection. Filter methods rely on a criterion that depends only on the data to assess the importance or relevance of each gene for class discrimination. A relevance scoring provides a ranking of the genes from which the top-ranking ones are generally selected as the most relevant genes. Filter methods ignore the correlations among genes and the

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 250–261, 2008.

interaction of the selected genes with the classifier. Wrapper approaches embed gene subset selection and evaluation with the same process and consequently overcome the above mentioned inconvenient.

In this paper, we propose a new wrapper approach for gene subset selection and classification of Microarray data. Our approach uses Fisher's Linear Discriminant Analysis (LDA) to provide useful information to a Genetic Algorithm (GA) for an efficient exploration of gene subsets space. LDA is a well-known method of dimension reduction and classification, where the data vectors are transformed into a low-dimensional subspace such that the class centroids are spread out as much as possible. It has been used for several classification problems and recently for Microarray data [8,27,28].

Our approach first extracts a set of interesting genes (about 100 genes) by a filter method in order to limit the search space. Then we use a dedicated GA to determine a small subset of genes that allows a high classification accuracy. Contrary to most previously GAs for gene selection that rely essentially on random genetic operators, we devise a problem specific GA that takes into account useful knowledge of the gene selection and classification problem. Our GA uses a LDA classifier to assess the fitness of a given candidate gene subset and LDA's discriminant coefficients in its crossover and mutation operators.

To evaluate the usefulness of the proposed approach, we carry out extensive experiments on seven public datasets and compare our results with 16 best performing algorithms from the literature. We observe that our approach is able to achieve a high prediction accuracy (from 81% to 100%) with a very small number of informative genes (from 2 to 19). Moreover, our approach enables to propose different subsets of discriminant genes, which may be of a great interest for biological research.

The remainder of this paper is organized as follows. Section 2 recalls the main characteristics of Fisher's LDA and discusses the calculus that must be done in the case of small sample size. Section 3 presents our LDA-based GA for gene selection. Section 4 shows the experimental results and comparisons. Finally conclusions are presented in Section 5.

2 LDA and Small Sample Size Problem

2.1 Linear Discriminant Analysis

LDA is a dimension reduction and classification method, where the data are projected into a low dimension space such that the classes are well separated. As we use this method for binary classification problems, we shall restrict the explanations to this case. We consider a set of n samples belonging to two classes C_1 and C_2, with n_1 samples in C_1 and n_2 samples in C_2. Each sample is described by q variables. So the data form a matrix $X = (x_{ij}), i = 1, \ldots, n; j = 1, \ldots, q$. We denote by μ_k the mean of class C_k and by μ the mean of all the samples:

$$\mu_k = \frac{1}{n_k} \sum_{x_i \in C_k} x_i \text{ and } \mu = \frac{1}{n} \sum_{x_i} x_i = \frac{1}{n} \sum_k n_k \mu_k$$

The data are described by two matrices S_B and S_W, where S_B is the between-class scatter matrix and S_W the within-class scatter matrix defined as follows:

$$S_B = \sum_k n_k(\mu_k - \mu)(\mu_k - \mu)^t \tag{1}$$

$$S_W = \sum_k \sum_{x_i \in C_k} (x_i - \mu_k)(x_i - \mu_k)^t \tag{2}$$

If we denote by S_V the covariance matrix for all the data, we have $S_V = S_B + S_W$.

LDA seeks a linear combination of the initial variables on which the means of the two classes are well separated, measured relatively to the sum of the variances of the data assigned to each class. For this purpose, LDA determines a vector w such that $w^t S_B w$ is maximized while $w^t S_W w$ is minimized. This double objective is realized by the vector w_{opt} that maximizes the criterion:

$$J(w) = \frac{w^t S_B w}{w^t S_W w} \tag{3}$$

One can prove that the solution w_{opt} is the eigen vector associated to the sole eigen value of $S_W^{-1} S_B$, when S_W^{-1} exists. Once this axis w_{opt} is determined, LDA provides a classification procedure (classifier), but in our case we are particularly interested in the *discriminant coefficients* of this vector: the absolute value of these coefficients indicates the importance of the q initial variables for the class discrimination.

2.2 Generalized LDA for Small Sample Size Problems

When the sample size n is smaller than the dimensionality of samples q, S_W is singular. In this case, it is not possible to compute S_W^{-1}. To overcome the singularity problem, recent works have proposed different methods like the null space method [28], orthogonal LDA [26], uncorrelated LDA [27,26] (see also [17] for a comparison of these methods). The two last techniques use the pseudo inverse method to solve the small sample size problem and this is the approach we apply in this work. When S_w is singular, the eigen problem is solved for $S_w^+ S_b$, where S_w^+ is the pseudo inverse of S_w. The pseudo-inverse of a matrix can be computed by Singular Value Decomposition. More specifically, for a matrix A of size $m \times p$ such that $rank(A) = r$, if we denote by $A = U \Sigma V T$ the singular value decomposition of A, where U of size $m \times r$ and V of size $r \times p$ have orthonormal columns, Σ of size $r \times r$, is diagonal with positive diagonal entries, then the pseudo-inverse of A is defined as $A^+ = V \Sigma^{-1} U T$.

2.3 Application to Gene Selection

Microarray data generally contain less than one hundred samples described by at least several thousands of genes. We limit this high dimensionality by a first pre-selection step, where a filter criterion (t-statistic) is applied to determine a subset of relevant genes. In this work, we typically retain 100 genes from which

an intensive exploration is performed using a genetic algorithm to select smaller subsets. In this process, LDA is used as a classification method to evaluate the classification accuracy that can be achieved on a selected gene subset. Moreover the coefficients of the eigen vector calculated by LDA are used to evaluate the importance of each gene for class discrimination.

For a selected gene subset of size p, if $p \leq n$ we rely on the classical LDA (Section 2.1) to obtain the projection vector w_{opt}, otherwise we apply the generalized LDA (Section 2.2) to obtain this vector. We explain in Section 3 how the LDA-based GA reduces progressively the number of selected genes.

3 LDA-Based Genetic Algorithm

In this section we describe our LDA-based Genetic Algorithm (LDA-GA) for gene subset selection. Notice that prior to the LDA-GA search, a filter (t-statistic) is first applied to retain a group G_p of p top ranking genes (typically $p \geq 100$, in this work, $p = 100$). Then, the LDA-based GA is used to conduct a combinatorial search within the space of size 2^p. The purpose of this search is to determine from this large search space small sized gene subsets allowing a high predictive accuracy. In what follows, we present the general procedure and then show the components of the LDA-based Genetic Algorithm. In particular, we explain how LDA is combined with the Genetic Algorithm.

3.1 General GA Procedure

Our LDA-based Genetic Algorithm follows the conventional scheme of a generational GA and uses also an elitism strategy.

- *Initial population*: the initial population is generated randomly in such a way that each chromosome contains a number of genes ranging from $p \times 60\%$ to $p \times 75\%$. The population size is fixed at 100 in this work.
- *Evolution*: the chromosomes of the current population P are sorted according to the fitness function (see Section 3.3). To generate the next population P', |P| new chromosomes are first created using crossover and mutation (see next point). These new chromosomes are then merged with the "best" 10% chromosomes of P to form P' while deleting the worst chromosomes to keep the population size constant.
- *Crossover and mutation*: mating chromosomes are determined from P by considering each pair of adjacent chromosomes (the last one is mated with the first one). By applying our specialized crossover operator (see Section 3.4), one child is created. This child then undergoes a mutation operation (see Section 3.5).
- *Stop condition*: the evolution process ends when a pre-defined number of generations is reached or when one finds a chromosome in the population having a very small gene subset (fixed at 2 genes in this work).

3.2 Chromosome Encoding

Conventionally, a chromosome is used simply to represent a candidate gene subset. Following the idea of [11], a chromosome in our GA encodes more information and is defined by a couple:

$$I = (\tau; \phi)$$

where τ and ϕ have the following meaning. The first part (τ) is a *binary vector* and effectively represents a *candidate gene subset*. Each allele τ_i indicates whether the corresponding gene g_i is selected (τ_i=1) or not (τ_i=0). The second part of the chromosome (ϕ) is a real-valued vector where each ϕ_i corresponds to the *discriminant coefficient* of the eigen vector for gene g_i. As explained in Section 2, the discriminant coefficient defines the contribution of gene g_i to the projection axis w_{opt}. A chromosome can thus be represented as follows:

$$I = (\tau_1, \tau_2, \ldots, \tau_p; \phi_1, \phi_2, \ldots, \phi_p)$$

The length of τ and ϕ is defined by p, the number of the pre-selected genes with a filter (t-statistics) (see beginning of this Section).

Notice that this chromosome encoding is more general and richer than those used in most genetic algorithms for feature selection in the sense that in addition to the candidate gene subset, the chromosome includes other information (LDA discriminant coefficients here) which are useful for designing powerful crossover and mutation operators (see Section 3.4 and 3.5).

3.3 Fitness Evaluation

The purpose of the genetic search in our LDA-GA approach is to seek "good" gene subsets having the minimal size and the highest prediction accuracy. To achieve this double objective, we devise a fitness function taking into account these (somewhat conflicting) criteria.

To evaluate a chromosome I=(τ;ϕ), the fitness function considers the classification accuracy of the chromosome (f_1) and the number of selected genes in the chromosome (f_2). More precisely, f_1 is obtained by evaluating the classification accuracy of the gene subset τ using the LDA classifier on the training dataset and is formally defined as follows[1]:

$$f_1(I) = \frac{TP + TN}{TP + TN + FP + FN} \tag{4}$$

where TP and TN represent respectively the true positive and true negative samples, *i.e.* the correct classifications; FP (FN) is the number of false (true) samples misclassified into the positive (negative) samples.

The second part of the fitness function f_2 is calculated by the formula:

$$f_2(I) = \left(1 - \frac{m_\tau}{p}\right) \tag{5}$$

[1] For the sake of simplicity, we use I (chromosome) instead of τ (gene subset part of I) in the fitness function even if it is the gene subset τ that is effectively evaluated.

where m_τ is the number of bits having the value "1" in the candidate gene subset τ, *i.e.* the number of selected genes; p is the length of the chromosome corresponding to the number of the pre-selected genes from the filter ranking.

Then the fitness function f is defined as the following weighted aggregation:

$$f(I) = \alpha f_1(I) + (1 - \alpha) f_2(I) \text{ subject to } 0 < \alpha < 1$$

where α is a weighted parameter that allows us to allocate a relative importance factor to f_1 or f_2. Assigning to α a value greater than 0.5 will push the genetic search toward solutions of high classification accuracy (probably at the expense of having more selected genes). Inversely, using small values of α helps the search go toward small sized gene subsets. So variations of α will change the search direction of the genetic algorithm.

3.4 LDA-Based Crossover

It is now widely acknowledged that, whenever it is possible, genetic operators such as crossover and mutation should be tailored to the target problem. In other words, in order for genetic operators to fully play their role, it is preferable to integrate problem-specific knowledge into these operators. In our case, we use the discriminant coefficients from the LDA classifier to design our crossover and mutation operators. Here, we explain how our LDA-based crossover operates (denoted by LDA-X hereafter).

LDA-X combines two parent chromosomes I^1 and I^2 to generate a new chromosome I^c in such a way that 1) top ranking genes in both parents are conserved in the child and 2) the number of selected genes in the child I^c is not greater than the number of selected genes in the parents. The first point ensures that "good" genes are transmitted from one generation to another while the second property is coherent with the optimization objective of small-sized gene subsets.

More formally, let $I^1=(\tau^1;\phi^1)$ and $I^2=(\tau^2;\phi^2)$ be two parent chromosomes, $I^c=(\tau^c;\phi^c)$ the child which will be generated by crossover, $\kappa \in [0,1)$ a parameter indicating the percentage of genes that will not be transmitted from the parents to the child. Then our LDA-X crossover performs the following steps to generate I^c, the child chromosome.

1. According to κ determine the number of genes of I^1 and I^2 (more precisely, τ^1 and τ^2) that will be discarded, denote them by n_1 and n_2;
2. Remove respectively from τ^1 and τ^2, the n_1 and n_2 least ranking genes according to the LDA discriminant coefficients;
3. Merge the modified τ^1 and τ^2 by the logic AND operator to generate τ^c;
4. Apply the LDA classifier to τ^c, fill ϕ^c by the resulting LDA discriminant coefficients;
5. If needed, remove the least discriminative genes from τ^c until τ^c contains no more genes than I^1 or I^2 does; update ϕ^c accordingly;
6. Create the child $I^c=(\tau^c;\phi^c)$.

Before inserting the child into the next population, I^c undergoes a mutation operation.

3.5 LDA-Based Mutations

In a conventional GA, the purpose of mutation is to introduce new genetic materials for diversifying the population by making local changes in a given chromosome. For binary coded GAs, this is typically realized by flipping the value of some bits ($1 \rightarrow 0$ or $0 \rightarrow 1$). In our case, mutation is used for dimension reduction; each application of mutation eliminates a single gene ($1 \rightarrow 0$). To determine which gene is discarded, two criteria are used, leading to two mutation operators.

- *Mutation using discriminant coefficient* (M1): Given a chromosome $I=(\tau; \phi)$, we identify the smallest LDA discriminant coefficient in ϕ and remove the corresponding gene (this is the least informative genes among the current candidate gene subset τ).
- *Mutation by discriminant coefficient and frequency* (M2): This mutation operator relies on a frequency information of each selected gene. More precisely, a frequency counter is used to count the number of times a selected gene is classified (according to the LDA classifier) as the least informative gene within a gene subset. Based on this information, we remove the gene that has the highest counter, in other words, the gene that is frequently considered as a poor predictor by the classifier.

4 Datasets and Experimental Setup

4.1 Microarray Gene Expression Datasets

To assess the performance of our LDA-based genetic algorithm, we performed our experiments on seven well-known public datasets, namely Leukemia, Colon cancer, DLBCL, CNS embryonal tumor, Lung, Prostate and Ovarian cancer. A summary of the datasets is provided in Table 1.

4.2 Experimental Settings

For our experimentations, we used the following experimental settings. Each initial dataset is split into a training set and a test set according to the literature. LDA-GA is applied on the training set in order to select relevant gene subsets.

Table 1. Summary of datasets used for experimentation

Dataset	Genes	Samples	References
Leukemia	7129	72	Golub et al [9]
Colon	2000	62	Alon et al [2]
Lung	12533	181	Gordon et al [10]
Prostate	12600	109	Singh et al [21]
CNS	7129	60	Pomeroy et al [20]
Ovarian	15154	253	Petricoin et al [19]
DLBCL	4026	47	Alizadeh et al [1]

Because our fitness function relies on two criteria (3.3), we carry out two types of experiments. In the first one, named Exp1 hereafter, we select the gene subset according to the second criterion trying to minimize the number of selected genes. In the second type of experiments, named Exp2, we focus on the accuracy achieved by the different solutions obtained by LDA-GA and we retain the gene subsets that provide the best accuracy. Because of the stochastic nature of our LDA-GA algorithm, we run 10 executions of the GA and we retain the best solution found during these 10 executions.

In both experiments, the final predictive accuracy of a selected gene subset is estimated by the LDA-classifier built on the gene subset obtained by the training step. As the data contain few samples, we use a 10-fold cross validation on the whole dataset to obtain a reliable estimation of the classification accuracy.

We have explained in Section 3 that our LDA-GA can apply two kinds of mutation (M1 and M2). That is why we report in the following subsection four results for each dataset: GA-M1/Exp1 is our GA with M1 mutation and we select the gene subset according to the conditions of Exp1 (focusing on the number of genes); GA-M1/Exp2 is the GA with M1 mutation and we select the gene subset according to the conditions of Exp2 (the best accuracy). Similarly, two results are reported with M2 mutation (named GA-M2/Exp1, GA-M2/Exp2).

4.3 Results and Comparisons

In this section, we propose a comparison of our LDA-GA with some state-of-the-art methods for gene selection and classification. A reliable comparison between two methods is only possible if they use the same experimental conditions. For this reason, we select 16 recent methods (since 2004) that seem to fulfill this condition.

We show in Table 2 the best results (in bold) obtained by these methods and by our LDA-based GA approach on the seven datasets presented previously. An entry with the symbol (-) in this table means that the paper does not treat the corresponding dataset. All the methods reported in this table use a process of cross validation, notice however that in some cases, the papers do not explain precisely how the experimentation is conducted.

From the results of Table 2, one observes that the proposed approach (last four lines) gives very competitive results compared with these reference methods. Indeed, our LDA-based GA achieves globally very high predictive accuracy (from 81.6% to 100%) with a very small number of selected genes (from 2 to 19).

The most remarkable results for our approach concern the DLBCL dataset. We obtain a perfect prediction with only 4 genes while the previously methods reach a prediction rate no greater than 98% with at least 20 genes. For the Ovarian cancer dataset, the LDA-GA gives a prediction accuracy of 98.4% with a subset of only 4 genes. The reference algorithms have a slightly better classification rate, but select much more genes (20, 26, 75). Notice that a perfect rate is reported in [15] with 50 genes. However the dataset used in [15] (30 cancerous and 24 normal samples, 1536 genes) is different from the Ovarian cancer dataset described in Table 1 (91 normal and 162 cancerous samples, 15154 genes).

Table 2. Results of our LDA-based GA (four last lines) compared to the most relevant works on cancer classification. The figures give the classification accuracy and in brackets, the number of genes when this is available.

Authors	Leukemia	Colon	Lung	Prostate	CNS	Ovarian	DLBCL
Ye et al [27]	97.5	85.0	–	92.5	–	–	–
Liu et al [14]	**100**(30)	91.9(30)	**100**(30)	97.0(30)	–	99.2(75)	98(30)
Tan & Gilbert [22]	91.1	95.1	93.2	73.5	88.3	–	–
Ding & Peng [7]	**100**	93.5	97.2	–	–	–	–
Cho & Won [6]	95.9 (25)	87.7(25)	–	–	–	–	93.0(25)
Yang et al [25]	73.2	84.8	–	86.88	–	–	–
Peng et al [18]	98.6 (5)	87.0(4)	**100**(3)	–	–	–	–
Wang et al [24]	95.8 (20)	**100**(20)	–	–	–	–	95.6(20)
Huerta et al [4]	**100**	91.4	–	–	–	–	–
Pang et al [16]	94.1(35)	83.8(23)	91.2(34)	–	65.0(46)	98.8(26)	–
Li et al [12]	97.1(20)	83.5(20)	–	91.7(20)	68.5(20)	99.9(20)	93.0(20)
Zhang et al [29]	**100**(30)	90.3(30)	**100**(30)	95.2(30)	80(30)	–	92.2(30)
Yue et al [28]	83.8(100)	85.4(100)	–	–	–	–	–
Hernandez et al [11]	91.5(3)	84.6(7)	–	–	–	–	–
Li et al [13]	**100**(4)	93.6(15)	–	–	–	–	–
Wang et al [23]	**100**(375)	93.5(35)	–	–	–	–	–
GA-M1/Exp1	97.2(**2**)	90.3(**2**)	97.7(**2**)	94.1(**2**)	78.3(**4**)	96.0(**2**)	91.4(**2**)
GA-M2/Exp1	97.2(**2**)	91.9(3)	98.3(2)	94.1(**2**)	85.0(**4**)	96.4(**2**)	93.6(**2**)
GA-M1/Exp2	98.6(5)	91.9(3)	97.7(**2**)	94.8(6)	81.6(8)	98.4(4)	**100**(8)
GA-M2/Exp2	**100**(5)	93.5(9)	98.3(**2**)	95.5(18)	86.6(7)	98.8(19)	**100**(4)

Finally, notice that the LDA classifier used in this paper is not the most powerful classifier. Effectively, in another experimentation, we also used a linear SVM classifier to estimate the predictive accuracy of the gene subsets selected by the LDA-GA, leading to slightly better results.

4.4 Discussion

We now discuss about two important issues of the LDA-GA approach: possible influences of the pre-selection on the prediction accuracy and the capacity of the approach to explore large sets of genes.

The search space of our LDA-GA is delimited by a first step which pre-selects a limited number (100 in this paper) of genes with the t-statistic filter criterion. One may wonder whether changing the filtering criterion and the number of selected genes affects the performance of the approach. In [5], an exhaustive study is presented concerning the influence of data pre-processing and filtering criteria on the classification performance. Three filtering criteria, BSS/WSS, t-statistic and Wilcoxon test were compared, and the results did not show any clear dominance of one criterion with respect to the others. However, the fuzzy pre-processing for date normalization and redundancy reduction presented in [5] does show a positive influence on the classification performance whatever the filtering criterion that is applied after.

The main interest of this genetic approach for gene selection is its ability to propose a combinatorial exploration of gene subsets. Clearly, this is not the case in classical approaches like backward selection. In recursive feature elimination for example, once a gene is discarded by the selection process, it is definitively ignored in the further steps even if its association to other genes can improve the classification result. Consider the Leukemia dataset, a perfect performance of 100% is reached with 5 genes (Table 2). LDA-GA also finds other gene subsets (with 5 to 10 genes) achieving a perfect cross-validation classification. More precisely, one of these subsets contains the genes placed in positions 3, 12, 63, 72, and 81 by the filter ranking criterion. Another gene subset that achieves a perfect classification contains the genes ranked in positions: 1, 2, 19, 72 and 81. Generally filter methods retain a small number of genes for classification (typically 30). Our observation shows that it is interesting and useful to explore a large set of genes because relevant subsets can contain genes that are not in the 30 top-ranking ones. Moreover, the possibility to examine diverse solutions constitutes a valuable feature for further biological investigations.

5 Conclusions

In this paper, we have introduced a new wrapper approach for selecting small gene subsets able to lead to high prediction accuracy. Our approach begins with a t-statistic filter that pre-selects a first set of genes (100 in this paper). To further reduce the gene dimension, we use a hybrid Genetic Algorithm to explore the gene subset space. The hybrid GA includes some original features that make it highly effective for identifying small sized and informative gene subsets. In particular, it uses Fisher's Linear Discriminant Analysis as its fitness function to assess the quality of each candidate gene subset. Moreover, useful discriminant information provided by the LDA classifier is directly integrated into its crossover and mutation operators. Indeed, the discriminant coefficients of LDA's eigen vector constitute a valuable indicator for recombining gene subsets (crossover) and for gene dimension reduction (mutation). The bi-criteria fitness function provides a very flexible way for the LDA-GA to explore the gene subset space either for the minimization of the selected genes or for the maximization of the prediction accuracy.

We have evaluated extensively our LDA-based GA approach on seven public datasets (Leukemia, Colon, DLBCL, Lung, Prostate, CNS and Ovarian) using a 10-fold cross validation process. A large comparison was carried out with 16 state-of-art algorithms that are based on a variety of methods. The results show clearly the interest of the LDA-GA approach for finding small sized informative gene subsets leading to high prediction accuracy. For all the datasets, our approach is able to select gene subsets of the smallest size while ensuring the best or the second best classification rate. For one dataset (DLBCL), we obtain the best result ever found with a perfect prediction with only 4 informative genes.

Finally, the proposed approach has another practically useful feature for biological analysis. In fact, instead of producing a single solution (gene subset),

our approach can easily and naturally provide multiple non-dominated solutions that constitute valuable candidates for further biological investigations.

Acknowledgments. This work is partially supported by the French Ouest Genopole and the Bioinformatics Program of the Region "Pays de La Loire". The first author of the paper is supported by a CoSNET scholarship. We would like to thank the referees for their helpful questions and suggestions.

References

1. Alizadeh, A., Eisen, B.M., Davis, R.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., et al.: Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. Proc. Nat. Acad. Sci. USA. 96, 6745–6750 (1999)
3. Ben-Dor, A., Bruhn, L., Friedman, N., Nachman, I., Schummer, M., Yakhini, Z.: Tissue classification with gene expression profiles. Journal of Computational Biology 7(3-4), 559–583 (2000)
4. Bonilla Huerta, E., Duval, B., Hao, J.K.: A hybrid ga/svm approach for gene selection and classification of microarray data. In: Rothlauf, F., Branke, J., Cagnoni, S., Costa, E., Cotta, C., Drechsler, R., Lutton, E., Machado, P., Moore, J.H., Romero, J., Smith, G.D., Squillero, G., Takagi, H. (eds.) EvoWorkshops 2006. LNCS, vol. 3907, pp. 34–44. Springer, Heidelberg (2006)
5. Bonilla Huerta, E., Duval, B., Hao, J.K.: Fuzzy logic for elimination of redundant information of microarray data. In: Genomics, Proteomics and Bioinformatics (June 2008) (to appear)
6. Cho, S.-B., Won, H.-H.: Cancer classification using ensemble of neural networks with multiple significant gene subsets. Applied Intelligence 26(3), 243–250 (2007)
7. Ding, C., Peng, H.: Minimum redundancy feature selection from microarray gene expression data. J. Bioinformatics and Computational Biology 3(2), 185–206 (2005)
8. Dudoit, S., Fridlyand, J., Speed, T.: Comparison of discrimination methods for the classification of tumors using gene expression data. Journal of the American Statistical Association 97, 77–87 (2002)
9. Golub, T., Slonim, D., Tamayo, P., et al.: Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. Science 286, 531–537 (1999)
10. Gordon, G.J., et al.: Translation of microarray data into clinically relevant cancer diagnostic tests using gene expression ratios in lung cancer and mesothelioma. Cancer Res. 62, 4963 (2002)
11. Hernandez Hernandez, J.C., Duval, B., Hao, J.K.: A genetic embedded approach for gene selection and 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)
12. Li, G.-Z., Zeng, X.-Q., Yang, J.Y., Yang, M.Q.: Partial least squares based dimension reduction with gene selection for tumor classification. In: Proc. of 7th IEEE Intl. Symposium on Bioinformatics and Bioengineering, pp. 1439–1444 (2007)
13. Li, S., Wu, X., Hu, X.: Gene selection using genetic algorithm and support vectors machines. Soft Comput. 12(7), 693–698 (2008)

14. Liu, B., Cui, Q., Jiang, T., Ma, S.: A combinational feature selection and ensemble neural network method for classification of gene expression data. BMC Bioinformatics 5(136), 1–12 (2004)
15. Marchiori, E., Sebag, M.: Bayesian learning with local support vector machines for cancer classification with gene expression data. In: Rothlauf, F., Branke, J., Cagnoni, S., Corne, D.W., Drechsler, R., Jin, Y., Machado, P., Marchiori, E., Romero, J., Smith, G.D., Squillero, G. (eds.) EvoWorkshops 2005. LNCS, vol. 3449, pp. 74–83. Springer, Heidelberg (2005)
16. Pang, S., Havukkala, I., Hu, Y., Kasabov, N.: Classification consistency analysis for bootstrapping gene selection. Neural Computing and Appli. 16, 527–539 (2007)
17. Park, H., Park, C.: A comparison of generalized linear discriminant analysis algorithms. Pattern Recognition 41(3), 1083–1097 (2008)
18. Peng, Y., Li, W., Liu, Y.: A hybrid approach for biomarker discovery from microarray gene expression data. Cancer Informatics, 301–311 (2006)
19. Petricoin, E.F., Ardekani, A.M., Hitt, B., Levine, P., Steinberg, S., Mills, G., Simone, C., Fishman, D., Kohn, E., Liotta, L.A.: Use of proteomic patterns in serum to identify ovarian cancer. Lancet 359, 572–577 (2002)
20. Pomeroy, S.L., Tamayo, P., Gaasenbeek, M., Sturla, L.M., Golub, T.R.: Prediction of central nervous system embryonal tumour outcome based on gene expression. Nature 415(6870), 436–442 (2002)
21. Singh, D., Febbo, P.B., Ross, K., Jackson, D.G., Manola, J., Ladd, C., Tamayo, P., Renshaw, A.A., Amico, A.V., Richie, J.P., et al.: Gene expression correlates of clinical prostate cancer behavior. Cancer Cell 1, 203–209 (2002)
22. Tan, A.C., Gilbert, D.: Ensemble machine learning on gene expression data for cancer classification. Appl. Bioinformatics 2(3 Suppl), 75–83 (2003)
23. Wang, S., Chen, H., Li, S., Zhang, D.: Feature extraction from tumor gene expression profiles using DCT and DFT. In: Neves, J., Santos, M.F., Machado, J.M. (eds.) EPIA 2007. LNCS (LNAI), vol. 4874, pp. 485–496. Springer, Heidelberg (2007)
24. Wang, Z., Palade, V., Xu, Y.: Neuro-fuzzy ensemble approach for microarray cancer gene expression data analysis. In: Proc. Evolving Fuzzy Systems, pp. 241–246 (2006)
25. Yang, W.-H., Dai, D.-Q., Yan, H.: Generalized discriminant analysis for tumor classification with gene expression data. In: Machine Learning and Cybernetics, pp. 4322–4327 (2006)
26. Ye, J.: Characterization of a family of algorithms for generalized discriminant analysis on undersampled problems. Journal of Machine Learning Research 6, 483–502 (2005)
27. Ye, J., Li, T., Xiong, T., Janardan, R.: Using uncorrelated discriminant analysis for tissue classification with gene expression data. IEEE/ACM Trans. Comput. Biology Bioinform. 1(4), 181–190 (2004)
28. Yue, F., Wang, K., Zuo, W.: Informative gene selection and tumor classification by null space LDA for microarray data. In: Chen, B., Paterson, M., Zhang, G. (eds.) ESCAPE 2007. LNCS, vol. 4614, pp. 435–446. Springer, Heidelberg (2007)
29. Zhang, L., Li, Z., Chen, H.: An effective gene selection method based on relevance analysis and discernibility matrix. In: Zhou, Z.-H., Li, H., Yang, Q. (eds.) PAKDD 2007. LNCS (LNAI), vol. 4426, pp. 1088–1095. Springer, Heidelberg (2007)

Sequential Forward Selection Approach to the Non-unique Oligonucleotide Probe Selection Problem

Lili Wang[1], Alioune Ngom[1,⋆], and Luis Rueda[2,⋆⋆]

[1] School of Computer Science, 5115 Lambton Tower
University of Windsor, 401 Sunset Avenue
Windsor, Ontario, N9B 3P4, Canada
{wang111v,angom}@uwindsor.ca
[2] Department of Computer Science,
University of Concepción. Edmundo Larenas 215,
Concepción, VIII Region, Chile
lrueda@udec.cl

Abstract. In order to accurately measure the gene expression levels in microarray experiments, it is crucial to design *unique*, highly specific and highly sensitive oligonucleotide probes for the identification of biological agents such as genes in a sample. Unique probes are difficult to obtain for closely related genes such as the known strains of HIV genes. The *non-unique* probe selection problem is to find one of the smallest probe set that is able to uniquely identify targets in a biological sample. This is an NP-hard problem. We present heuristic for finding near-minimal non-unique probe sets. Our method is a variant of the *sequential forward selection* algorithm, which used for feature subset selection in pattern recognition systems. The heuristic is guided by a probe set selection criterion which evaluates the efficiency and the effectiveness of a probe set in classifying targets genes as present or absent in a biological sample. Our methods outperformed all currently published greedy algorithms for this problem.

Keywords: Probe Selection, Gene Expression.

1 Introduction

Oligonucleotide microarrays are widely used tools, in molecular biology, providing a fast and cost-effective method for monitoring the expression of thousands of genes simultaneously [7]. In order to measure the expression level of a specific gene in a sample, one must design a microarray containing short strands of known DNA sequences of 8 to 30 bp, called *oligonucleotide probes*, which are

⋆ Research partially funded by Canadian NSERC Grant #RGPIN228117-2006 and CFI grant #9263.
⋆⋆ Research partially funded by Chilean FONDECYT Grant #1060904.

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 262–275, 2008.

complementary to the gene's segments, called *targets*. These targets, if present in the sample, should bind to their complementary probes by means of *hybridization*. The success of a microarray experiment depends on how well each probe hybridizes to its target under specified experimental conditions such as temperature and salt concentration. However, choosing good probes is a difficult task since different sequences have different hybridization characteristics.

A probe is *unique*, if it is designed to hybridize to a single target. However, due to hybridization errors, there is no guarantee that unique probes will hybridize to their intended targets only. Many parameters such as secondary structure, salt concentration, GC content, free energy and melting temperature also affect the hybridization quality of probes [7], and their values must be carefully determined to design high quality probes. It is particularly difficult to design unique probes for closely related genes, given the probe length and melting temperature constraints. An alternative approach is to devise a method that can make use of *non-unique* probes, i.e. probes that are designed to hybridize to at least one target [7]. Also, a smaller probe set can be used with non-unique probes than can be with unique probes. Minimizing the number of probes in a microarray experiment is also a reasonable objective, since it is proportional to the cost of the experiment. The *non-unique probe selection problem* is to determine a smallest set of probes able to identify all targets present in a biological sample. This is an NP-hard problem [1], for which several approaches have been proposed recently [2] [6] [7] [8] [9].

Schliep *et al.* [7] first introduced the non-unique probe selection problem and described a simple but fast greedy heuristic, which computes an approximate solution that guarantees $s_{\min}$-separation for pairs of small target groups. Klau *et al.* [1] proposed two ILP formulations for this problem, respectively for single targets and for target groups, then solved it using the ILP solver CPLEX on pre-reduced problem instances. They also proved that the non-unique probe selection problem is NP-hard. Meneses *et al.* [2] proposed a deterministic greedy heuristic, for single targets only, which first constructs an initial feasible solution through local search, and then applies a reduction method to further reduce this solution. Ragle *et al.* [6] developed an *optimal cutting-plane* ILP heuristic, for single targets only, to find optimal solutions within practical computational limits. Wang *et al.* [8] proposed deterministic greedy heuristics that select probes based on their ability to help satisfy the constraints. Recently, Wang *et al.* [9] combined the probe selection functions with evolutionary methods and produced results that are at least comparable to those obtained by the method of [6], which is the best published approach for this problem.

2 Non-unique Probe Selection Problem

Given a target set, $T = \{t_1, \ldots, t_m\}$, and probe set, $P = \{p_1, \ldots, p_n\}$, an $m \times n$ *target-probe incidence matrix* $H = [h_{ij}]$ is such that $h_{ij} = 1$, if probe p_j hybridizes to target t_i, and $h_{ij} = 0$ otherwise. Table 1 shows an example of a matrix with $m = 4$ targets and $n = 6$ probes. A probe p_j *separates* two targets,

Table 1. A 4×6 target-probe incidence matrix

	p_1	p_2	p_3	p_4	p_5	p_6
t_1	1	1	0	1	0	1
t_2	1	0	1	0	0	1
t_3	0	1	1	1	1	1
t_4	0	0	1	1	1	0

t_i and t_k, if it is a substring of either t_i or t_k, that is, if $|h_{ij} - h_{kj}| = 1$. For example, if t_i = AGGCAATT and t_k = CCATATTGG, then probe p_j = GCAA separates t_i and t_k, since it is a substring of t_i only, whereas probe p_l = ATT does not separate t_i and t_k, since it is a substring of both targets [2]. Two targets, t_i and t_k, are *s-separated*, $s \geq 1$, if there exist at least s probes such that each separates t_i and t_k; in other words, the Hamming distance between rows i and k in H is at least s. For example, in Table 1 targets t_2 and t_4 are 4-separated. A target t is *c-covered*, $c \geq 1$, if there exist at least c probes such that each hybridizes to t. In Table 1, target t_2 is 3-covered. Due to hybridization errors in microarray experiments, it is required that any two targets be $s_{\min}$-separated and any target be $c_{\min}$-covered; usually, we have $s_{\min} \geq 2$ and $c_{\min} \geq 2$. These two requirements are called *separation constraints* and *coverage constraints*.

Given a matrix H, the aim of the non-unique probe selection problem is to find a minimal probe set that determines the presence or absence of specified targets, and such that all constraints are satisfied. In Table 1, if $s_{\min} = c_{\min} = 1$ and assuming that exactly one of $t_1, \ldots, t_4$ is in the sample, then the goal is to select a minimal set of probes that allows us to infer the presence or absence of a single target. In this case, a minimal solution is $\{p_1, p_2, p_3\}$ since for target t_1, probes p_1 and p_2 hybridize while p_3 does not; for target t_2, probes p_1 and p_3 hybridize while p_2 does not; for target t_3, probes p_2 and p_3 hybridize while p_1 does not; and finally for target t_4, only probe p_3 hybridize. Thus, each single target will be identified by the set $\{p_1, p_2, p_3\}$, if it is the only target present in the sample; moreover, all constraints are satisfied. For $s_{\min} = c_{\min} = 2$, a minimal solution that satisfies all constraints is $\{p_2, p_3, p_5, p_6\}$. Of course, $\{p_1, \ldots, p_6\}$ is a solution but it is not minimal, and hence is not cost-effective.

Stated formally, given an $m \times n$ matrix H with a target set $T = \{t_1, \ldots, t_m\}$ and a probe set $P = \{p_1, \ldots, p_n\}$, and a minimum coverage parameter $c_{\min}$, a minimum separation parameter $s_{\min}$ and a parameter $d_{\max} \geq 1$, the aim of the non-unique probe selection problem is to determine a subset $P_{\min} = \{q_1, q_2, \cdots, q_s\} \subseteq P$ such that:

1. $s = |P_{\min}| \leq n$ is minimal.
2. Each target $t_i \in T$ is $c_{\min}$-covered by some probes in $P_{\min}$.
3. Each target-pair $(t_i, t_k) \in T \times T$ is $s_{\min}$-separated by some probes in $P_{\min}$.
4. Each pair of small groups of targets is $s_{\min}$-separated by some probes in $P_{\min}$.

This problem was proved to be NP-hard, in [1], by performing a reduction from the *set covering* problem. It is NP-hard even for $c_{\min} = 1$ or $s_{\min} = 1$. The work of [1] formulated the non-unique probe selection problem as an *integer linear programming* (ILP) problem. Let $C = \{(i,k) \mid 1 \leq i < k \leq m\}$ be the set of all combinations of target indices. Assign $x_j = 1$ if probe p_j is chosen and 0 otherwise. We have:

$$\text{Minimize: } \sum_{j=1}^{n} x_j \ . \tag{1}$$

Subject to:

$$x_j \in \{0,1\} \qquad 1 \leq j \leq n \ , \tag{2}$$

$$\sum_{j=1}^{n} h_{ij} x_j \geq c_{\min} \qquad 1 \leq i \leq m \ , \tag{3}$$

$$\sum_{j=1}^{n} |h_{ij} - h_{kj}| \, x_j \geq s_{\min} \qquad 1 \leq i < k \leq m \ . \tag{4}$$

Function (1) minimizes the number of probes. The probe selection variables are binary-valued in Restriction (2). Constraints (3) and (4) are the coverage and separation constraints, respectively. Note that Constraints (4) are for single targets only. As opposed to this, in [1], another ILP formulation was proposed, which includes the separation constraints for small groups of targets. In this paper, we solve the ILP formulation, above, using a deterministic greedy heuristic based on a feature subset selection method used in pattern recognition. Note that one can easily check if the probes in the original set of candidate satisfy all the constraints. If not, then there are no feasible solutions. In this case, we can insert *unique virtual probes* in the original probe set only for those targets or target-pairs that are not $c_{\min}$-covered or $s_{\min}$-separated. This will ensure the existence of feasible solutions.

3 Probe Selection Functions

We want to select a minimum number of probes such that each target is $c_{\min}$-covered and each target-pair is $s_{\min}$-separated. Consider a target-probe incidence matrix, H, the parameters $c_{\min}$ and $s_{\min}$, the initial feasible candidate set of probes, $P = \{p_1, \ldots, p_n\}$, and the set of targets $T = \{t_1, \ldots, t_m\}$. Let P_{t_i} be the set of probes hybridizing to target t_i, and $P_{t_{ik}}$ be the set of probes separating the target-pair t_{ik}. A probe $p \in P_{t_i}$ is an *essential covering probe* if and only if $|P_{t_i}| = c_{\min}$. In Table 1, for instance, the probes in $P_{t_2} = \{p_1, p_3, p_6\}$ are essential covering probes if $c_{\min} = 3$. *Essential separating probes* are defined similarly. Essential probes must be contained in any minimal solution; that is, removing any such probe will make the solution unfeasible. A *redundant probe* is the one for which a feasible solution remains feasible when the probe is removed. Note that a probe may be redundant for some candidate solutions but non-redundant for others. There is a degree of redundancy between probes such

that highly redundant probes are in very few or no minimal solutions. Our approach associates with each probe and each probe set a *degree of contribution* to minimal solutions (or, *degree of non-redundancy*)[8]. This degree corresponds to the ability of a probe, or a probe set, to help satisfy *all* the constraint.

3.1 Coverage Function

We want to choose the minimum number of probes such that each target is $c_{\min}$-covered. Given H, the parameter $c_{\min}$, the probe set $P = \{p_1, \ldots, p_n\}$ and the target set $T = \{t_1, \ldots, t_m\}$, we defined the function $\text{cov}_{\text{drc}} : P \times T \mapsto [0, 1]$ in [8] as follows:

$$\text{cov}_{\text{drc}}(p_j, t_i) = h_{ij} \times \frac{c_{\min}}{|P_{t_i}|}, \quad p_j \in P_{t_i}, \quad t_i \in T \ , \tag{5}$$

where, P_{t_i} is the set of probes hybridizing to target t_i; $\text{cov}_{\text{drc}}(p_j, t_i)$ is the amount that p_j contributes to satisfy the coverage constraint for target t_i. For target t_i, p_j is likely to be redundant for a larger value of $|P_{t_i}|$ and likely to be non-redundant for a smaller value of $|P_{t_i}|$. We defined the *coverage function* $C_{\text{drc}} : P \mapsto [0, 1]$ in [8] as follows:

$$C_{\text{drc}}(p_j) = \max_{t_i \in T_{p_j}} \{\text{cov}_{\text{drc}}(p_j, t_i) \quad | \quad 1 \leq j \leq n\} \ , \tag{6}$$

where T_{p_j} is the set of targets covered by p_j. $C_{\text{drc}}(p_j)$ is the maximum amount that p_j can contribute to satisfy the minimum coverage constraints. Table 2 shows the coverage function table produced from Table 1.

Function C_{drc} favors the selection of probes that $c_{\min}$-cover targets t_i that have the smallest subsets P_{t_i}; these are the essential or near-essential covering probes. In Table 2, for example, target t_2 has the minimal value $|P_{t_2}| = 3$, and hence any probe that covers it can be selected first. In particular, function C_{drc} guarantees the selection of near-essential covering probes that $c_{\min}$-cover *dominated targets*; t_i *dominates* t_k if $P_{t_k} \subset P_{t_i}$. In Table 2, for example, t_3 dominates t_4 since $P_{t_4} = \{p_3, p_4, p_5\} \subset \{p_2, p_3, p_4, p_5, p_6\} = P_{t_3}$. Any $c_{\min}$-cover of the dominated target t_k will also $c_{\min}$-cover all its dominant targets, and

Table 2. Coverage function table obtained from Table 1

	p_1	p_2	p_3	p_4	p_5	p_6
t_1	$\frac{c_{\min}}{4}$	$\frac{c_{\min}}{4}$	0	$\frac{c_{\min}}{4}$	0	$\frac{c_{\min}}{4}$
t_2	$\frac{c_{\min}}{3}$	0	$\frac{c_{\min}}{3}$	0	0	$\frac{c_{\min}}{3}$
t_3	0	$\frac{c_{\min}}{5}$	$\frac{c_{\min}}{5}$	$\frac{c_{\min}}{5}$	$\frac{c_{\min}}{5}$	$\frac{c_{\min}}{5}$
t_4	0	0	$\frac{c_{\min}}{3}$	$\frac{c_{\min}}{3}$	$\frac{c_{\min}}{3}$	0
C_{drc}	$\frac{c_{\min}}{3}$	$\frac{c_{\min}}{4}$	$\frac{c_{\min}}{3}$	$\frac{c_{\min}}{3}$	$\frac{c_{\min}}{3}$	$\frac{c_{\min}}{3}$
C_{dps}	$\frac{c_{\min}}{6}$	$\frac{c_{\min}}{8}$	$\frac{c_{\min}}{4}$	$\frac{c_{\min}}{4}$	$\frac{c_{\min}}{6}$	$\frac{c_{\min}}{4}$

therefore, more targets are $c_{\min}$-covered. Probes covering the dominated target t_k have larger cov_{drc} values than probes covering its dominant targets t_i, since $|P_{t_k}| < |P_{t_i}|$, and hence they will be selected first.

We would also like to favor the selection of *dominant probes*; p_j *dominates* p_l if $T_{p_l} \subset T_{p_j}$. In Table 2, for instance, p_6 dominates p_1 since $T_{p_1} = \{t_1, t_2\} \subset \{t_1, t_2, t_3\} = T_{p_6}$. Selecting dominant probes instead of dominated probes covers more targets. In the example, however, we have $C_{\text{drc}}(p_1) = C_{\text{drc}}(p_6)$, and hence p_1 could be selected for target coverage rather than p_6, depending on a particular order of the probes. On the other hand, p_6 dominates p_2 and $C_{\text{drc}}(p_6) > C_{\text{drc}}(p_2)$, and hence p_6 will be selected first. To favor the selection of a dominant probe among dominated probes equal in value C_{drc}, we penalize each probe p by an amount proportional to $|T_p|$, as follows:

$$C_{\text{dps}}(p_j) = C_{\text{drc}}(p_j) \times \frac{1}{m - |T_{p_j}| + 1} \quad , \tag{7}$$

and probes that cover fewer targets are penalized more than probes that cover more targets. Table 2 shows the values of C_{dps} for each probe.

3.2 Separation Function

We want to choose the minimum number of probes such that each target-pair is $s_{\min}$-separated. We defined the function $\text{sep}_{\text{drc}} : P \times T^2 \mapsto [0, 1]$ in [8] as follows:

$$\text{sep}_{\text{drc}}(p_j, t_{ik}) = |h_{ij} - h_{kj}| \times \frac{s_{\min}}{|P_{t_{ik}}|}, \quad p_j \in P_{t_{ik}}, \quad t_{ik} \in T^2 \quad , \tag{8}$$

where, $P_{t_{ik}}$ is the set of probes separating target-pair t_{ik}; $\text{sep}_{\text{drc}}(p_j, t_{ik})$ is what p_j can contribute to satisfy the separation constraint for target-pair t_{ik}. We defined the *separation function* $S_{\text{drc}} : P \mapsto [0, 1]$ in [8] as follows:

$$S_{\text{drc}}(p_j) = \max_{t_{ik} \in T^2_{p_j}} \{\text{sep}_{\text{drc}}(p_j, t_{ik}) \quad | \quad 1 \le j \le n\} \quad , \tag{9}$$

where $T^2_{p_j}$ is the set of target-pairs separated by p_j. $S_{\text{drc}}(p_j)$ is the maximum amount that p_j can contribute to satisfy the minimum separation constraints. The full separation function table can be found in [9].

Function S_{drc} also favors the selection of probes that $s_{\min}$-separate target-pairs t_{ik} which have the smallest subsets $P_{t_{ik}}$ and further favors the selection of near-essential separating probes that $s_{\min}$-separate *dominated target pairs*. To favor the selection of a dominant probe that has the same value, S_{drc}, as some of its dominated probes, we penalize each probe p by an amount proportional to $|T^2_p|$, as follows:

$$S_{\text{dps}}(p_j) = S_{\text{drc}}(p_j) \times \frac{1}{\frac{m(m-1)}{2} - |T^2_{p_j}| + 1} \quad , \tag{10}$$

and probes that separate fewer target-pairs are penalized more than probes that separate more target-pairs.

3.3 Selection Function

We want to select the minimum number of probes such that all coverage and separation constraints are satisfied; that is, we must select a probe according to its ability to help satisfy both coverage *and* separation constraints. In [8], we combined functions C_{drc} and S_{drc} into a single probe selection function, $D_{\text{drc}} : P \mapsto [0, 1]$ as follows:

$$D_{\text{drc}}(p_j) = \max\{(C_{\text{drc}}(p_j), S_{\text{drc}}(p_j)) \mid 1 \leq j \leq n\} \ . \tag{11}$$

$D_{\text{drc}}(p_j)$ is the degree of contribution of p_j, that is, the maximum amount required for p_j to satisfy all constraints. D_{drc} ensures that all essential probes p_j will be selected for inclusion in the subsequent candidate solution, since $C_{\text{drc}}(p_j) = 1$ or $S_{\text{drc}}(p_j) = 1$. With our definition of D_{drc}, probes p that cover dominated targets or separate dominated target-pairs have the highest $D_{\text{drc}}(p)$ values. By selecting a probe p to cover a dominated target t_i or to separate a dominated target-pair t_{ik}, we are also selecting p to cover as many targets as possible (all targets that dominate t_i) or to separate as many target-pairs as possible (all target-pairs that dominate t_{ik}). This is the main greedy probe selection strategy in our heuristics in Section 5. In this paper, we use the following probe selection function, $D_{\text{dps}} : P \mapsto [0, 1]$:

$$D_{\text{dps}}(p_j) = \max\{(C_{\text{dps}}(p_j), S_{\text{dps}}(p_j)) \mid 1 \leq j \leq n\} \ , \tag{12}$$

to favor the dominant probes among all probes that have equal values in D_{drc}; this is the secondary greedy selection principle. These two greedy principles together allow larger coverage and separation when using D_{dps} than D_{drc} in a greedy search method.

4 Subset Selection Criteria

Given the initial probe set, $P = \{p_1, \ldots, p_n\}$, the sequential search algorithm, discussed in Section 5, greedily selects the best subset of probes among a collection, $\mathcal{P} \subseteq 2^P$, of subsets; 2^P is the power set of P. In this section, we define the criteria required to decide which is the best subset to select. Let $P^{1\ldots u} = \{q_1, \ldots, q_u\} \subseteq P$ be a probe set to be evaluated, where $q_j \in P, 1 \leq j \leq u$ and $1 \leq u \leq n$, and $P^{1\ldots 0} = \emptyset$. $P^{1\ldots u}$ $c_{\min}$-covers a target t_i if at least $c_{\min}$ probes in $P^{1\ldots u}$ cover t_i. $P^{1\ldots u}$ $s_{\min}$-separates a target-pair t_{ik} if at least $s_{\min}$ probes in $P^{1\ldots u}$ separate t_{ik}. Our aim is to select the subset $P^{1\ldots u}$ which $c_{\min}$-covers as many target as possible and $s_{\min}$-separates as many target-pairs as possible, or, which satisfies all the constraints with the least cardinality u.

4.1 Coverage Criterion

Given a collection $\mathcal{P} \subseteq 2^P$, we want to choose the subset $P^{1\ldots u} \subseteq P$ such that each target is $c_{\min}$-covered by $P^{1\ldots u}$. Given the matrix H, the parameter $c_{\min}$,

Table 3. Example of subset coverage obtained from Table 1

	$\{p_3\}$	$\cup$	$\{p_1\}$	$=$	P_{31}	P_{32}	P_{34}	P_{35}	P_{36}
t_1	0	$+$	$\frac{c_{\min}}{4}\frac{2}{4}$	$=$	$\frac{c_{\min}}{8}$	$\frac{c_{\min}}{8}$	$\frac{3c_{\min}}{16}$	0	$\frac{3c_{\min}}{16}$
t_2	$\frac{c_{\min}}{3}\frac{3}{4}$	$+$	$\frac{c_{\min}}{3}\frac{2}{4}$	$=$	$\frac{5c_{\min}}{12}$	$\frac{c_{\min}}{4}$	$\frac{c_{\min}}{4}$	$\frac{c_{\min}}{4}$	$\frac{c_{\min}}{2}$
t_3	$\frac{c_{\min}}{5}\frac{3}{4}$	$+$	0	$=$	$\frac{3c_{\min}}{20}$	$\frac{c_{\min}}{10}$	$\frac{3c_{\min}}{10}$	$\frac{c_{\min}}{10}$	$\frac{3c_{\min}}{10}$
t_4	$\frac{c_{\min}}{3}\frac{3}{4}$	$+$	0	$=$	$\frac{3c_{\min}}{20}$	$\frac{c_{\min}}{4}$	$\frac{c_{\min}}{2}$	$\frac{5c_{\min}}{12}$	$\frac{c_{\min}}{4}$
C_{dps}	$\frac{c_{\min}}{3}\frac{3}{4}$	$+$	$\frac{c_{\min}}{3}\frac{2}{4}$	$=$	$\frac{5c_{\min}}{12}$	$\frac{c_{\min}}{4}$	$\frac{c_{\min}}{2}$	$\frac{5c_{\min}}{12}$	$\frac{c_{\min}}{2}$

the candidate probe set $P = \{p_1, \ldots, p_n\}$ and the target set $T = \{t_1, \ldots, t_m\}$; to evaluate the ability of subset $P^{1\ldots u}$ to $c_{\min}$-cover T, we generalize the coverage function as follows:

$$C_{\text{dps}}(P^{1\ldots u}) = \max_{t_i \in T_{P^{1\ldots u}}} \left\{ \sum_{j=1}^{j=u} \text{cov}_{\text{drc}}(q_j, t_i) \times \frac{1}{m - |T_{q_j}| + 1} \quad | \quad q_j \in P^{1\ldots u} \right\} , \quad (13)$$

where $T_{P^{1\ldots u}} = T_{q_1} \cup \ldots \cup T_{q_u}$ is the set of targets covered by $P^{1\ldots u}$. $C_{\text{dps}}(P^{1\ldots u}) : 2^P \mapsto \Re^+$ is the maximum amount that $P^{1\ldots u}$ can contribute to satisfy the minimum coverage constraints. Table 3 shows an example of a subset coverage table obtained from Table 1, given five subsets. In the example, P_{ab} means the subset $\{q_a, q_b\}$. We also show, for P_{31}, the computation of Equation (13).

Clearly, $C_{\text{dps}}(P^{1\ldots u})$ is maximal if $C_{\text{dps}}(q_j)$ is maximal for each $q_j \in P^{1\ldots u}$. Thus, for subsets of probes, function C_{dps} favors the selection of those subsets that contain probes having the highest coverage values. For example in Table 2, probes p_3, p_4 and p_6 have the highest coverage values, and hence, subsets such as P_{34} and P_{36} have the best values. C_{dps} indicates only how much a subset contributes in satisfying the coverage constraints, not how well the subset satisfies the coverage constraints. For instance, in the table, subsets P_{31} and P_{35} produce a tie, but P_{31} should be preferred since it covers more targets. Also, between the two subsets, which attain the same value of C_{dps}, the one that satisfies all coverage constraints (or, closer to satisfying all coverage constraints) should be preferred. We define the *coverage criterion*, $F_{C_{\text{dps}}} : 2^P \mapsto \Re^+$, as follows:

$$F_{C_{\text{dps}}}(P^{1\ldots u}) = C_{\text{dps}}(P^{1\ldots u}) \times \frac{|T_{P^{1\ldots u}}| - |U_{P^{1\ldots u}}|}{m - |U_{P^{1\ldots u}}|} \times \frac{\sum_{t_i \in T \setminus U_{P^{1\ldots u}}} \text{fea}\left(P_{t_i}^{1\ldots u}\right)}{(m - |U_{P^{1\ldots u}}|) \cdot c_{\min}} , \quad (14)$$

where, $U_{P^{1\ldots u}}$ is the set of targets already $c_{\min}$-covered by $P^{1\ldots u}$ (probes need not be selected to cover such targets); $P_{t_i}^{1\ldots u}$ is the set of probes in $P^{1\ldots u}$ that cover t_i, and $\text{fea} : 2^P \mapsto \Re^+$ defined as

$$\text{fea}\left(P_{t_i}^{1\ldots u}\right) = \begin{cases} \left|P_{t_i}^{1\ldots u}\right| , \text{ if } \left|P_{t_i}^{1\ldots u}\right| < c_{\min} \\ c_{\min} , \text{ otherwise} \end{cases} , \quad (15)$$

specifies how much the coverage constraint is satisfied on t_i; the sum equals $(m - |U_{P^{1\ldots u}}|)\, c_{\min}$ when all coverage constraints are satisfied. Hence, the second

term penalizes subsets that cover fewer targets and the third term penalizes subsets that satisfy fewer coverage constraints. $F_{C_{\text{dps}}}$ is maximal when all three terms are maximal.

4.2 Separation Criterion

The derivation of the *separation criterion* is similar to that of coverage, except that we use terms and variables related to separation; such as, target-pair, $s_{\min}$, and so on, in the equations below. Given a collection $\mathcal{P} \subseteq 2^P$, we want to choose the subset $P^{1\ldots u} \subseteq P$ such that each target-pair is $s_{\min}$-separated by $P^{1\ldots u}$. Consider the matrix H, the parameter $s_{\min}$, the candidate probe set $P = \{p_1, \ldots, p_n\}$ and the target set $T = \{t_1, \ldots, t_m\}$. Following the same reasoning as in Section 4.1, we obtain the following equations for separation:

$$S_{\text{dps}}(P^{1\ldots u}) = \max_{t_{ik} \in T^2_{P^{1\ldots u}}} \left\{ \sum_{j=1}^{j=u} \text{sep}_{\text{drc}}(q_j, t_{ik}) \times \frac{1}{\frac{m(m-1)}{2} - \left|T^2_{q_j}\right| + 1} \mid q_j \in P^{1\ldots u} \right\} , \tag{16}$$

where $T^2_{P^{1\ldots u}} = T^2_{q_1} \cup \ldots \cup T^2_{q_u}$ is the set of target-pairs separated by $P^{1\ldots u}$. $S_{\text{dps}}(P^{1\ldots u}) : 2^P \mapsto \Re^+$ is the maximum amount that $P^{1\ldots u}$ can contribute to satisfy the minimum separation constraints. The *separation criterion* is given by:

$$F_{S_{\text{dps}}}(P^{1\ldots u}) = S_{\text{dps}}(P^{1\ldots u}) \times \frac{\left|T^2_{P^{1\ldots u}}\right| - \left|U^2_{P^{1\ldots u}}\right|}{\frac{m(m-1)}{2} - \left|U^2_{P^{1\ldots u}}\right|} \times \frac{\sum_{t_{ik} \in T^2 \setminus U^2_{P^{1\ldots u}}} \text{fea}\left(P^{1\ldots u}_{t_{ik}}\right)}{\left(\frac{m(m-1)}{2} - \left|U^2_{P^{1\ldots u}}\right|\right) \cdot s_{\min}} , \tag{17}$$

where, $U^2_{P^{1\ldots u}}$ is the set of target-pairs already $s_{\min}$-separated by $P^{1\ldots u}$ (probes need not be selected to separate such target-pairs); $P^{1\ldots u}_{t_{ik}}$ is the set of probes in $P^{1\ldots u}$ that separate t_{ik}, and fea : $2^P \mapsto \Re^+$ defined as

$$\text{fea}\left(P^{1\ldots u}_{t_{ik}}\right) = \begin{cases} \left|P^{1\ldots u}_{t_{ik}}\right| \text{, if } \left|P^{1\ldots u}_{t_{ik}}\right| < s_{\min} \\ s_{\min} \text{, otherwise} \end{cases} , \tag{18}$$

specifies how much the separation constraint is satisfied on t_{ik}; the sum equals $\left(\frac{m(m-1)}{2} - \left|U^2_{P^{1\ldots u}}\right|\right) s_{\min}$ when all separation constraints are satisfied. Thus, the second term penalizes subsets that separate fewer target-pairs and the third term penalizes subsets that satisfy fewer separation constraints. $F_{S_{\text{dps}}}$ is maximal when all three terms are maximal.

4.3 Selection Criterion

As in the selection function of Section 3.3, we combine both the coverage criterion and the separation criterion into a single subset *selection criterion*

$$F_{D_{\text{dps}}}(P^{1\ldots u}) = \max \left\{ F_{C_{\text{dps}}}(P^{1\ldots u}) , F_{S_{\text{dps}}}(P^{1\ldots u}) \right\} , \tag{19}$$

which specifies the degree to which a subset of probes satisfies *all* constraints.

5 Sequential Forward Probe Selection Algorithm

In this section, a sub-optimal technique from pattern recognition is applied for the first time, to the best of our knowledge, to the non-unique probe selection problem. In particular, the well-known *sequential forward selection* (SFS) algorithm [5], for feature subset selection, is adapted to find near-minimal feasible probe sets. Feature selection (FS) constitutes one of the two principal phases of pattern recognition system design, the other being the design of pattern classification stage which employs the selected features. The main goal of FS is to select a subset of d features from the given set of D measurements, $d < D$, without significantly degrading or with possibly improving the performance of the recognition system. Given a suitable criterion function for assessing the *effectiveness* of feature subsets to classify data, FS is reduced to a combinatorial search problem that finds an optimal subset based on the selected measure. The SFS is among the methods [3] [4] [5] proposed by researchers to avoid searching the feature space exhaustively.

A microarray design experiment is a pattern recognition system where the measurements are provided by a biological sample and a target set (augmented with the set of all target-pairs, if non-unique probes are used), and where the classifier system is a probe set that classifies each target, or target-pair, as present or absent in the sample. However, with microarrays, the problem is to reduce the complexity of the classifier system (i.e., the size of the probe set) while still able to correctly classify each target and target-pair as present or absent in the biological sample. Here, the feature space representing the sample, which includes the targets and the target-pairs, is not subject to optimization.

We adapt the SFS to find a near minimal probe set as follows: the best probe set is constructed by adding, to the current non-feasible probe set, one probe at a time until we obtain a feasible probe set with the hope it has the least cardinality u. More specifically, to form the best feasible subset of probes, the starting point of the search is the empty set, $P^{1\ldots 0}$, which is then successively built up. This is known as the bottom up approach. This method is generally sub-optimal since the best probe is always added to a working subset of probes, $P^{1\ldots u}$.

The *sequential forward probe selection* (SFPS) method (Algorithm 1) is based on the SFS algorithm. SFPS uses the $F_{D_{\text{dps}}}$ function as the criterion for selecting the best subset among a collection of probe sets. The best probe, q^+, to insert in a working subset, $P^{1\ldots u}$, is the one that maximizes the criterion, $F_{D_{\text{dps}}}$, when it is included. SFPS terminates when $P^{1\ldots u}$ is feasible; which is then reduced to a near-minimal solution, $P_{\min}$, in Algorithm 2, by removing the redundant probes.

SFPS locally searches the power set, 2^P, of the probe set P. That is, at each subset selection step, the neighborhood of the working subset $P^{1\ldots u} \in 2^P$ is the collection $\mathcal{P}^{1\ldots(u+1)} = \left\{P^{1\ldots u} \cup \{q_1\}, P^{1\ldots u} \cup \{q_2\}, \ldots, P^{1\ldots u} \cup \{q_{n-u}\}\right\} \subset 2^P$, $q_j \in P \setminus P^{1\ldots u}$ for $1 \leq j \leq n-u$. The subset to select is the one in $\mathcal{P}^{1\ldots(u+1)}$ that maximizes the criterion $F_{D_{\text{dps}}}$.

Algorithm 1. Sequential Forward Probe Selection

Input: $T = \{t_1, \ldots, t_m\}$, $P = \{p_1, \ldots, p_n\}$, and $H = [h_{ij}]$
Output: Near-minimal solution $P_{\min}$
  Compute $D_{\text{dps}}(p)$ for all $p \in P$;
  $u \leftarrow$ number of essential probes;
  $P^{1\ldots u} \leftarrow$ set of essential probes;
  repeat
    $q^+ \leftarrow \arg\max_{q \in P \setminus P^{1\ldots u}} F_{D_{\text{dps}}}\left(P^{1\ldots u} \cup \{q\}\right)$;
    $P^{1\ldots(u+1)} \leftarrow P^{1\ldots u} \cup \{q^+\}$;
    $u \leftarrow u + 1$;
  until $P^{1\ldots u}$ is feasible;
  Return $P_{\min} \leftarrow$ Reduction$(P^{1\ldots u}, P, T, H)$.

Algorithm 2. Reduction

Input: $P^{1\ldots u}$, P, T, H
Output: Reduced solution P_{red}
  $P_{\text{red}} \leftarrow P^{1\ldots u}$;
  $H \leftarrow H|_{P_{\text{red}}}$, */* restrict to P_{red} */*;
  Compute $D_{\text{dps}}(q)$ for all $q \in P_{\text{red}}$;
  Sort $P_{\text{del}} \leftarrow \{q \in P_{\text{red}} \mid D_{\text{dps}}(q) < 1\}$ in increasing $D_{\text{dps}}(q)$;
  if $P_{\text{red}} \setminus \{p\}$ is feasible for each $q \in P_{\text{del}}$ **then**
    $P_{\text{red}} \leftarrow P_{\text{red}} \setminus \{q\}$;
  end if
  Return P_{red}.

6 Computational Experiments

We performed experiments to show the minimization ability of SFPS and that it outperform all the greedy methods currently published in literature for the non-unique probe selection problem. The programs were written in C and all tests ran on two Intel Xeon$^{\text{TM}}$ CPUs 3.60GHz with 3GB of RAM under Ubuntu 6.06 i386.

We conducted experiments on ten artificial data sets and three real data sets, that were kindly provided by Dr. Ragle and Dr. Pardalos [6]. These data sets were used in all previous studies mentioned in Section 1, except for HIV-1 and HIV-2 sets which were used only in [2] [6] [8] [9]. Table 4 shows, in the second and third columns, the dimension $|T| \times |P|$ (number of targets $\times$ number of probes) of the incidence matrix for each set (M for Meiobenthos is the largest set). Column A is the number of required virtual probes inserted into P to maintain the feasibility of the initial probe sets P. Due to space constraints, we refer the readers to [1] [2] [7] for the full details on the construction of these data sets. All experiments were performed with parameters $c_{\min} = 10$ and $s_{\min} = 5$, as in all previous studies.

Table 4 shows, for all data sets, the minimum sizes $|P_{\min}|$ attained by the greedy methods, GrdS of [7], GrdM of [2], DRC and DPS of [8], our SFPS

Table 4. Size of $P_{\min}$ for each heuristic

Set	$\|T\|$	$\|P\|$	A	GrdS	GrdM	DRC	DPS	SFPS	ILP
a1	256	2786	6	1163	568	549	547	530	503
a2	256	2821	2	1137	560	552	537	516	519
a3	256	2871	16	1175	613	590	577	557	516
a4	256	2954	2	1169	597	579	578	557	540
a5	256	2968	4	1175	605	583	571	558	504
b1	400	6292	0	1908	961	974	921	883	879
b2	400	6283	1	1885	976	1013	942	890	938
b3	400	6311	5	1895	951	953	915	896	891
b4	400	6223	0	1888	1001	1019	956	920	915
b5	400	6285	3	1876	1022	1019	969	933	946
M	679	15139	75	3851	2336	2084	2068	2036	3158
HIV-1	200	4806	20	-	531	487	472	468	-
HIV-2	200	4686	35	-	578	506	501	492	-

method, and the integer linear programming technique, ILP of [1]. In the table, the final $P_{\min}$'s include the virtual probes inserted into P.

Table 5 reports the improvements, Imp, of SFPS over GrdS, GrdM, DRC, DPS and ILP, computed as in Equation 20 below.

$$\text{Imp} = \frac{P_{\min}^{\text{SFPS}} - P_{\min}^{\text{Heu}}}{P_{\min}^{\text{Heu}}} \times 100 \ , \tag{20}$$

where Heu is either GrdS, GrdM, DRC, DPS or ILP. A negative (positive) value of Imp means that a SFPS result is Imp% better (worse) than Heu result. Consequently, Imp is negative when SFPS returns a probe set smaller than $P_{\min}^{\text{Heu}}$. Therefore, the smaller the value of Imp, the better is SFPS.

SFPS substantially outperformed all the other greedy methods in all instances. GrdS and GrdM use different local search methods to find probes that satisfy the constraint on each target and target-pair. They use no probe selection function and thus, they do not *know* which probes are good or bad to select. DRC and DPS use a local search method similar to that in GrdM, but are guided by probe selection functions to decide which probes are best to select or not. SFPS uses the probe selection function, D_{dps}, of DPS but only to evaluate the effectiveness of each individual probe in a probe set. SFPF does not *select* the best probes, as in DPS, to construct a near minimal probe set; it uses the criterion $F_{D_{\text{dps}}}$ to select the best subset from a collection of probe sets, as explained in Section 5. DPS locally searches the probe set P, where the neighborhood of a probe $q \in P$ is the set of probes that cover the same targets and separate the same target-pairs as q. SFPS locally searches the power set 2^P; which is *more global* than DPS search strategy. Therefore, as expected, SFPS performs better than

Table 5. Improvements of SFPS over GrdS, GrdM, DRC, DPS and ILP

Set	GrdS	GrdM	DRC	DPS	ILP
a1	−54.43	−6.69	−3.46	−3.11	+5.37
a2	−54.62	−7.86	−6.52	−3.91	−0.58
a3	−52.60	−9.14	−5.59	−3.47	+7.95
a4	−52.35	−6.70	−3.80	−3.63	+3.15
a5	−52.51	−7.77	−4.29	−2.28	+10.71
b1	−53.72	−8.12	−9.34	−4.13	+0.46
b2	−52.79	−8.81	−12.14	−5.52	−5.12
b3	−52.72	−5.78	−5.98	−2.08	+0.56
b4	−51.27	−8.09	−9.72	−3.77	+0.55
b5	−50.27	−8.71	−8.44	−3.72	−1.37
M	−47.13	−12.84	−2.30	−1.55	−35.53
HIV-1	-	−11.86	−3.90	−0.85	-
HIV-2	-	−14.88	−2.77	−1.80	-

DPS, DRC, GrdM and GrdS, due to its ability to assess the effectiveness of a probe set and its ability to search 2^P.

Also, SFPS achieved a greater reduction on the M set than all the others methods, including ILP. The authors of [1], first applied GrdS to reduce the initial probe sets (and to reduce the ILP running time), and then further optimized the *reduced probe sets* with ILP solver CPLEX (CPLEX is one of the leading mathematical programming software packages available and few heuristics, if any, are able to compete with its results). CPLEX was *restricted* to search only a small portion of the solution space, hence ILP was not aware of the full initial probe sets. SFPS had no such restriction. The improvements of SFPS over ILP are still quite small, but it implies that one could obtain better results than ILP, with better functions, than our D_{dps} or $F_{D_{\text{dps}}}$, or with a better search method, than our SFPS method.

7 Conclusions and Future Research

In this paper, the sequential forward search algorithm is applied for the first time to solve the non-unique probe selection problem. SFPS outperformed all the currently published greedy algorithms for non-unique probes and gave results close to the optimal search method of ILP. SFPS also suffers from the *nesting effect* of SFS; that is, a probe that was selected cannot be discarded later to correct a wrong decision, and hence, the solution tends to be sub-optimal. The main cause of the nesting effect is the use of a non-monotonic criterion such

as our $F_{D_{\text{dps}}}$ criterion. We are investigating sequential methods, such as the *floating search* methods of [5], which reduces the nesting effect and cope with non-monotonic criterion functions.

References

1. Klau, G.W., Rahmann, S., Schliep, A., Vingron, M., Reinert, K.: Integer Linear Programming Approaches for Non-unique Probe Selection. Discrete Applied Mathematics 155, 840–856 (2007)
2. Meneses, C.N., Pardalos, P.M., Ragle, M.A.: A New Approach to the Non-Unique Probe Selection Problem. Annals of Biomedical Engineering 35(4), 651–658 (2007)
3. Moret, B.M.E., Shapiro, H.D.: On Minimizing a Set of Tests. SIAM Journal on Scientific and Statistical Computing 6(4), 983–1003 (1985)
4. Payne, R.W., Preece, D.A.: Identification Keys and Diagnostic Tables: a Review. Journal of the Royal Statistical Society, Series A 143(3), 253–292 (1980)
5. Pudil, P., Ferri, F.J., Novovičová, J., Kittler, J.: Floating Search Methods for Feature Selection with Nonmonotonic Criterion Functions. In: IAPR 12th International Conference on Pattern Recognition, Jerusalem, Israel, vol. 2, pp. 279–283 (1994)
6. Ragle, M.A., Smith, J.C., Pardalos, P.M.: An Optimal Cutting-Plane Algorithm for Solving the Non-Unique Probe Selection Problem. Annals of Biomedical Engineering 35(11), 2023–2030 (2007)
7. Schliep, A., Torney, D.C., Rahmann, S.: Group Testing with DNA Chips: Generating Designs and Decoding Experiments. In: IEEE Computer Society Bioinformatics Conference (CSB 2003), pp. 84–91. IEEE Press, Stanford (2003)
8. Wang, L., Ngom, A., Rueda, L., Gras, R.: Selection Based Heuristics for the Non-Unique Oligonucleotide Probe Selection Problem in Microarray Design. In: IEEE/ACM Transactions on Computational Biology and Bioinformatics (under review) (2008)
9. Wang, L., Ngom, A., Gras, R.: Non-Unique Oligonucleotide Microarray Probe Selection Method Based on Genetic Algorithms. In: 2008 IEEE Congress on Evolutionary Computation, Hong Kong, China, pp. 1004–1011. IEEE Press, Los Alamitos (2008)

On Finding and Interpreting Patterns in Gene Expression Data from Time Course Experiments

Yvonne E. Pittelkow and Susan R. Wilson

Mathematical Sciences Institute, Australian National University
Canberra ACT, Australia 0200

Abstract. Microarrays are being widely used for studying gene activity throughout a cell cycle. A common aim is to find those genes that are expressed during specific phases in the cycle. The challenges lie in the extremely large number of genes being measured simultaneously, the relatively short length of the time course studied and the high level of noise in the data. Using a well-known yeast cell cycle data set, we compare a method being used for finding genes following a periodic time series pattern with a method for finding genes having a different phase pattern during the cell cycle. Application of two visualisation tools gives insight into the interpretation of the patterns for the genes selected by the two approaches. It is recommended that (i) more than a single approach be used for finding patterns in gene expression data from time course experiments, and (ii) visualisation be used simultaneously with computational and statistical methods to interpret as well as display these patterns.

1 Introduction

DNA microarrays have enabled the simultaneous monitoring of the expression patterns of thousands of genes during cellular differentiation and response. A major challenge has been to find and interpret patterns in these massive data sets. Cluster analysis is one of the main methodologies used to study such data and find patterns. It is well known that there is no straightforward, rigorous way to quickly extract clusters from complex, high-dimensional data and hundreds of algorithms have been proposed [1]. As noted in [2] 'All the [cluster analysis] methods have their limitations and weak points. That is why it is so important to look at the clustering problem from multiple perspectives'.

One widely used application of DNA microarrays is for the study of the cell cycle transcriptome. For some time it has been clear that certain genes are expressed at specific stages of the cell cycle [3]. So these genes show a periodic pattern of expression when monitored during consecutive cell cycles. Clustering methods have been proposed for finding patterns in these time course data; including self-organizing maps [4], principal component analysis [5], amongst other methods (see [6]). Alternative approaches specifically for finding relevant, periodic patterns in time course data have been proposed; see, for example, [3], [7], [8], [9]. Generally, the foremost aim for these latter methods is to find cell cycle regulated genes.

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 276–287, 2008.

When searching for, and evaluating, patterns of gene expression, a statistical modelling approach has the following, major advantage compared with non-statistically based, computational modelling approaches that underpin most clustering methods. Namely, the type of patterns being found can be compared in a rigorous manner [10]. For time course data, recently we proposed a novel statistical model for selection of genes with different phases and/or amplitudes [9] and compared the results of applying this approach to the Cho *et al* data with the results previously published ([11], [3]).

As noted above, there are limitations for all approaches, and it is important to use and compare different perspectives. So we focus here on two approaches for finding relevant patterns in gene expression time course data, namely (i) Fisher's exact test for hidden periodicities of unspecified frequency [12], [7], and (ii) the absolute sine model [9]. A valuable step when analysing genome-wide expression is to visualise the results from the analysis in such a way as to facilitate interpretation of the data, including the patterns found, as well as those not found [6]. Here two useful visualisation tools for interpreting patterns simultanously in a large number of expression measures are used, the h-profile plot [9] and the *GE*-biplot [14].

The next section summarises the two approaches for finding patterns in time course data and the visualisation tools, and the Cho et al [11] data are described in section 3. The statistical approaches and visualisation tools are applied to these data in section 4, and the results presented there illustrate the usefulness of the two statistical approaches for finding different types of patterns in the time course data, and of the visualisation methods for interpreting these patterns.

2 Methods Summary

2.1 Fisher's Exact Test for Hidden Periodicities (FET)

Consider an observed time series $z_1, ..., z_N$ of gene expression values (possibly transformed). Fisher devised an exact procedure based on the periodogram to test the null hypothesis of Gaussian white noise against the alternative of an added deterministic periodic component of unspecified frequency. Basically, the null hypothesis will be rejected if the periodogram contains a value significantly greater than the average value; see 10.2 in [12]. Writing $[r]$ for the integer part of [r], the statistic is given by

$$\xi_r = \{\max_{1 \le k \le [r]} I(\omega_k)\} / \{\Sigma_{k=1}^{[r]} I(\omega_k)\}$$

where

$$I(\omega_k) = N^{-1} |\Sigma_{t=1}^{N} z_t e^{-it\omega_k}|^2, \ \omega_k = 2\pi k/N.$$

In [7], $[N/2]$ is used for $[r]$, but $[(N-1)/2]$ is the correct form; see [12], [13]. The significance level for the corresponding test is given by

$$P(\xi_r \ge x) = 1 - \Sigma_{j=0}^{[r]} (-1)^j \binom{[r]}{j} (1 - jx)_+^{[r]-1}$$

where $y_+ = \max(y, 0)$.

This procedure has been applied to multiple time series data derived from microarray experiments ([7], [15], [8]). For this application, it is referred to as the '(g)-statistic' and 'g test'. The challenge of multiple testing is addressed using the False Discovery Rate (FDR) that controls the expected proportion of false positives. If G genes are considered, first the corresponding p-values are ordered, $p_{(1)}, ..., p_{(G)}$ with corresponding genes $g_{(1)}, ..., g_{(G)}$, then j_q, the largest j such that $p_{(j)} \leq (j/G)q$, is determined. The null hypothesis is rejected for genes $g_{(1)}, ..., g_{(j_q)}$. This controls the FDR at level q. The GeneCycle package in R [16] implements the approach outlined in [7] that we refer to as FET-gs. Note that during revision of the paper, GeneCycle was updated to use the correct form of [r].

2.2 Absolute Sine Model (ASM)

Many genes, rather than having periodic behaviour instead may be following a different pattern in each cycle. Based on many of the profile plots that appeared to depict this behavior, we proposed the following model [9] for gene expression data

$$Z_t = |Asin(2\pi Kt + L)|,$$

where A is the amplitude, K the period and L the part of the cycle at time zero, i.e. related to phase. This model allows the selection of genes with different phases and/or amplitudes.

If $K = 1$ and time t is scaled to run from 0 to 1, there are exactly two cycles so that genes whose profiles complete two cycles and have approximately equal amplitude in both cycles will be selected. A scale free estimate of residual error $RSS = \sum_t ((z_t - \hat{z_t}^*)/\hat{A})^2$ is obtained, where $\hat{z_t}^* = |\hat{A}sin(2\pi t + \hat{L})|$. To assess fit based on RSS, simulation was used to determine the quantiles of the distribution of RSS, denoted by $\hat{c}$. Estimates of RSS_g for all genes, g, were calculated and compared to $\hat{c}$. All genes, where $\hat{RSS}_g \leq \hat{c}$ were selected as being compatible with the two cycle model, where different values of $\hat{c}$ select genes with more or less compatability with the model.

Values of $\hat{A}$ provide information on the extent of gene expression change (amplitude), and the corresponding value of t is an estimate of the time of maximal expression. The value of the intercept parameter $\hat{L}$ gives an estimate of the expression at the beginning of the cycle.

2.3 Visualisation Tools

It is advised that 'an essential *first* step [ ] when considering *any* time series is simply to plot the observation against time' [17]. This is straightforward if one is considering a handful of time series data, but when there are hundreds, or even thousands, of series it is more problematic. Two visualisation tools, the h-profile plot and the covariance-biplot (and a variant called the *GE*-biplot) are useful in this setting, and are now described.

Let $\mathbf{Z}$ be the matrix of expression values, or functions of gene expression values, with G 'genes' in the columns and N microarrays (one for each time point

in this application) in the rows, such that the column (gene) means are zero. To illustrate ideas, we describe the methods as if Z contained gene expression values and refer to the columns as 'genes', although this is not strictly correct.

Let the SVD of $\mathbf{Z}$ be $\mathbf{Z} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^T$, where $\mathbf{U}$, size N by N, and $\mathbf{V}^T$, size G by G, are orthogonal matrices such that $\mathbf{U}^T\mathbf{U} = \mathbf{I}$ and $\mathbf{V}^T\mathbf{V} = \mathbf{I}$ (where $\mathbf{I}$ is used to denote a conformable identity matrix).

In both plots, the coordinates for all genes, in d dimensions, are defined as

$$\tilde{\mathbf{G}}_d^T = \frac{1}{\sqrt{N-1}}\mathbf{\Lambda}_d\mathbf{V}_d^T,$$

where $\mathbf{U}_d$ and $\mathbf{V}_d$ are matrices comprising the first d columns of $\mathbf{U}$ and $\mathbf{V}$ respectively and $\mathbf{\Lambda}_d$ is a sub matrix of $\mathbf{\Lambda}$ formed from the first d columns and rows of $\mathbf{\Lambda}$. For two dimensional representation $d = 2$ and then $\tilde{\mathbf{G}}_2^T$ consists of pairs of co-ordinates, one for each gene, defining the location of gene points on the horizontal and vertical axes.

Genes, represented by gene points, have expression values which increase in variance as distance from the origin increases. The angular separations between the gene points are approximations to cosines of correlation between the gene expression profiles. So genes that are highly correlated will lie approximately on a line passing through the origin, with those that are positively correlated on the same side of the origin, while those that are negatively correlated lie on the opposite side.

In the h-profile plot reduced versions of a plot (thumbnail) for each gene is placed at the gene points. When the reduced plot is a time series graph, and Z contains the gene expression values, profiles of similar 'shape' are located together, while those of 'reversed' shape lie on the opposite side of the origin.

For the covariance-biplot, where microarrays and the genes are simultaneously displayed, the gene coordinates are as in the h-profile plot and the microarray coordinates are given by

$$\tilde{\mathbf{C}}_d = \sqrt{N-1}\mathbf{U}_d.$$

When Z consists of gene expression values that have been logged, standardized over each microarray, and finally column mean corrected, the covariance-biplot is called the *GE*-biplot [14].

In these biplots, the scalar product between the t^{th} row point (microarray point) and g^{th} column point (gene point) with respect to the origin is approximately equal to the $(t, g)^{th}$ element, $z_{t,g}$, of $\mathbf{Z}$. The juxtapositions of the gene points to the microarray points provides an approximation to the value of the (transformed) gene expression values on the microarrays. The inner product can be viewed geometrically as the product of the signed length of one of the vectors and the length of the projection of the other vector onto it. Thus if a gene point is close to a microarray point then the gene will be relatively up regulated in that microarray and if the gene point is on the opposite side of the plot to the microarray point then the gene will be relatively down regulated on that microarray. The accuracy of these predictions depends on how good the approximation is in the lower ranked space; two measures of fit, I_1 and I_2, ranging from 0 to 1, can be determined [14], [9]. R source code is available at [18].

For time course data, one would expect those microarrays falling in the same cell phase to be relatively close to one another, and the greater the separation between the different cell phases, the greater the distance between the corresponding microarrays.

A novel application of these plots for time series data demonstrated in this paper, replaces the gene expression values in the columns of Z by the residuals arising after first fitting 'some model or other' to the series. Such (initial) model fitting is often necessary in time series analyses [17].

3 Mitotic Cell Cycle Data

The aim of the time-course experiment described in Cho *et al* [11] was to characterize mRNA transcript levels during the cell cycle of the budding yeast S. *cerevisiae*. Synchronous yeast cultures were arrested in late G1 and the cell cycle re-initiated with cells collected at 10 minute intervals, covering two full cell cycles. The time course was divided into early G1, late G1, S, G2 and M phases based on the size of the buds, the cellular position of the nucleus, and standardization to known transcripts.

For our analyses of these data, the negative values were truncated to .01, the data logged using base 2 (as is commonly done, see for example [7]), and finally standardized so that, for each microarray, the mean over all values is zero and the corresponding variance is one. This latter transformation effectively 'normalizes' the distributions so that the first two moments of the distributions on each microarray agree. Control genes were removed leaving a total of 6565 genes. Although not technically correct the transformed gene expression is often referred to as simply 'gene expression' to avoid cumbersome phrases. Preprocessing can have a large impact on analyses, but such considerations are beyond the scope of this paper. The sample at time zero, that is immediately after arrest, was eliminated from the following analyses, leaving 16 microarrays, one at each 10 minute interval, from time 10 to 160 minutes.

4 Results

Using FET-gs in the GeneCycle package [16] with an FDR of 0.05, 532 genes were selected. On the left of Fig.1 is the *GE*-biplot where the genes are shown as symbols, marking their positions relative to the microarray points which are shown as numerals indicating the time in minutes. The microarrays are coloured according to their cell phase determined by Cho *et al*, and the same colouring is applied to the genes according to the phase in which the time of maximum (TOM) occurred. TOM was determined by averaging the four values in each phase and then finding the maximum. On the right of Fig.1 is the h-profile plot using a periodogram as the thumbnail [16], with different colours differentiating the estimates of k. The measures of fit are $I_1 = 0.65$ and $I_2 = 0.88$.

Previously, in the *GE*-biplot for these data using genes selected by ASM [9], microarrays allocated to the different coloured phases appeared in the same

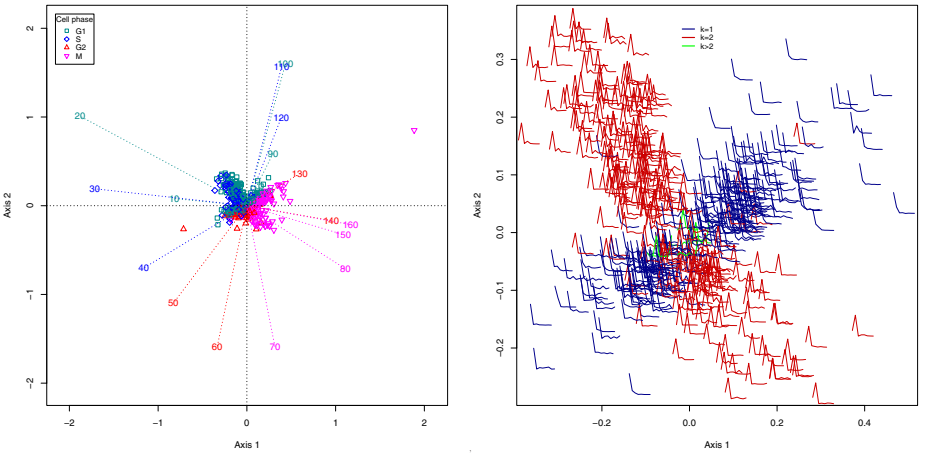


Fig. 1. On the left is a *GE*-biplot using the 532 genes selected by FET-gs (FDR=0.05). The microarray points are shown as coloured numerals (time in minutes) indicating the phase determined by Cho *et al* (see legend). The genes are shown as coloured symbols, marking their positions relative to the microarray points, where the colour (and symbol) analogously are according to the phase in which time of maximum occurred. On the right is a corresponding h-profile plot (outliers removed) showing the periodograms for 531 of these genes (after removing the outlier). The periodograms are coloured to differentiate the k values (see legend, and the equations in Section 2.1).

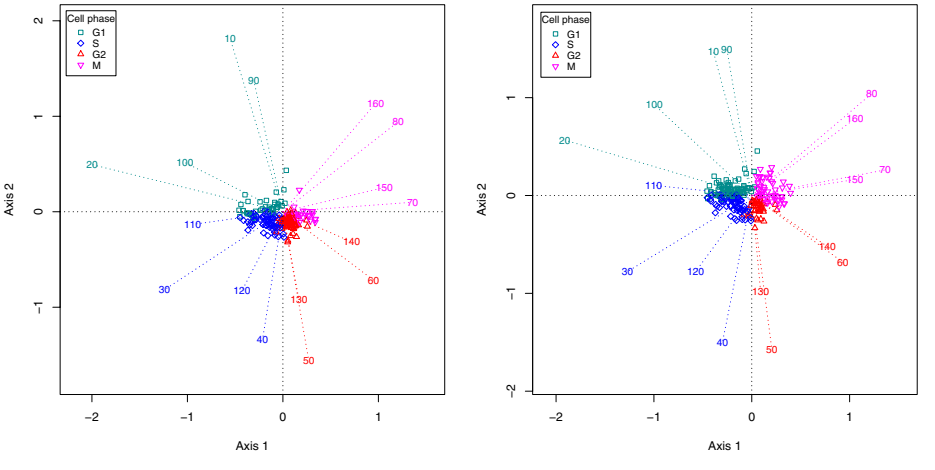


Fig. 2. On the left is the *GE*-biplot using the 221 genes determined by ASM ($\hat{c} = 0.6$) and on the right the *GE*-biplot using the 254 genes determined by FET-gs (FDR=0.05) restricted to $k = 2$. The colours, numbers and symbols are as described for the *GE*-biplot in the caption for Fig.1.

region and showed a strict ordering of the microarrays in time around the origin indicating strong cyclic behaviour in time. The plot on the left of Fig.2, that uses the 221 genes selected by ASM (with $\hat{c} = 0.6$), exemplifies this form of cyclic behaviour. Such clear, cyclic, behaviour is not apparent in Fig.1. For example, microarrays in the first G2 cell phase (50, 60 minutes) are quite separated from those in the second G2 phase (130, 140 minutes). In the plot on the left of Fig. 2, these microarrays are relatively close to each other.

In the h-profile plot in Fig.1, it is clear that only about half (254) of the 532 genes correspond to $k = 2$, the value one would expect for data collected for two cell cycles. The *GE*-biplot for these (254) genes (when $k = 2$) is shown on the right in Fig.2. Now it can be seen that the microarrays are positioned in such a way as to reflect the cell phases that are known for these yeast data. The two plots in Fig.2 are quite similar to each other, with clear separation of the microarrays and (most of) the genes into the four distinct phases. The measures of fit are essentially identical, and much better than those for Fig.1, namely $I_1 = 0.82$ and $I_2 = 0.99$. The corresponding h-profile plots are given in Fig.3.

From the FET-gs results, 7 genes had estimated k values greater than 2 (2 with a value of 3, 1 a value of 4, and 4 with value 8). The remaining 271 genes had an estimated k value of 1, and Fig.4 gives a *GE*-biplot and an h-profile plot for these genes. The measures of fit are essentially identical to those for Fig.2 The three groups of genes seen in the biplot of Fig.4 correspond to the first 6 time points (10 to 60 minutes), the next 5 (70 to 110 minutes) and the final 5 (120 to 160 minutes). From the h-profile plot in Fig.4, it appears that gene profiles towards the upper right corner might have an upward slope, while those

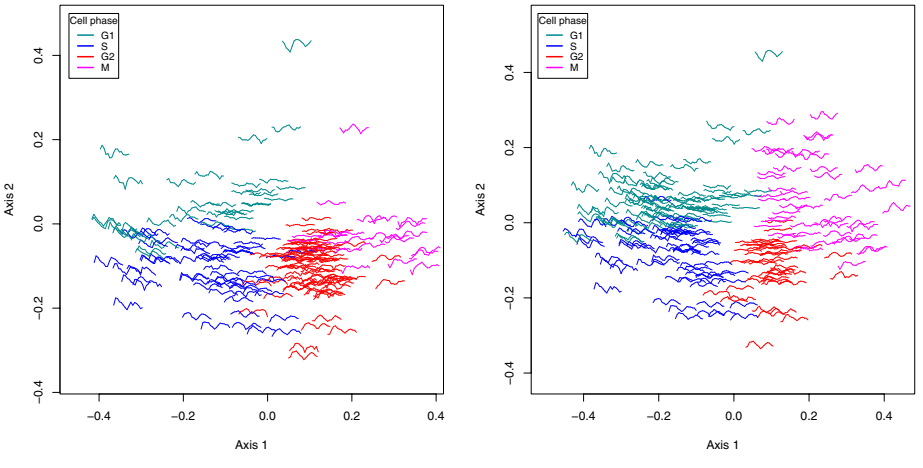


Fig. 3. On the left is the h-profile plot for the 221 genes determined by ASM ($\hat{c} = 0.6$), and on the right the corresponding plot for the 254 genes determined by FET-gs (FDR=0.05) restricted to $k = 2$. The h-profile plot uses the standard time series plots of the (transformed) gene expression values plotted over the 16 time points. The colours are described for the *GE*-biplot in the caption for Fig.1.

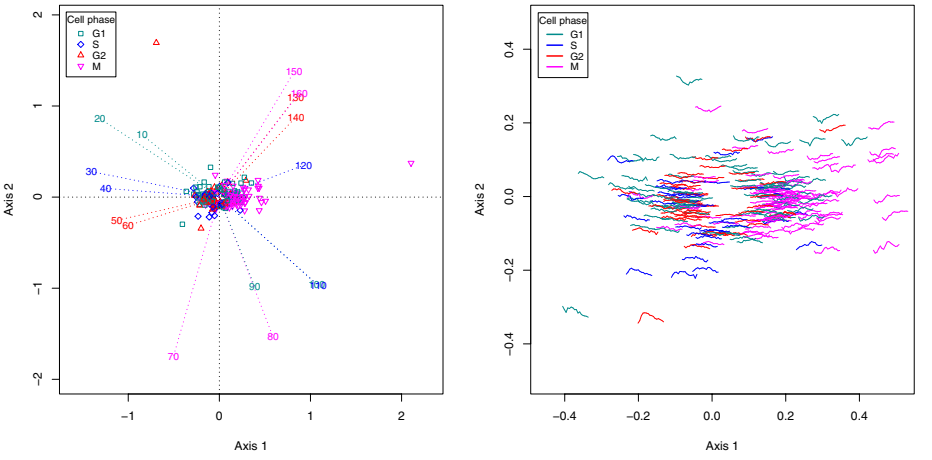


Fig. 4. On the left is the *GE*-biplot using the 271 genes determined by FET-gs (FDR=0.05) restricted to $k = 1$, and on the right is the corresponding h-profile plot with outlying genes removed. The colours, numbers and symbols are as described for the *GE*-biplot in the caption for Fig.1, with the profiles as described for Fig. 3.

towards the lower left might slope downward. These slopes are not so obvious in the h-profile plot on the right of Fig.3 for the 2-cycle genes found by FET-gs.

Now FET is a test for periodicity with the null hypothesis of randomness. Such a test is likely to be affected by other deviations from randomness such as a trend. So we detrended the (transformed) gene expression data, by applying ordinary least squares (linear) regression, and used the residuals in FET-gs (FDR=0.05). The outcome was quite stunning. The number of genes selected fell dramatically, from 532 to just 82. The I_1 and I_2 measures of fit improved slightly (now 0.74 and 0.9 respectively, compared with 0.65 and 0.88 previously). The covariance-biplot is given on the left in Fig.5. Amongst the 82 genes, 59 (72%) had a k-value of 2, 22 a value of 1 and one a value of 4. Amongst the 59 two-cycle genes, 21 were in the top 59 genes found using ASM. On the right of Fig.5, we give an h-profile plot showing these 21 genes as well as the 38 that were uniquely found for the detrended data using FET-gs, FDR=0.05, k=2, and the 38 that were unique to the top 59 found from fitting ASM.

We selected 8 genes for closer comparisons, namely 4 of the genes unique to FET-gs lying on the left hand side of the h-profile plot in Fig.5, and 4 that were unique to ASM and well separated from the first 4 genes. The genes are identified on the h-profile plot. In Fig.6 we give time series profiles for these 8 genes, distinguishing the 4 unique to FET-gs on the left plot, from the 4 unique to ASM on the right. Note the different shapes of the profiles for the genes selected by FET-gs, after accommodating the (slight) phase shift compared with the profiles for the genes selected by ASM. This demonstrates that different approaches are selecting genes with profiles that have distinct patterns.

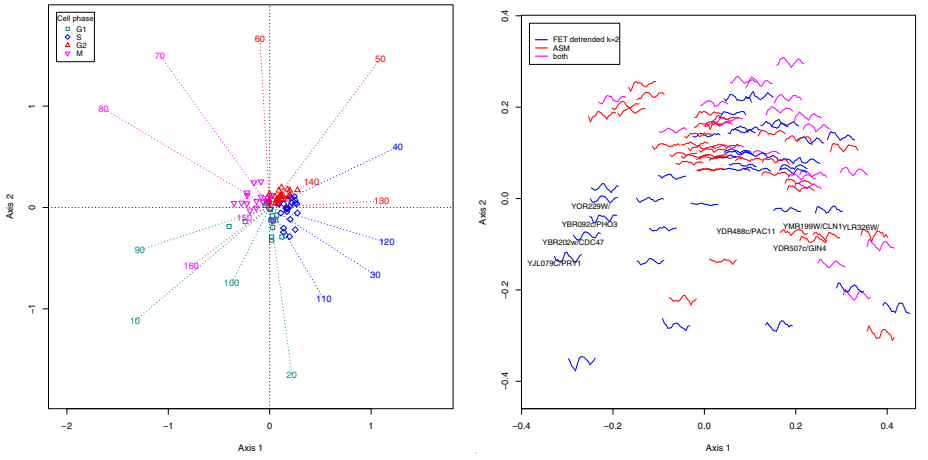


Fig. 5. On the left is the covariance-biplot using the 82 genes determined by FET-gs (FDR=0.05) after detrending (fitting a linear model to) the (transformed) gene expression values. The coordinates use the residuals from the linear model. On the right is an h-profile plot using the genes selected by different approaches; see legend. Eight of the genes selected for Fig.6 are identified.

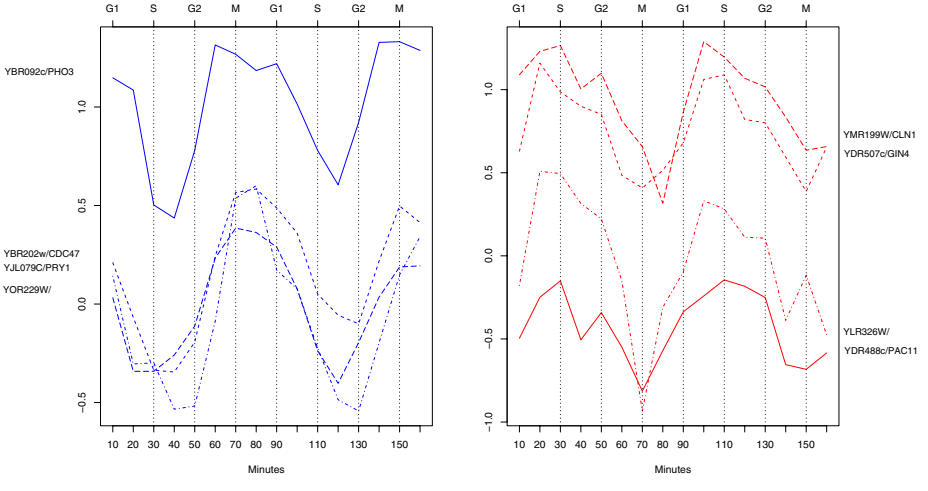


Fig. 6. Detailed time series profiles for 4 of the genes uniquely selected by FET-gs (using residuals, FDR=0.05) on the left, and 4 genes selected uniquely by ASM ($\hat{c} = 0.6$) on the right. The colours correspond to those used on the right in Fig.5.

Two of Cho *et al*'s [11] landmark genes are in our selection. One of these is CLN1 (found by ASM, and depicted in the plot in the right in Fig.6), and the other is CDC47 (found by FET-gs, and depicted in the plot on the left in Fig.6). Cho *et al* identified CLN1 as being characterized by the specific cell cycle

phase (late) G1. Previously we used CLN1 to show how one can determine those genes that are highly correlated with a (landmark) gene of interest [9], and found there, as here, that it peaks in the G1 phase. CLN1 was one of the two genes that Cho *et al* found to have the largest change of 25-fold. So it is interesting that this gene was not found using FET-gs (with the detrended data). In Fig.6, CDC47 has a profile that peaks in the M phase of the cell cycle. Cho *et al* used CDC47 as an early G1 phase landmark, although from their (Fig4C) plot it would appear to peak during phase M. In the space available, these results briefly illustrate that the ability to visualise the profiles allows the researcher to examine the approaches being used for finding genes and gives added insight into the findings from the analyses.

5 Discussion

There are no true gold standards for finding expressed genes that vary systematically during the cell cycle, and so absolute assessment of different methods is not possible. Until now, many approaches have been restricted to finding periodically expressed genes using time series methods. We advocate that alternative models should also be applied.

We note that complete evaluation also requires the integration of results from the genes found and the visual interpretation along with biological background. For example, it has been argued (e.g. [7], [3]) that due to the synchronisation technique used, the cells may be perturbed so some of the observed periodic genes are due to stress response rather than cell cycle activity. In other words some genes selected may be artifacts due to the treatment of the cells that would not occur in freely growing cells. Certainly this could explain the genes selected for $k = 1$.

On first consideration, one might think that the two approaches, ASM and FET, should be directly comparable. In general, they are not. ASM was developed to *model* the pattern of gene expression in a cycle in a specified manner (see section 2.2) while, on the other hand, FET is a *test* for periodicity with the null hypothesis of randomness and the (implicit) underlying model is quite different [12]. Further, the ASM could be generalised to allow a differently shaped function with a more pronounced peak where the gene is "switched on" compared with the expression value in the other phases, where say the gene is "switched off".

For FET, residuals from other detrending or modelling approaches, such as from fitting a quadratic function, could result in different genes being selected. We also note the possibility of different results from using FET-gs from [16], rather than FET in [12]. Further, the FDR assumes independence of the genes that obviously does not hold, as many genes are expected to be highly correlated with one another. Evaluation of this assumption is beyond the scope of the paper.

Preprocessing and transformation of the data has an effect whose size has not been consistently evaluated. Currently, there are no agreed guidelines, and consideration of the sensitivity of the results to this decision is beyond the scope of this paper. Further, we note that the common practice of 'norming' the gene

expression values for each gene will distort the plots as then the gene points will tend to lie on the circumference of a circle, as the variance has been set to 1 for all genes.

FET as proposed in [7] has been used for unevenly spaced time points [8], and detrending does not seem to have been considered. The ASM can be used whether the time points are evenly or unevenly spaced. A robust alternative to FET has been proposed [13] and this could be usefully evaluated, as could the application of methods for testing for fixed periodicities, as outlined in [12], when one was only interested in, say, finding all genes completing two cycles during the experiment.

In [7], averaging was recommended. We note that if there are approximately an equal number of genes in different phases then averaging over all the genes would result in no periodicity being able to be detected. Averaging would only determine periodically expressed genes if the number exhibiting the identical periodic behaviour were significantly greater than the remainder.

6 Conclusions

Fisher's exact test (FET) is being widely used for finding periodic patterns for gene expression data from time course experiments. The Absolute Sine Model (ASM) is an alternative approach to finding patterns during a cell cycle. Using a well-known yeast data set, we applied these two approaches, as well as two visualisation methods that enable (i) genes and arrays to be displayed simultaneously (the *GE*-biplot) and (ii) profiles for a large number of genes to be displayed (the h-profile plot). These visualisation tools enabled many insights to be obtained, including the differentiation of the genes into groups according to their periodicity, and the need to detrend the gene expression values first before applying FET.

This paper highlights the advantages of (i) using visualisation methods simultaneously with computational and statistical approaches, and (ii) using more than a single approach for finding patterns in gene expression data from time course experiments, as different approaches can highlight uniquely different aspects of the gene expression patterns.

References

1. Kettenring, J.R.: The practice of cluster analysis. J. Classif. 23, 3–30 (2006)
2. Kettenring, J.R.: A perspective on cluster analysis. Stat. Anal. Data Mining, 52–53 (2008)
3. de Lichtenberg, U., Jensen, L.J., Fausboll, A., Jensen, T.S., Bork, P., Brunak, S.: Comparison of computational methods for the identification of cell cycle-regulated genes. Bioinformatics 21, 1164–1171 (2005)
4. Tamayo, P., Slonim, D.S., Mesirov, J., Zhu, Q., Kitareewan, S., Dmitrovsky, E., Lander, E.S., Golub, T.R.: Interpreting patterns of gene expression with self-organising maps: Methods and application to hematopoietic differentiation. Proc. Natl. Acad. Sci. 96, 2907–2912 (1999)

5. Yeung, K.Y., Ruzzo, W.L.: Principal component analysis for clustering gene expression data. Bioinformatics 17, 763–774 (2001)
6. Johansson, D., Lindgren, P., Berglund, A.: A multivariate approach applied to microarray data for identification of genes with cell cycle-coupled transcription. Bioinformatics 19, 467–473 (2003)
7. Wichert, S., Fokianos, K., Strimmer, K.: Identifying periodically expressed transcripts in microarray time series data. Bioinformatics 20, 5–20 (2004)
8. Liew, A.W.-C., Xian, J., Wu, S., Smith, D., Yan, H.: Spectral estimation in unevenly sampled space of periodically expressed microarray time series data. BMC Bioinformatics 8, 137 (2007)
9. Pittelkow, Y.E., Wilson, S.R.: h-Profile plots for the discovery and exploration of patterns in gene expression data with an application to time course data. BMC Bioinformatics 8, 486 (2007)
10. Pittelkow, Y.E., Rosche, E., Wilson, S.R.: Interpreting models in gene expression data. In: Francis, A.R., Matawie, K.M., Oshlack, A., Smyth, G.K. (eds.) Statistical Solutions to Modern Problems: Proceedings of the 20th International Workshop on Statistical Modelling, Sydney 2005, pp. 381–391 (2005)
11. Cho, R.J., Campbell, M.J., Winzeler, E.A., Steinmetz, L., Conway, A., Wodicka, L., Wolfsberg, T.G., Gabrielian, A.E., Landsman, D., Lockhart, D.J., Davis, R.W.: A Genome-Wide Transcriptional Analysis of the Mitotic Cell Cycle including control of mRNA transcription. Molecular Cell 2, 65–73 (1998)
12. Brockwell, P.J., Davis, R.A.: Time Series: Theory and Methods. Springer, New York (1991)
13. Ahdesmäki, M., Lähdesmäki., H., Pearson, R., Huttunen, H., Yli-Harja, O.: Robust detection of periodic time series measured from biological systems. BMC Bioinformatics 6, 117 (2005)
14. Pittelkow, Y.E., Wilson, S.R.: Visualisation of gene expression data - the GE-biplot, the Chip-plot and the Gene-plot. Stat. Appl. Genet. Mol. Biol. 2, 6 (2003)
15. Chen, J.: Identification of significant periodic genes in microarray gene expression data. BMC Bioinformatics 6, 286 (2005)
16. The GeneCycle Package, http://www.strimmerlab.org/software/genecycle/
17. Everitt, B.S.: Time Series. In: Armitage, P., Colton, T. (eds.) Encyclopedia of Biostatistics, 2nd edn., pp. 5451–5454. Wiley, Chichester (2005)
18. R source code for GE-biplot, http://dayhoff.anu.edu.au/software.html

Microarray Design Using the Hilbert–Schmidt Independence Criterion

Justin Bedo

The Australian National University,
NICTA, and the University of Melbourne

Abstract. This paper explores the design problem of selecting a small subset of clones from a large pool for creation of a microarray plate. A new kernel based unsupervised feature selection method using the Hilbert–Schmidt independence criterion (HSIC) is presented and evaluated on three microarray datasets: the Alon colon cancer dataset, the van 't Veer breast cancer dataset, and a multiclass cancer of unknown primary dataset. The experiments show that subsets selected by the HSIC resulted in *equivalent or better* performance than supervised feature selection, with the added benefit that the subsets are not target specific.

1 Introduction

Feature selection is an important procedure in data mining. The elimination of features leads to smaller and more interpretable models and can improve generalisation performance. Supervised methods produce feature subsets tailored towards the prediction target and are applicable when labels are available. In contrast, unsupervised methods select features that capture some of the information contained within the whole dataset without requiring labels; as no labels are used the feature selections are not target specific.

The problem of designing a sugarcane microarray plate by choosing a subset of approximately 7000 clones from an initial pool of 50,000 clones is studied herein. As the initial pool of clones contains some highly correlated pairs, there is a preference towards *decorrelation*. Furthermore, the array must remain as general as possible and not be tailored towards any specific phenotypes. As such, this is an *unsupervised* selection problem.

The *Hilbert–Schmidt independence criterion* [1] (HSIC) HSIC is a dependence measure between two random variables which is closely related to *kernel target alignment* [2] and *maximum mean discrepancy* [3] (MMD). Previous papers [4,5] used the HSIC for supervised feature selection and demonstrated that the method had good performance and flexibility on several genomics datasets. This paper presents an unsupervised variant named *unsupervised feature selection by the* HSIC (UBHSIC, pronounced ['u.bə-sik]).

UBHSIC was evaluated in several experiments comparing the selection using various kernels to supervised feature selection. As labels are not available on the sugarcane dataset, UBHSIC was evaluated on three cancer genomics datasets: the

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 288–298, 2008.

Alon colon cancer dataset [6,7,8,9], the van 't Veer breast cancer dataset [10], and a multiclass cancer of unknown primary (CUP) dataset [11]. The CUP dataset closely resembles the sugarcane problem as it was intended for the development of a clinical test on a lower resolution platform.

2 HSIC and UBHSIC

The HSIC is a quantity that measures the mutual dependence between two variables. For the task of unsupervised feature selection, the dependence between subsets of features and the full set of features is measured by the HSIC; a subset with *maximum dependence* on the full dataset is desired. This section gives an overview of the HSIC and specifies the unsupervised feature selection problem as a constrained optimisation problem.

Let

$$X := \begin{bmatrix} x_{11} & \cdots & x_{1m} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nm} \end{bmatrix}$$

be a finite dataset in matrix form with $x_{ij} \in \mathbb{R}$, where n is the number of samples and m is the number of features. Each row $\mathbf{x}_{i\cdot}$ corresponds to a sample, and each column $\mathbf{x}_{\cdot j}$ corresponds to a feature.

Let $\theta \in 2^m$ be a subset of features, where 2^m denotes the power set of $\{1, \ldots, m\}$, and define X_θ as the dataset *restricted to only the features* in θ, i.e., the features with indices not in θ are discarded. By this definition, X_θ is a matrix with dimension $n \times |\theta|$, where $|\cdot|$ denotes set cardinality. The dependence between the reduced dataset X_θ and the full dataset X is the quantity we wish to maximise. The HSIC measures this dependence through *kernel functions* [12,13].

A kernel function defines the inner product between two points of a Hilbert space, and can be considered intuitively as a measure of similarity. Indeed, the correlation function $\text{cor}(\mathbf{x}, \mathbf{x}') := \frac{\langle \mathbf{x}, \mathbf{x}' \rangle}{||\mathbf{x}|| ||\mathbf{x}'||}$, where $\langle \cdot, \cdot \rangle$ denotes the inner product and $||\mathbf{x}|| = \sqrt{\langle \mathbf{x}, \mathbf{x} \rangle}$, is a kernel function used in the experiments section. Given a kernel function k, the *kernel matrix* is defined $[K_{ij}]_{1 \leq i,j \leq n} := k(\mathbf{x}_{i\cdot}, \mathbf{x}_{j\cdot})$. The kernel matrix of the full dataset X is referred to as K, and the kernel matrix of the reduced dataset X_θ as K_θ.

An estimator for the HSIC using these two kernel matrices [1] is

$$\text{tr}(K_\theta H K H), \tag{1}$$

where tr is the *matrix trace* (the sum over the elements of the main diagonal), $H := Id - \frac{1}{n}$, Id is the identity matrix, and the subtraction is *element wise*. Using this dependence measure, the unsupervised selection task can simply be stated as

$$\max_\theta \text{tr}(K_\theta H K H) \tag{2}$$

$$\text{such that}$$

$$|\theta| = m'$$

for some $1 \leq m' < m$. Solving this optimisation equation for a set θ gives the UBHSIC solution.

The solution to the optimisation equation is explicit in the linear kernel case where $K := XX^T$. Let $M := HKH$. The HSIC estimator is then

$$\begin{aligned} \mathrm{tr}(KM) &= \mathrm{tr}(XX^T M) \\ &= \mathrm{tr}(X^T MX) \\ &= \sum_j \mathbf{x}_{\cdot j}^T M \mathbf{x}_{\cdot j}. \end{aligned}$$

Thus, in the case of a linear kernel the features are *independent* and can be ranked by $\mathbf{x}_{\cdot j}^T M \mathbf{x}_{\cdot j}$ and greedily selected.

For other kernels, an analytical solution does not exist and a good subset must be found through searching. The forward selection and recursive elimination greedy nested subset strategies [14] can be used to find a good solution if the number of features is not large. This approach was used for the supervised variant presented by Song *et al.* [4,5]. Alternatively, a good solution can be found using combinatorial optimisation algorithms such as simulated annealing. For the sugarcane dataset, a good solution to (2) is found using an annealing algorithm as nested subset selection is unattractive due to the large pool of initial features. Efficient solving the optimisation problem is an open problem.

3 Results and Discussion

The proposed method was analysed on several cancer genomics datasets using different kernels. These kernels are defined as follows:

Radial basis function (RBF): $k(\mathbf{x}, \mathbf{x}') := \exp(-\sigma ||\mathbf{x} - \mathbf{x}'||_2^2)$ with σ set as the inverse median of the squared distances $||\mathbf{x} - \mathbf{x}'||_2^2$ between points in the dataset

Linear: $k(\mathbf{x}, \mathbf{x}') := \langle \mathbf{x}, \mathbf{x}' \rangle$

Polynomial: $k(\mathbf{x}, \mathbf{x}') := (\langle \mathbf{x}, \mathbf{x}' \rangle + 1)^d$ for $d \in \{2, 3\}$

Variance: $k(\mathbf{x}, \mathbf{x}') := \frac{\langle \mathbf{x}, \mathbf{x}' \rangle^2}{\langle \mathbf{x}, \mathbf{x} \rangle \langle \mathbf{x}', \mathbf{x}' \rangle}$

The variance kernel was chosen to produce highly decorrelated selections. The preference towards decorrelation is indirectly encoded as $\langle \mathbf{x}, \mathbf{x}' \rangle / \sqrt{\langle \mathbf{x}, \mathbf{x} \rangle \langle \mathbf{x}', \mathbf{x}' \rangle}$ is the cosine of the angle between the two vectors $\mathbf{x}$ and $\mathbf{x}'$. Thus, as adding a feature highly correlated with another already selected feature will not affect the angle between the vectors as much as a feature orthogonal to all selected features, one may postulate that the kernel used with UBHSIC will produce highly decorrelated selections.

Three cancer genomics datasets were analysed, the van 't Veer breast cancer dataset [10] and a colon cancer dataset [6,7,8,9]. The van 't Veer dataset consists of 98 samples, 46 with a distant metastasis and 52 with no metastasis. Each sample has 5952 dimensions. The colon cancer dataset has 62 samples, 22 normal

and 40 cancerous, and 2000 dimensions per sample. Both datasets are 2-class classification problems.

The final cancer genomics dataset is a cancer of unknown primary (CUP) dataset [11]. This is a multiclass classification dataset where the aim is to develop a predictor for the site of origin of a tumour from a microarray of a sample. The dataset consists of 14 classes, 220 samples, and 9630 features. Not each class is represented equally, with the smallest class containing only 3 samples and the largest containing 34.

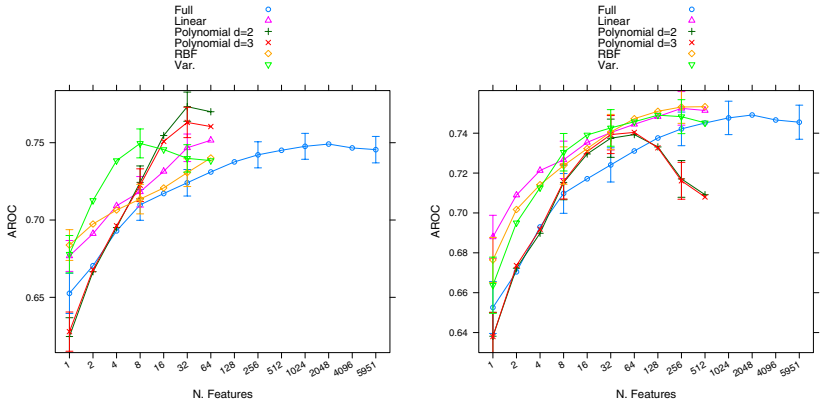
To gauge the utility of feature subsets selected by UBHSIC for prediction, the reduced datasets were evaluated using supervised classification and generalisation estimation. The performance achievable from the reduced datasets were also compared to a fully supervised selection approach.

The classification and supervised feature selection algorithm used was a centroid based classifier and supervised feature selector [15]. This method was chosen as it is simple, fast, and has performed well on these particular datasets [15]. For the multiclass CUP dataset, a one-vs-all architecture [16] was used in conjunction with the centroid classifier to produce a multiclass classifier. For generalisation performance estimation, the ϵ-0 bootstrap estimator [17] was used with 200 repetitions. The *area under the* ROC *curve* (AROC) [18] was used as a performance metric for the two-class datasets. A multiclass extension to the AROC was used [19] for the CUP dataset.

Each dataset was analysed by applying UBHSIC with the various kernels to reduce the full dataset. The centroid classifier and supervised feature selector was then applied to the UBHSIC reduced datasets to evaluate the performance. The same centroid classifier and supervised feature selector was applied to the full dataset to obtain the performance achievable using supervised selection only without any UBHSIC pre-filtering.

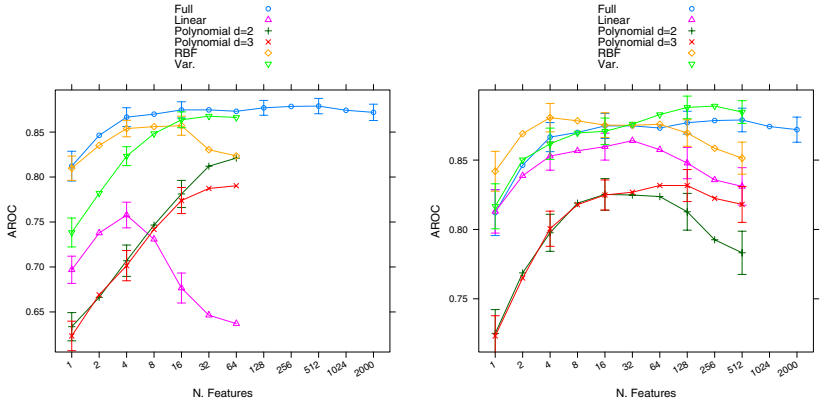
Figure 1 shows the results of pre-filtering using UBHSIC down to 50 (Subfigure 1a) and 500 features (Subfigure 1b) on the van 't Veer dataset. With the reduction to 500 features, the linear, RBF and variance kernels do very well; they achieve a level of performance equivalent to the full dataset at higher numbers of features and exceed the performance at lower numbers of features. The two polynomial kernels initially perform poorly, but after mild supervised feature selection the performance equals that of the other kernels and the full dataset. Under aggressive reduction down to 50 features, somewhat surprising results are obtained; the maximum performance achieved was *substantially better* than the full dataset using a polynomial kernel of degree 2 despite the operating with only 32 features. Furthermore, the variance kernel achieves very high performance at the eight features operating point. Both are significantly fewer than the original 70 genes proposed for classification by the original paper [10].

Performing the same experiments on the colon cancer dataset yielded the results in Figure 2. Again, strong performance when using the variance and RBF kernels is observable in Subfigure 2b; RBF produced very good results after further supervised filtering down to a few features (4) while the variance kernel produced very similar results to the full dataset. The linear and polynomial



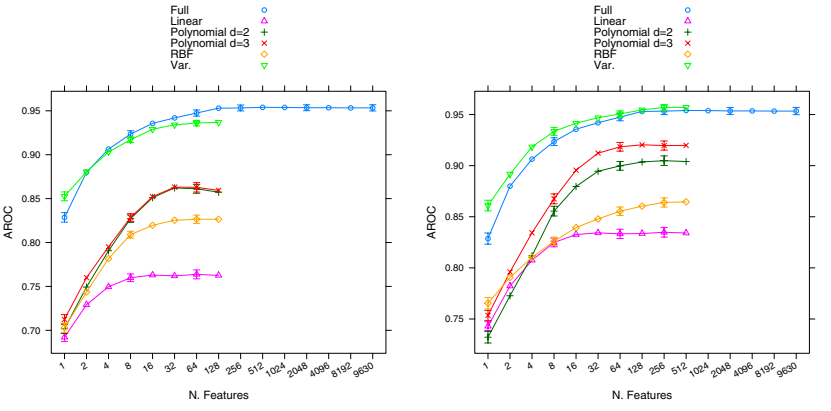
(a) Aggressive reduction to 50 features (b) Reduction to 500 features

Fig. 1. van 't Veer dataset with centroid classifier and feature selector. Results are using the ϵ-0 bootstrap with 200 repetitions. Error bars show 95% confidence interval. Subfigure (a) shows the performance of the dataset reduced to 50 features using the UBHSIC procedure and various kernels. Each plot corresponds to a different kernel, with the purple plot corresponding to the CFS-centroid method on the entire dataset (i.e., without prefiltering using UBHSIC). The 5 plots where prefiltering using UBHSIC was used do not extend above 50 features, and further supervised filtering using the CFS was applied to determine the maximum performance achievable from the reduced datasets. Subfigure (b) is similar to subfigure a, except with less aggressive UBHSIC reduction (reduced to 500 features instead of 50).



(a) Aggressive reduction to 50 features (b) Reduction to 500 features

Fig. 2. Colon cancer dataset with centroid classifier and feature selector. ϵ-0 bootstrap with 200 repetitions. Error bars show 95% confidence interval. The experiment is identical to Figure 1, except with a different dataset.



(a) Reduction to 100 features (b) Reduction to 500 features

Fig. 3. CUP cancer dataset with centroid classifier and feature selector. ϵ-0 bootstrap with 200 repetitions. Error bars show 95% confidence interval. Number of features shown is per class not overall. Experiment details are as in Figure 1.

kernels do not perform well on this dataset; this is supported by the results shown in Subfigure 2a where the linear and polynomial kernels again perform poorly, but the RBF and variance kernels perform well.

Finally, the results of applying the unsupervised feature selection to the CUP dataset is shown in Figure 3. As this dataset is a larger dataset (220 samples) than both the colon and van 't Veer datasets, a less aggressive filtering was applied. Subfigure 3b shows the performance curves obtained after filtering to 500 features. At 500 features, the variance kernel produces a subset with equivalent performance to the full dataset. At the aggressive reduction to 100 features, the performance does not suffer greatly for the variance kernel. The other kernels do not perform well on this dataset.

Furthermore, the 500 feature subset selected by the variance kernel outperformed the full dataset at low numbers of features. The performance achieved below 32 features is greater than the performance at the same operating point obtained with the full dataset. Given this performance, a satisfactory operating point at 16 features or even 8 features per class may be chosen, resulting in a very sparse predictor.

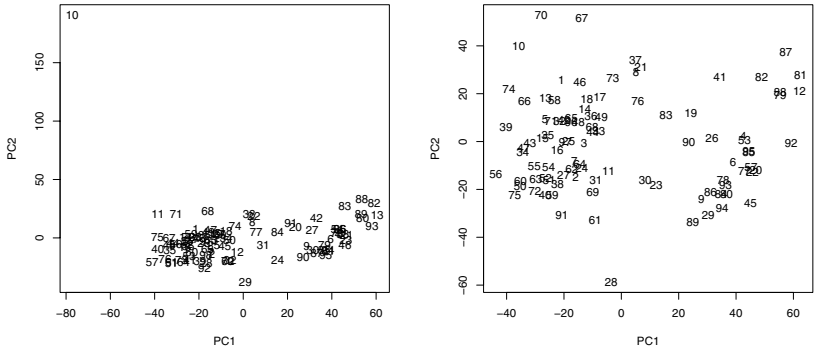
In summary, these results show that unsupervised pre-filtering does not degrade the classification performance and can actually improve the performance at few features. The RBF and variance kernels perform very well across both two-class datasets, with the other kernels not performing as consistently. On the multiclass dataset, the variance kernel is the only kernel that performed well. The aggressive feature reduction down to 50 features for the two-class datasets and 100 features for the CUP dataset showed surprisingly good performance, suggesting that the full datasets contains significant redundancy and can be highly compressed without significant loss of performance.

3.1 van 't Veer in Detail

To gain a better understanding of the relation between features selected by UBHSIC, the feature subsets obtained on the van 't Veer data were visualised. Subfigure 4a shows the full unfiltered dataset projected down onto the first two principal components with each sample represented by a number. It is clear from the projection that sample 10 is an outlier, sitting far away from the other samples. Excluding this sample and reprojecting the data obtains the embedding shown in Subfigure 4b. Here one can observe that the samples roughly form two groups separated mostly by the first principal component (x-axis).

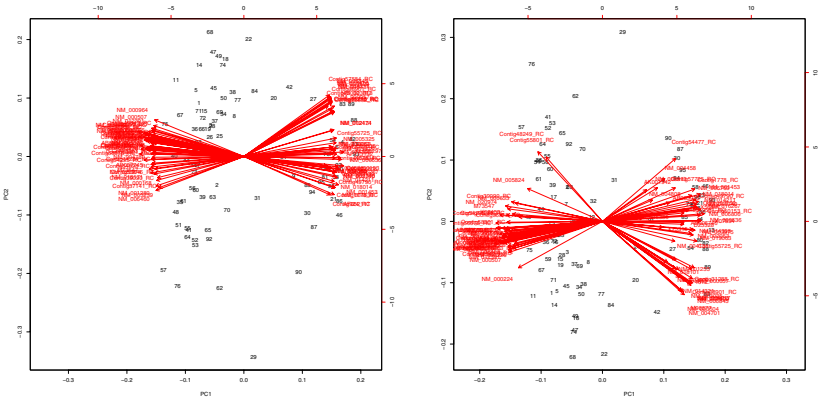
Subfigure 5a displays a biplot [20] of the dataset filtered down to 100 features using the linear kernel and UBHSIC. In the figure, samples are shown as black points and features as red vectors. If two feature vectors have a small angle then they are highly correlated. From the figure the two-group structure observable on the original projection (Subfigure 4b) is maintained. Furthermore, the selected features are strongly positioned along the first principal component. This is not unexpected as a linear kernel is expected to favour the first principal component, and as features are selected independently it is also expected to select highly correlated feature sets. Indeed, a selection of 100 features most correlated with the first principal component yields a subset of features with 77 features in common with the subset selected by the linear kernel and UBHSIC.

The biplot produced using the RBF kernel (Figure 6) resembles the linear kernel results in that the two-group structure is preserved with many features selected along the first principal component. However, in comparison the features are more spread out in two fan-like structures, each spanning one of the groups well, whereas the "fans" formed by the linear kernel are not as spread out and



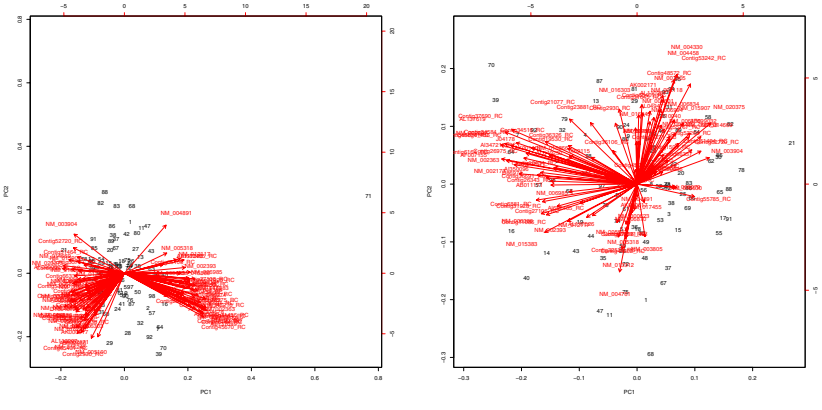
(a) All samples (b) Outlier removed

Fig. 4. Biplot of samples and features projected onto first two principal components using the full van 't Veer dataset. The x-axis is the first principal component, and the y-axis is the second. The sample marked as 10 in subfigure a is clearly an outlier; removing the outlier and reprojecting the samples produces the embedding shown in subfigure b.



(a) Linear kernel (b) RBF kernel

Fig. 5. Biplot after filtering the van 't Veer dataset down to 100 features using the linear and RBF kernels. Both kernels produce selections polarised along the first principal component, though the RBF kernel selections span the samples better than the linear kernel selections.



(a) All samples (b) Outlier removed

Fig. 6. Biplot after filtering the van 't Veer dataset down to 100 features using a polynomial kernel of degree 2. Unlike the linear and RBF kernels, the pattern is more radial, suggesting the selection has less coregulation. With this selection, an outlier is apparent in subfigure (a). Subfigure (b) shows the biplot with the outlier removed.

well aligned with the groups. The RBF kernel is selecting sets with high cross-correlation; this is evident from the number of feature vectors with small interior angles.

Running the same analysis using the polynomial filter of degree 2 yields the results shown in Figure 6. Interestingly, the selected feature subset appears to have

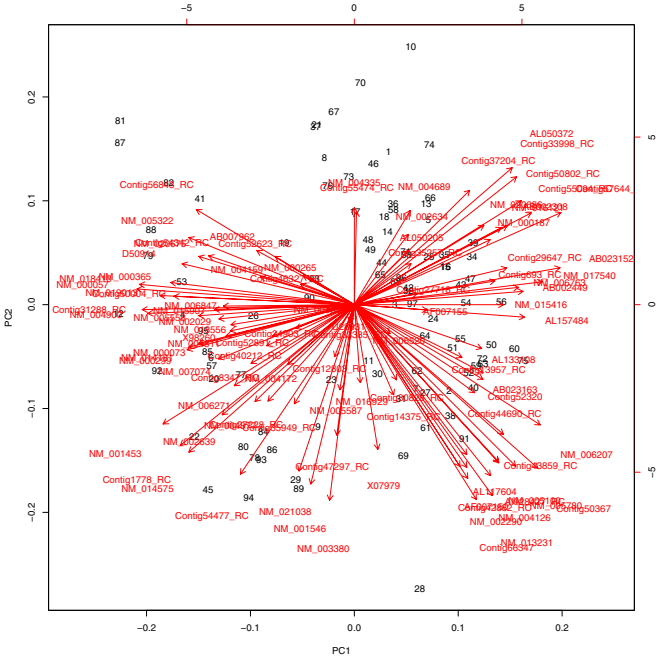


Fig. 7. Biplot after filtering the van 't Veer dataset down to 100 features using the variance kernel. A highly radial pattern is visible, more-so than the polynomial kernel, with no clear outliers.

generated an outlier that is clearly visible in Subfigure 6a; removing this outlier produces a vastly different projection as shown in Subfigure 6b. In this figure the feature vectors can be observed to have a "radial" pattern, indicating the selected features do not have high cross-correlation. Similar results are obtained with the polynomial kernel of degree 3 (not shown). The indication here is that polynomial kernels tend to favour feature subsets with lower cross-correlation than the RBF and liner kernels.

Finally, the variance kernel is shown in Figure 7. Unlike the polynomial kernel, the variance kernel did not produce any new outliers and resulted in a more "radial" pattern than the polynomial filter. This indicates that the selected features are highly decorrelated as postulated previously.

These results indicate the linear and RBF kernels produce subsets with high cross-correlations; the linear kernel is especially highly cross-correlated and aligned with the first principal component while the RBF kernel spans the samples well and is less cross-correlated. The polynomial kernel and variance kernels result in much more decorrelated results, with the variance kernel producing highly decorrelated selections. Given the classification performance observed on the van 't Veer datasets, the RBF and variance kernels are both good choices and can be selected depending if one wishes to obtain whitened data or not.

4 Conclusions

A method for unsupervised feature selection, UBHSIC, was presented and evaluated on several bioinformatics datasets. The results were very promising: on the cancer genomics datasets the classification performance after pre-filtering using UBHSIC was equivalent or better than the performance obtained using the full dataset. The RBF and variance kernels show good performance on all two-class datasets, and the variance kernel showed good performance on the multiclass dataset. Furthermore, the variance kernel producing highly decorrelated selections as postulated.

The high level of classification performance observed after filtering strongly suggests shifting to a lower resolution platform by selecting a subset of clones using the presented method is a viable option. In particular, UBHSIC may be a reasonable solution to the inspiring sugarcane microarray plate design problem. Furthermore, the feature subsets obtained using UBHSIC procedure are not tailored for a specific target and thus may be used to predict many different phenotypes, though further supervised feature selection may be needed to reach the maximum performance.

Acknowledgment

I acknowledge the permission of NICTA to publish this paper. NICTA is funded by the Australian Governments Department of Communications, Information Technology and the Arts and the Australian Council through Backing Australias Ability and the ICT Centre of Excellence programs. I thank Adam Kowalczyk, Geoff Macintyre, and Alex Smola for helpful discussions and help in preparing this manuscript.

References

1. Gretton, A., Bousquet, O., Smola, A., Schölkopf, B.: Measuring statistical dependence with hilbert-schmidt norms. In: Algorithmic Learning Theory: 16th International Conference (January 2005)
2. Cristianini, N., Shawe-Taylor, J.: On kernel-target alignment. Neural Information Processing Systems 14 (January 2002)
3. Gretton, A., Borgwardt, K., Rasch, M., Schölkopf, B., Smola, A.: A kernel method for the two-sample-problem. Advances in Neural Information Processing Systems (January 2007)
4. Song, L., Bedo, J., Borgwardt, K., Gretton, A., Smola, A.: Gene selection via the bahsic family of algorithms. Bioinformatics (January 2007)
5. Song, L., Smola, A., Gretton, A., Borgwardt, K., Bedo, J.: Supervised feature selection via dependence estimation. In: Proceedings of the 24th international conference on Machine Learning (January 2007)
6. Alon, U., Barkai, N., Notterman, D., 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 (January 1999)

7. 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)
8. Guyon, I., Weston, J., Barnhill, S., Vapnik, V.: Gene selection for cancer classification using support vector machines. Machine Learning 46, 389–422 (2002)
9. Huang, T., Kecman, V.: Gene extraction for cancer diagnosis by support vector machines—an improvement. Artificial Intelligence In Medicine (January 2005)
10. van 't Veer, L., Dai, H., van de Vijver, M.J., He, Y.D., Hart, A.A.M., Mao, M., Peterse, H.L., van der Kooy, K., Marton, M.J., Witteveen, A.T., Schreiber, G.J., Kerkhoven, R.M., Roberts, C., Linsley, P.S., Bernards, R., Friend, S.: Gene expression profiling predicts clinical outcome of breast cancer. Nature 415(6871), 530–536 (2002)
11. Tothill, R.W., Kowalczyk, A., Rischin, D., Bousioutas, A., Haviv, I., van Laar, R.K., Waring, P.M., Zalcberg, J., Ward, R., Biankin, A., Sutherland, R.L., Henshall, S.M., Fong, K., Pollack, J.R., Bowtell, D., Holloway, A.J.: An expression-based site of origin diagnostic method designed for clinical application to cancer of unknown origin. Cancer Res. 65(10), 4031–4040 (2005)
12. Berlinet, A., Thomas-Agnan, C.: Reproducing Kernel Hilbert Spaces in Probability and Statistics. Springer, Heidelberg (2003)
13. Schölkopf, B., Smola, A.J.: Learning with Kernels. MIT Press, Cambridge (2002)
14. Guyon, I.: An introduction to variable and feature selection. JMLR 3, 1157–1182 (2003)
15. Bedo, J., Sanderson, C., Kowalczyk, A.: An efficient alternative to svm based recursive feature elimination with applications in natural language processing and bioinformatics. In: Proceedings of the Australian Joint Conference on Artifical Intelligence (2006)
16. Rifkin, R., Klautau, A.: In defense of one-vs-all classification. The Journal of Machine Learning Research (January 2004)
17. Efron, B.: How biased is the apparent error rate of a prediction rule? Journal of the American Statistical Association (January 1986)
18. Hanley, J.A., McNeil, B.J.: The meaning and use of the area under a receiver operating characteristic (roc) curve. Radiology 143(1), 29–36 (1982)
19. Hand, D., Till, R.: A simple generalisation of the area under the roc curve for multiple class classification problems. Machine Learning (January 2001)
20. Gabriel, K.: The biplot graphic display of matrices with application to principal component analysis. Biometrika (January 1971)

Identifying Non-random Patterns from Gene Expression Profiles

Radhakrishnan Nagarajan[1], Meenakshi Upreti[2], and Mariofanna Milanova[3]

[1] Department of Biostatistics
rnagarajan@uams.edu
[2] Department of Biochemistry and Molecular Biology,
University of Arkansas for Medical Sciences, USA
mupreti@uams.edu
[3] Department of Computer Sciences, University of Arkansas at Little Rock, USA
mgmilanova@ualr.edu

Abstract. There has been considerable interest in identifying biologically relevant genes from temporal microarray gene expression profiles using linear and nonlinear measures. The present study uses two distinct approaches namely: classical order zero-crossing count (ZCC) and Lempel-Ziv (LZ) complexity in identifying non-random patterns from temporal gene expression profiles. While the former captures the linear statistical properties of the time series such a power-spectrum, the latter has been used to capture nonlinear dynamical properties of gene expression profiles. The results presented elucidate that ZCC can perform better than LZ in identifying biologically relevant genes. The robustness of the findings are established on the given gene expression profiles as well as their noisy versions. The performance of these two techniques is demonstrated on publicly available yeast cell-cycle gene expression data. A possible explanation for the better performance of the ZCC over LZ complexity may be attributed to inherent cyclic patterns characteristic of the yeast cell-cycle experiment. Finally we discuss the biological relevance of new genes identified using ZCC not previously reported.

Keywords: Gene expression, Time series, Zero-crossing count, Lempel-Ziv complexity.

1 Introduction

Microarrays have proven to be useful tools in capturing simultaneous expression of a large number of genes in a given paradigm. These high-throughput assays provide system-level understanding of the given paradigm. Recently, microarrays have been used to generate temporal expression profiles. Such profiles capture the transcriptional activity of genes as a function of time, hence their dynamics. There has been considerable interest in developing appropriate techniques to understand functional relationships and network structures from temporal gene expression profiles (see [1] and references therein). These include global analysis techniques such as PCA, hierarchical clustering and self-organizing maps (see [2] and references

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 299–310, 2008.

therein). As pointed out by [3], such techniques treat the expression across the time points as independent entities, hence immune to the temporal structure/dynamics. Such global analyses may also be susceptible to inherent transcriptional delays. Therefore, it might be necessary to explore alternate techniques that are sensitive to the temporal structure of the data. Temporal expression profiles of biologically relevant genes orchestrating a specific paradigm follow characteristic and reproducible patterns. This in turn supports their inherent non-random nature. While genetic networks are undoubtedly nonlinear dynamical systems, these dynamical nonlinearities need not necessarily manifest in the external recording such as microarray expression profiles. This can be attributed to noisiness and nonlinearities at the measurement and dynamical levels [4]. On a related note, nonlinear dynamical systems can also give rise to simple cyclic/periodic behavior (limit cycle) for certain choice of the system parameters which can be modeled as linear stochastic processes. Therefore, measures sensitive to linear as well as nonlinear statistical properties have been used to investigate patterns in gene expression profiles. In the present study, we compare the performance of two measures in identifying biologically relevant genes, namely. (a) *zero-crossing count* (ZCC), [5] which is related to the linear statistical properties of temporal data such as power-spectrum and (b) *Lempel-Ziv (LZ) complexity measure*, [6,7] which is sensitive to linear as well as nonlinear dynamical properties in the given data. We show that ZCC can prove to be a better choice than LZ complexity in identifying biologically significant genes. The better performance of ZCC may be due to inherent cyclic patterns in the yeast cell-cycle data. Such patterns can be modeled as linear stochastic processes with Gaussian innovations. Therefore, measures sensitive to the linear correlation structure may be sufficient to describe them.

Techniques such as power-spectral analysis which capture the linear statistical properties of the stationary time series are a natural choice for investigating experimental time series [8]. Power-spectrum is related to the auto-correlation function (Wiener-Khinchin theorem) which in turn is used to estimate the optimal process parameters of linear stochastic processes (Yule-Walker equations) [9, 10]. Alternatively, auto-correlation function is sufficient statistics for describing normally distributed linear stochastic processes. Uncorrelated noise is characterized by a flat power-spectrum representing equal power across all frequencies. A significant skew in the power-spectrum towards lower-frequencies is indicative of correlation or non-random signatures in the given time series. These in turn may be indicative of biologically relevant genes. Classical spectral estimation may not be possible due to the small length of the temporal gene expression profiles. However, the spectral properties of a given time series can also be captured by zero-crossing count [5]. The latter overcomes some of the caveats encountered in classical spectral estimation and its interpretation is fairly straight forward. In the present study, biologically relevant genes will be identified from ZCC. In order to establish statistical significance, ZCC estimates on the given data are compared to those obtained on random shuffled surrogates. The random shuffled surrogates represent uncorrelated counterpart of the given data.

Information theoretic approaches have also been popular in capturing patterns in gene expression profiles. These include entropy-based approaches [11]. However, entropy estimates are governed solely by the probability distribution of the expression values, hence immune to the temporal expression profile. For instance a gene exhibiting a periodic pattern across 9 time points (001001001) has the same Shannon

entropy as gene exhibiting seemingly random pattern (101000100). The latter is obtained by randomly shuffling the former. Complexity measures which overcome some of the caveats of entropy have recently been proposed to investigate gene expression profiles [12, 15, 16]. Traditionally, uncorrelated noise is considered to be maximally complex or random. Any deviation from randomness ensures correlation and accompanied by a decrease in complexity. From the perspective of gene expression analysis, genes with low complexity are hypothesized to be biologically relevant. A recent study, proposed several measures of complexity for the analysis of temporal gene expression profiles in yeast cell-cycle experiment [12]. Such an analysis was carried out in an unbiased manner in the absence of any prior knowledge about the given data. The authors successfully identified biologically relevant genes in addition to those that exhibited characteristic cyclic behavior [13, 14]. Complexity measures have also been successfully used to gain insight into the dynamical aspects of genetic networks from the temporal expression profiles [15, 16]. In [15], the distribution of the Lempel-Ziv complexity from experimental gene expression profiles was compared to those generated from synthetic random Boolean networks (RBN) using Kullback-Leibler divergence and Euclidean distance. Subsequently, the dynamics of the genetic network governing the paradigm was found to lie in the ordered regime or between order and chaos. More recently, a variant of the Kolmogorov-complexity (i.e. normalized compression distance) was used to argue in favor of criticality in macrophage dynamics. In the present study, Lempel-Ziv (LZ) complexity is used identify biologically relevant genes. Unlike ZCC, LZ is sensitive to linear as well as nonlinear correlations in the given data. For instance, qualitative behavior of LZ complexity has been found to mirror invariants such Lyapunov exponents in nonlinear dynamical systems [7, 17]. Periodic time series such as sine-waves are highly compressible and result in low values of LZ complexity. Ideally, uncorrelated noise cannot be compressed hence accompanied by large values of LZ complexity. As in the case of ZCC, LZ values obtained on the given data are compared to those obtained on random shuffled surrogates in order to establish statistical significance. There is no direct relation between LZ complexity and ZCC. However, it should be noted that ZCC as well LZ is likely to increase monotonically with increasing noise in the given data. Noise being reflected by high-frequency components in the power-spectrum.

2 Methods

2.1 Spectral Analysis by Zero-Crossing

Consider a zero-mean normally distributed stationary process $x_t, t = 1...N$. The corresponding binary sequence of the differenced series $z_t = x_t - x_{t-1}$ is generated as

$$\begin{aligned} y_t &= 1 \quad \text{if} \quad z_t > 0 \\ &= 0 \quad \text{otherwise} \end{aligned} \tag{1}$$

Expression (1) essentially quantizes the given input signal z_t onto a coarse-grained binary sequence. The sequence y_t is subsequently differenced and passed through a memoryless nonlinear transform as

$$\nabla_t = (y_t - y_{t-1})^2 \tag{2}$$

A zero-crossing is said to occur if $\nabla_t = 1$. Subsequently, the zero-crossing count is given by

$$D_1 = \sum_{t=1}^{N-1} \nabla_t \tag{3}$$

The zero-crossing count is related to the linear correlation, hence the power-spectrum [see 5 and references therein]. For certain class of colored Gaussian noise, zero-crossing analysis may be sufficient to describe the process dynamics [5]. Examples include cyclic patterns such as those encountered in the Spellman alpha-factor synchronization yeast cell-cycle experiment. At this point, it might be necessary to point out certain resemblance between the zero-crossing analysis and the complexity maps proposed in [12]. The first-order crossing $\delta_t = y_t - y_{t-1}$ shares resemblances to the map γ_{Δ_1} proposed recently in [12]. On a similar note the maps γ_{Δ_2} and γ_{Δ_3} in [12] share resemblance to the expressions δ_t^2 (i.e.$\delta(\delta_t) = \delta y_t - \delta y_{t-1}$) and δ_t^3 (i.e.$\delta(\delta_t^2)$) for higher order crossings obtained repeated application of the difference operator (high-pass filter) [5]. On closer observation, we note subtle differences in their definitions. The complexity maps [12] use (*i*) ranks of the values as opposed to the sign of the mean-subtracted values (1). (*ii*) Discontinuous memoryless function as opposed to continuous memoryless function (2). While the maps in [12] capture the complexity of the given process, ZCC captures the spectral characteristics of the process.

2.2 Lempel-Ziv Complexity

Lempel-Ziv complexity (LZ) [6] and its extensions [7] have been used widely to understand the dynamics in biomedical [18, 19] and genomic signals [15]. An elegant implementation of the LZ algorithm for binary sequences was proposed in [7]. In the present study, binary sequence of the expression profiles was generated by thresholding about the mean. The objective of the LZ algorithm is to reconstruct the given sequence *s* of length *n*, using two fundamental operations, namely: *copy* and *insert* by parsing it from left to right. This information in turn is used for estimating the algorithmic complexity of that. The working principle of the LZ algorithm is shown below for completeness. Prior to the discussion of the example we introduce the notation $v(s)$ in the following example corresponds to the vocabulary set [6]. Consider $s = 00$, then v(s) represents all possible words that can be reconstructed from s when scanning from left to right, i.e. $v(s) = \{0, 00\}$. It is important to note that while 0 can be generated from $v(s)$, 1 cannot be generated from $v(s)$.

Consider a period 3 sequence s_0 = 001001001……. as before

(a) The first digit 0 is unknown hence have to be inserted, resulting in $c(n) = 1$ and $s^* = 0$.

(b) Consider the second digit 0. Now $s = 0$, $q = 0$; $sq = 00$; $sq\pi = 0$; $q \in v(sq\pi)$; therefore copying is sufficient resulting in no change in the complexity i.e. $c(n) = 1$ and $s^* = 0.0$

(c) Consider the third digit 1. Now s = 0, q = 01; sq = 001; $sq\pi$ = 00; $q \notin v(sq\pi)$; therefore insertion is required, resulting in $c(n)$ = 2 and s^* = 0.01.
(d) Consider the fourth digit 0: s = 001; q = 0; sq = 0010; $sq\pi$ = 001; $q \in v(sq\pi)$; therefore copying is sufficient, resulting in $c(n)$ = 2 and s^* = 0.01.0
(e) Consider the fifth digit 0: s = 001; q = 00; sq = 00100; $sq\pi$ = 0010; $q \in v(sq\pi)$; therefore copying is sufficient, resulting in $c(n)$ = 2 and s^* = 0.01.00
(f) Consider the sixth digit 1: s = 001; q = 001; sq = 001001; $sq\pi$ = 00100; $q \in v(sq\pi)$; therefore copying, is sufficient resulting in $c(n)$ = 2 and s^* = 0.01.001

It is important to note that subsequent addition of symbols from s does not change $c(n)$. This can be attributed to the inherent periodicity of the sequence s. Since s^* does not end in a dot (.) we add one to the complexity, resulting in $c(n)$ = 3. In the present study, we consider the normalized complexity measure (γ) given by the expression

$$\gamma = \frac{c(n)}{b(n)}, \text{ where } b(n) = \frac{n}{\log_2 n} \tag{4}$$

The normalized complexity (γ) tends to unity in the asymptotic limit for sequences whose Shannon entropy is unity [6, 7].

2.3 Random Shuffled Surrogates

Resampling techniques are encouraged in literature for establishing statistical significance where the null distribution is unknown. Resampling without replacement is used widely within the context of correlated data analysis [20, 21]. Such an approach retains certain statistical properties of the given empirical sample in the surrogates. For the same reason these surrogates are termed as *constrained realizations* [21]. In the present context, the objective is to argue in favor of correlation in the given gene expression profile, i.e. the statistics considered namely: D_1 (3) and γ (4) are sensitive to correlation in the given data. Therefore, the objective is the reject the null that the given data is uncorrelated noise. The surrogates under the above null are generated by randomly shuffling (RS) the temporal expression profile of that gene. The constraint here is on retaining the distribution of the gene expression profile in the surrogates. Alternatively, the distribution of the gene expression profile is treated as a nuisance variable [21]. The discriminant statistics (3) and (4) are sensitive to the temporal structure, hence are expected to exhibit a significant discrepancy between the empirical sample and the surrogate counterpart in the case of correlated expression profiles. More importantly, estimates of (3) and (4) on the surrogate realizations will be higher than those estimated on the empirical sample for correlated gene expression profiles. Therefore, a one-sided test is sufficient to establish statistical significance. In the present study, the number of surrogates were chosen as (n_s = 99) corresponding to a significance level α = 1/(99+1) = 0.01 [21]. It is important reiterate that the null is rejected only when the estimate of (3) and (4) is lesser than each of the 99 surrogate realizations.

3 Data

In order to establish reproducibility and direct comparison, we have used the same data sets as those investigated recently by [12] using a battery of complexity measures. The final list of genes was obtained from the Ahnert et al., (personal communication). The authors in [12] identified 150 genes as top-ranked by one of their proposed complexity measures $k(f/\gamma_{\Delta 3})$. 52 of these 150 genes were listed as biologically relevant either by Spellman (104 genes, *http://genome-www.stanford.edu /cellcycle/data/rawdata/-Knowngenes.doc*) or Simon (140 genes, *http://web.wi.mit. edu/young/cellccycle/-Table of Regulated genes*). Their set of 150 genes also included genes not listed in either Spellman (104) or Simon (140).

LIST OF 133 GENES: Since identifying the missing data is not one of the objectives of the proposed study, we first eliminate all genes whose values are missing even at a single time point in the Spellman alpha-factor synchronization yeast cell-cycle experiment (6178 genes across 18 time points) [14]. The reduced set consisted of 4491 genes. Therefore, all subsequent discussions will be restricted to this set of 4491 genes. Spellman et al., 1998 [14] identified 104 genes as well-documented through extensive literature survey. Out of these 104 genes, 72 genes had values across all 18 time points (i.e. overlapped with the list of 4491 genes). The ORFs of these 72 genes were subsequently identified. In a related study, Simon et al., 2001 [13] identified 140 genes as being relevant to yeast cell-cycle. The gene IDs of these 140 genes were retrieved by comparing their ORFs to those spotted on the array. Only 125 of the 140 identified had gene IDs. Out of these 125 genes, 93 genes had values across all 18 time points (i.e. overlapped with the set of 4491 genes). Thus in essence we have 72 out of the 104 genes from study [14] and 93 out of the 140 genes from study [13]. The union of the above sets resulted in 133 genes relevant to yeast cell-cycle.

LIST OF 40 GENES: The performance of the measures (3) and (4) are also determined using the set of 40 genes as ground truth from Ahnert et al., (personal communication). Of the 52 genes identified by Ahnert et al., 2006 40 had values across all 18 time points (i.e. overlapped with the 4491 genes).

The objective of the present study is to determine the effectiveness of the measures (3) and (4) on retrieving biologically relevant genes from the set of 4491 genes. Of interest is to investigate the false-positive and false-negative rates using the 133 genes and 40 genes as ground truths. The effect of noise on the performance of the two measures (3), (4) is also investigated. Finally the usefulness of the ZCC (3) in identifying new genes not previously reported in either Simon [13] or Spellman [14] is discussed. The temporal expression profiles of the 4491 genes were mean-subtracted prior to the analysis. The present study also assumes the gene expression profiles to be generated from stationary stochastic processes.

4 Results

Prior to investigating the gene expression profiles we demonstrate the effectiveness of the proposed measures in quantifying regularity in patterns across synthetic linear and nonlinear dynamical processes.

4.1 Synthetic Data

Periodic sine-wave

Consider a periodic sine-wave given by $s(t) = \sin(2\pi f_1 \omega)$ with (f_1 = 2, N = 200), Fig 1a. The sampling frequency (F_s = 15) was chosen so as to satisfy the Nyquist criterion, i.e. $F_s > 2f_1$. Periodic signatures can be modeled as linear stochastic processes such as auto-regressive moving average processes (ARMA), hence linear statistical properties are sufficient to describe them. However, periodic signatures can also be generated by nonlinear dynamical systems (limit cycles). The power-spectrum of the periodic sine-wave exhibits a dominant frequency, Fig. 1b. The results of the zero-crossing analysis (3) and the normalized complexity (4) are shown in Fig. 1c and 1d respectively. The statistical measures (3) and (4) of the periodic sine-wave is clearly lesser than those obtained on the random shuffled surrogates (n_s = 99, α = 0.01) rejecting the null as expected, Figs. 1c and 1d.

Chaotic logistic map

Consider a chaotic logistic map given by $x_{n+1} = 3.8.x_n.(1\text{-}x_n)$, Fig 1e. Unlike periodic sine-wave, chaotic logistic map is nonlinearly correlated and is accompanied by a broad-band power-spectrum characteristic of random noise, Fig. 1f. The results of zero-crossing analysis (3) and the normalized complexity (4) are shown in Fig. 1g and 1h respectively. The time series was mapped onto a binary sequence by thresholding about the superstable fixed point (0.5). This particular choice has been shown to capture the dynamics faithfully [17]. Zero-crossing estimate (3) on the chaotic process was considerably higher than those on the random shuffled surrogates failing to reject

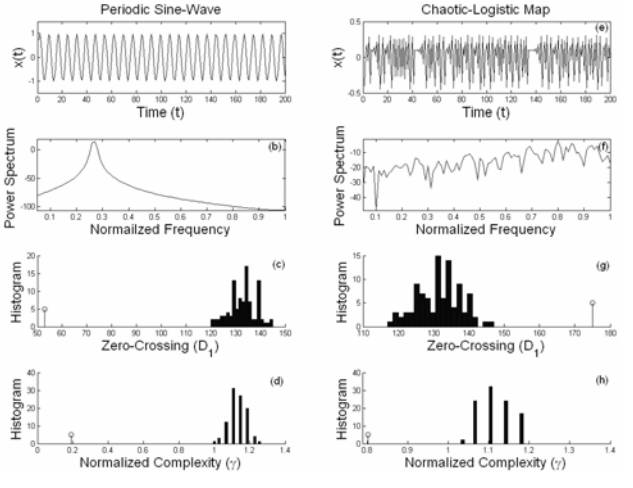


Fig. 1. Time series generated from periodic sine-wave (left) and chaotic logistic map (right) are shown in (a) and (e) respectively. The corresponding power-spectra are shown in (b) and (f) respectively. Zero-crossing crossing analysis (c and g) and normalized complexity (d and h) estimates of the empirical samples are shown by hollow circles. The histogram of their estimates on (n_s = 99) random shuffled surrogates are also shown in the corresponding subplots (black bars).

the null (n_s = 99, α = 0.01), Fig. 1g that the given process is uncorrelated noise. The one-step auto-correlation estimated using (7) and directly from the data was negative failing to provide any meaningful insight into the dynamics. However, analysis using the normalized complexity clearly rejected the null, Fig. 1h. Thus the zero-crossing analysis may have clear limitations when analyzing from nonlinear processes.

4.2 Yeast Cell-Cycle Experiment

Zero-crossing analysis (3) of the 4491 genes (Sec. 3) identified 101 genes as being significant (α = 0.01). A similar analysis using normalized complexity (4) identified 133 genes as being significant (α = 0.01). Prior to a detailed analysis we investigated two genes namely: YBR010W (Fig. 2a) and YPR150W (Fig.2b) using D_1 and γ. The choice of these two genes can be attributed to a recent study which investigated their biological relevance using a battery of complexity measures [12]. Visual inspection of the temporal expression profiles of YBR010W (Fig. 2a) revealed characteristic low-frequency non-random signatures unlike YPR150W (Figs. 2b). Zero-crossing (D_1), Fig. 2c, as well as normalized complexity (γ), Fig. 2e, estimates on YBR010W were considerably less than those obtained on the surrogate realizations rejecting the null that YBR010W is uncorrelated noise. The results of a similar analysis of YPR150W using (D_1) and (γ) are shown in Figs. 2d and 2f respectively. Both the measures failed to reject the null in the case of YPR150W. Earlier studies have reported YBR010W (or HHT1) to be actively involved in cell-cycle [13]. In contrast, YPR150W is an open-reading frame (ORF) with no documented evidence of its role in yeast cell-cycle. The performance of the two measures (D_1, γ) were subsequently investigated by introducing noise in the zero-mean temporal expression profile for a gene x as

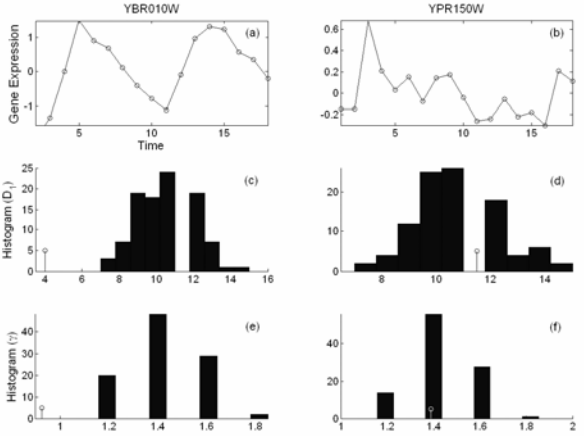


Fig. 2. Temporal expression profiles of two genes YBR010W and YPR150W are shown in (a) and (b) respectively. The corresponding zero-crossing (D_1) estimates (circle) along with the distribution of the estimates on the 99 random shuffled surrogates (black bars) for each of the three genes are shown right below them in (c) and (d) respectively. The results of a similar analysis using normalized complexity (γ) for the genes are shown right below them in (e) and (f) respectively.

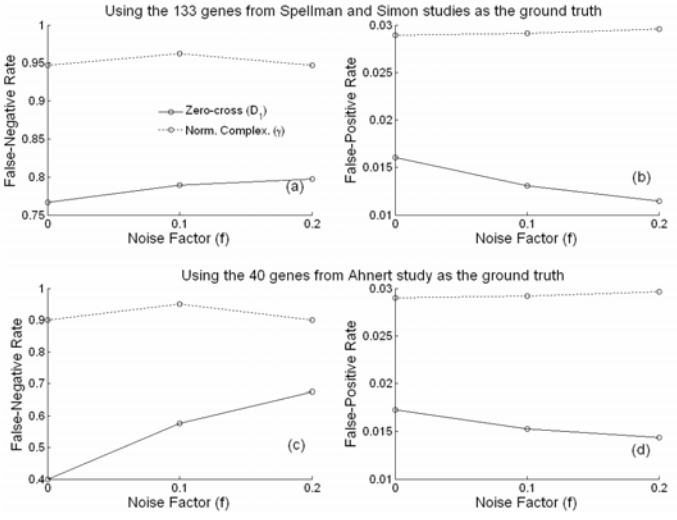


Fig. 3. False-negative rate and false positive rate obtained assuming the 133 genes [13,14] as ground truth using zero-crossing (D_1, solid lines) and the normalized complexity (γ, dotted lines) for noise factors (f = 0, 0.10, and 0.20) is shown in (a) and (b) respectively. False-negative rate and false-positive rate obtained by a similar analysis assuming the 40 genes [12] as ground truth is shown in (c) and (d) respectively.

($y_t = x_t + f.e_t$), where e_t is i.i.d Gaussian noise. Introducing noise to the observed expression falls under measurement noise ubiquitous in microarray studies.

Assuming the 133 genes identified by the union of 140 genes and 104 genes [13, 14] whose values are known across all time points as the ground truth.
Zero-crossing analysis (D_1) identified 37 of the 133 biologically relevant genes (i.e. 31/133 ~ 23%). The normalized complexity (γ) identified only 7 out of the 133 genes (i.e. 7/133 ~ 5%). These have to be compared to those of [12], who identified 52 genes from an union set of 195 genes (52/195 ~ 26.7%) using complexity measure $k(f/d3)$. The discrepancy in the number of genes between [12] and the present study (133) can be attributed to the fact that we considered only genes whose expression values are known across all time points. The false-positive rate (FPR) and false-negative rate (FNR) for the normalized complexity (γ) and zero-crossing analysis (D_1) across noise factors (f = 0, 0.10, 0.20) is shown in Figs. 3a and 3b respectively. From Fig. 3a it is evident that the FPR and FNR of the normalized complexity are considerably higher than that of the zero-crossing analysis.

Assuming the 40 genes from (52) [12] whose values are known across al the time points as the ground truth.
It should be noted that these 40 genes are present in the union set of 133 genes discussed above. While zero-crossing analysis (D_1) identified (24/40) genes, the normalized complexity (γ) identified only (4/40) genes from [12]. The FPR and FNR for the two measures assuming these 40 genes as the ground truth across noise factors (f = 0, 0.10, 0.20) is shown in Figs. 3c and 3d respectively. The results are similar to

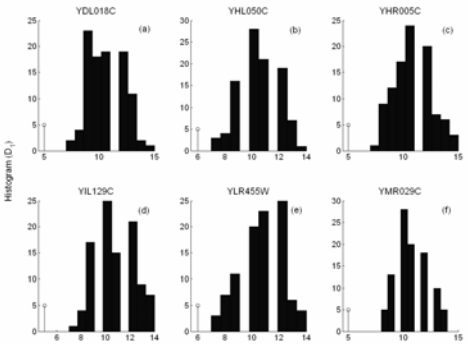


Fig. 4. Six genes YDL018C, YHL050C, YHR005C, YIL129C, YLR455W and YMR029C identified by D_1 and not present in the list of 133 genes. The estimate of D_1 on the gene expression profile (circle) and their ($n_s = 99$) random shuffled surrogates (black bars) are shown in each of the subplots.

those obtained for the 133 genes with considerably higher FNR and FPR for (D_1) compared to γ.

New genes discovered by the zero-crossing analysis

As noted earlier, out of the 101 genes detected as being statistically significant by zero-crossing analysis, 37 exhibited overlap with the documented 133 genes. The remaining 64 genes consisted of YDL018C (ERP3) [22], YHL050C, YHR005C (GPA1) [23], YIL129C (TAO3) [24, 25], YLR455W (Uncharacterized ORF) [26], YMR029C (FAR8) [27], see Fig. 4. YDL018C or ERP3 had been shown to be involved in the G1/S phase of the cell cycle [22]. YHL050C is an uncharacterized open-reading frame but we believe its non-random nearly periodic pattern is compelling for us to hypothesize it as a likely candidate in yeast cell cycle regulation. There has been evidence [23] on the role of YHR005C (GPA1, G-protein alpha subunit 1) on the mating-factor mediated cell cycle arrest. YIL129C (TAO3) is a member of the RAM (Regulation of Ace2p activity and cellular Morphogenesis) [24] signaling network which governs cell separation, integrity and progression [25]. There has been evidence of YLR455W being involved in S-phase of the cell-cycle [26]. YMR029C (FAR8) has been documented to be involved in G1 cell cycle arrest in response to pheromone [27].

5 Discussion

There has been considerable interest in understanding temporal gene expression profiles using suitable techniques. Biologically relevant genes are hypothesized to exhibit non-random and reproducible temporal expression profiles. Understanding the correlation in these patterns has been of great interest. The present study investigated two distinct measures namely: zero-crossing count and the normalized Lempel-Ziv complexity in identifying biologically relevant genes from their expression profile. While the former is sensitive to only the linear correlation, the latter is sensitive to

linear as well as nonlinear correlations in the given data. These techniques implicitly assume that the given temporal expression profiles are sampled from a stationary process. From the results presented, it is evident that zero-crossing count which mimics the spectral content of the given data may prove to be useful in determining biologically relevant genes from their expression patterns. Its performance was found to be better than that of the normalized complexity. The results were demonstrated on yeast cell cycle expression profiles with and without noise. The better performance of the zero-crossing count may be attributed to cyclic patterns which can be modeled as linear stochastic processes.

Acknowledgments. R.N. was partially supported by NIH Grant # P20 RR-16460 from the IDeA Networks of Biomedical Research Excellence (INBRE) Program of the National Center for Research Resources. R.N would like to thank Xiaoyan (Iris) Leng for useful comments on gene annotation and Sebastian Ahnert for sharing the list of genes identified in his study.

References

1. Bar-Joseph, Z.: Analyzing time series gene expression data. Bioinformatics 20(16), 2493–2503 (2004)
2. Butte, A.: The use and analysis of microarray data. Nat. Rev. Drug Discov. 1(12), 951–960 (2002)
3. Leng, X., Müller, H.G.: Classification using functional data analysis for temporal gene expression data. Bioinformatics 22(1), 68–76 (2006)
4. Nagarajan, R., Aubin, J.E., Peterson, C.A.: Modeling genetic networks from clonal analysis. Journal of Theoretical Biology 230(3), 359–373 (2004)
5. Kedem, B.: Time Series Analysis by Higher Order Crossings. IEEE Press, Los Alamitos (1994)
6. Lempel, A., Ziv, J.: On the Complexity of Finite Sequences. Information Theory, IEEE Transactions 22(1), 75–81 (1976)
7. Kaspar, F., Schuster, H.G.: Easily calculable measure for the complexity of spatio-temporal patterns. Phys. Rev. A. 36, 842–848 (1987)
8. Butte, A.J., Bao, L., Reis, B.Y., Watkins, T.W., Kohane, I.S.: Comparing the similarity of time-series gene expression using signal processing metrics. J. Biomed. Inform. 34(6), 396–405 (2001)
9. Proakis, J.G., Manolakis, D.G.: Digital Signal Processing, Principles Algorithms and Applications. Prentice-Hall, Englewood Cliffs (1996)
10. Papoulis, A., Pillai, S.U.: Probability, Random Variables and Stochastic Processes, 4th edn. McGraw-Hill, New York (2002)
11. Fuhrman, S., Cunningham, M.J., Wen, X., Zweiger, G., Seilhamer, J.J., Somogyi., J.: The Application of Shannon Entropy in the Identification of Putative Drug Targets. Biosystem 55(1-3), 5–14 (2000)
12. Ahnert, S.E., Willbrand, K., Brown, F.C.S., Fink, T.M.A.: Unbiased pattern detection in microarray data series. Bioinformatics 22, 1471–1476 (2006)
13. Simon, I., et al.: Serial regulation of transcriptional regulators in the yeast cell cycle. Cell 106, 697–708 (2001)

14. Spellman, P.T., et al.: Comprehensive identification of cell cycle-regulated genes of the yeast Saccharomyces cerevisiae by microarray hybridization. Mol. Bio. Cell 9, 3273–3297 (1998)
15. Shmulevich, I., Kauffman, S.A., Aldana, M.: Eukaryotic cells are dynamically ordered or critical but not chaotic. Proc. Nat. Acad. of Sci (USA) 102(38), 13439–13444 (2005)
16. Nykter, M., Price, N.D., Aldana, M., Ramsey, S.A., Kauffman, S.A., Hood, L., Yli-Harja, O., Shmulevich, I.: Gene Expression Dynamics in the Macrophage Exhibit Criticality. Proc. Nat. Acad. of Sci (USA) 105(6), 1897–1900 (2008)
17. Rasband, N.: Chaotic Dynamics of Nonlinear Systems. Wiley-Interscience, Chichester (1997)
18. Nagarajan, R.: Quantifying physiological data with Lempel-Ziv complexity - certain issues. IEEE Trans Biomed. Engg. 49(11), 1371–1372 (2002)
19. Nagarajan, R., Szczepanski, J., Wajnryb, E.: Interpreting non-random signatures in biomedical signals with Lempel–Ziv complexity. Physica. D. 237(3), 359–364 (2008)
20. Theiler, J., Eubank, S., Longtin, A., Galdrikian, B., Farmer., J.D.: Testing for nonlinearity in time series: the method of surrogate data. Physica D 58, 77–94 (1992)
21. Schreiber, T., Schmitz, A.: Surrogate time series. Physica D 142, 346–382 (2000)
22. Bean, J.M., et al.: High functional overlap between MluI cell-cycle box binding factor and Swi4/6 cell-cycle box binding factor in the G1/S transcriptional program in Saccharomyces cerevisiae. Genetics 171(1), 49–61 (2005)
23. Miyajima., I., et al.: GPA1, a haploid-specific essential gene, encodes a yeast homolog of mammalian G protein which be involved in mating factor signal transduction. Cell 50(7), 1011–1019 (1987)
24. Jorgensen, P., et al.: High-resolution genetic mapping with ordered arrays of Saccharomyces cerevisiae deletion mutants. Genetics 162, 1091–1099 (2002)
25. Bogomolnaya, L.M., Pathak, R., Guo, J., Polymenis, M.: Roles of the RAM signaling network in cell cycle progression in Saccharomyces cerevisiae. Curr. Genet. 49(6), 384–392 (2006)
26. de Lichtenberg, U., et al.: New weakly expressed cell cycle-regulated genes in yeast. Yeast 22(15), 1191–11201 (2005)
27. Kemp, H.A., Sprague Jr., G.F.: Far3 and five interacting proteins prevent premature recovery from pheromone arrest in the budding yeast Saccharomyces cerevisiae. Mol. Cell. Biol. 23(5), 1750–1763 (2003)

A Study on the Importance of Differential Prioritization in Feature Selection Using Toy Datasets

Chia Huey Ooi, Shyh Wei Teng, and Madhu Chetty

Faculty of Information Technology
Monash University, Australia
chiahuey.ooi@gms.edu.sg,
{shyh.wei.teng,madhu.chetty}@infotech.monash.edu.au

Abstract. Previous empirical works have shown the effectiveness of differential prioritization in feature selection prior to molecular classification. We now propose to determine the theoretical basis for the concept of differential prioritization through mathematical analyses of the characteristics of predictor sets found using different values of the DDP (degree of differential prioritization) from realistic toy datasets. Mathematical analyses based on analytical measures such as distance between classes are implemented on these predictor sets. We demonstrate that the optimal value of the DDP is capable of forming a predictor set which consists of classes of features which are well separated and are highly correlated to the target classes – a characteristic of a truly optimal predictor set. From these analyses, the necessity of adjusting the DDP based on the dataset of interest is confirmed in a mathematical manner, indicating that the DDP-based feature selection technique is superior to both simplistic rank-based selection and state-of-the-art equal-priorities scoring methods. Applying similar analyses to real-life multiclass microarray datasets, we obtain further proof of the theoretical significance of the DDP for practical applications.

1 Introduction

The aim of feature selection is to form, from all available features in a dataset, a relatively small subset of features capable of producing the optimal classification accuracy. This subset is called the predictor set. A feature selection technique is made up of two components: the predictor set scoring method (which evaluates the goodness of a candidate predictor set); and the search method (which searches the gene subset space for the predictor set based on the scoring method). This study focuses on filter-based technique which its classifiers are not invoked in the predictor set scoring method.

An important principle behind most filter-based feature selection studies can be summarized by the following statement: A good predictor set should contain features highly correlated to the target class concept, and yet uncorrelated with each other [1]. The predictor set attribute referred to in the first part of this statement, 'relevance', is the backbone of simple rank-based feature selection techniques. The aspect alluded to in the second part, 'redundancy', refers to pairwise relationships between all pairs of features in the predictor set.

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 311–322, 2008.

Previous studies [1, 2] have based their feature selection techniques on the concept of relevance and redundancy having equal importance in the formation of a good predictor set. We call the predictor set scoring methods used in such correlation-based feature selection techniques *equal-priorities scoring methods*. On the other hand, it is demonstrated in another study [3] using a two-class problem that seemingly redundant features may improve the discriminant power of the predictor set instead, although it remains to be seen how this scales up to multiclass domains with thousands of features. A previous study was implemented on the effect of varying the importance of redundancy in predictor set evaluation [4]. However, due to its use of a relevance score that is inapplicable to multiclass problems, the study was limited to only two-class classification.

Currently, when it comes to the use of filter-based feature selection for multiclass tumor classification, three popular recommendations are: 1) no selection [5, 6]; 2) select based on relevance alone [5, 7]; and finally, 3) select based on relevance and redundancy [2, 8]. Thus, so far, relevance and redundancy are the two existing criteria which have ever been used in predictor set scoring methods for multiclass tumor classification.

To these two criteria we introduce a third criterion: the relative importance placed between relevance and redundancy [9]. We call this criterion the degree of differential prioritization (DDP). DDP compels the search method to prioritize the optimization of one of the two criteria (of relevance or redundancy) at the cost of the optimization of the other. Unlike other existing correlation-based techniques, the DDP-based feature selection technique does not take for granted that the optimizations of both elements of relevance and redundancy are to have equal priorities in the search for the predictor set [10].

Although a large body of work has provided empirical support regarding the efficacy of the DDP concept in feature selection [9-11] , we have yet to establish the theoretical strengths and merits of the DDP-based feature selection technique. This is precisely the aim of this paper, which is to be realized through vigorous mathematical analyses of predictor sets found using the DDP-based feature selection technique and simple but illustrative examples using toy datasets.

To generate toy datasets for this purpose, we employ a model which is well-known and recognized not only in the domains of molecular classification and microarray analysis but also conventional data minings. Later in this paper, we also show how close conditions in real-life multiclass microarray datasets resemble those of our toy datasets. Additional advantages of toy datasets include the unlimited number of datasets we can generate [vs. the limited number of available real-life microarray datasets]; the control we are able to exercise over dataset characteristics such as the number of classes and features [11]; and prior knowledge of the members of the ideal predictor set, which provides the ultimate means for measuring the efficacy of the feature selection technique without involving the inductions of actual classifiers.

The organization of the paper is as follows: Beginning with a description of the DDP-based feature selection technique, we proceed to present a model for producing the toy datasets. Then, we analyze the class separation property of the predictor sets obtained from each of the toy datasets. We then apply the same analysis to eight real-life multiclass microarray datasets. Finally, we present the conclusions of the study.

2 Differential Prioritization

For gene expression datasets, the terms *gene* and *feature* may be used interchangeably. From the total of N genes, the objective is to form the subset of genes, called the predictor set S, which gives the optimal classification accuracy.

The score of goodness for predictor set S is given as follows.

$$W_{A,S} = (V_S)^{\alpha} \cdot (U_S)^{1-\alpha} \tag{1}$$

V_S represents the relevance of S. V_S measures the average of the correlation of the members of S to the target class concept.

$$V_S = \frac{1}{|S|} \sum_{i \in S} F(i) \tag{2}$$

The target class concept is represented by the target class vector **y**, which is defined as $[y_1, y_2, \ldots, y_{|T|}]$, $y_j \in [1, K]$ in a K-class dataset. y_j is the class label of sample j. The training set, T, consists of all training samples of K classes. Based on **y**, the relevance of gene i is computed as follows.

$$F(i) = \frac{\sum_{j \in T} \sum_{k=1}^{K} I(y_j = k)(\bar{x}_{i,k} - \bar{x}_{i,\bullet})^2}{\sum_{j \in T} \sum_{k=1}^{K} I(y_j = k)(x_{i,j} - \bar{x}_{i,k})^2} \tag{3}$$

where $I(.)$ is an indicator function returning 1 if the condition inside the parentheses is true, otherwise it returns 0. $\bar{x}_{i,\bullet}$ is the average of the expression of gene i across all training samples in T. $\bar{x}_{i,k}$ is the average of the expression of gene i across training samples belonging to class k. $x_{i,j}$ is the expression of gene i in sample j. $F(i)$ is the BSS/WSS (between-groups sum of squares/within-groups sum of squares) ratio for gene i [12]. It indicates the ability of the gene in discriminating among samples belonging to K different classes.

U_S represents the antiredundancy of S. Antiredundancy is a measure opposite to redundancy in quality [9].

$$U_S = \frac{1}{|S|^2} \sum_{i,j \in S, i \neq j} 1 - |R(i,j)| \tag{4}$$

The absolute value of the Pearson product moment correlation coefficient between genes i and j, $|R(i,j)|$, is used to measure the correlation between genes i and j.

The power factor $\alpha \in (0, 1]$ in Eq. 1 denotes the DDP between maximizing relevance and maximizing antiredundancy. Decreasing the value of α forces the search

method to put more priority on maximizing antiredundancy at the cost of maximizing relevance. Raising the value of α increases the emphasis on maximizing relevance (and at the same time decreases the emphasis on maximizing antiredundancy) during the search for the predictor set [10].

A predictor set found using a larger value of α has more features with strong relevance to the target class concept, but also more redundancy among these features. Conversely, a predictor set obtained using a smaller value of α contains less redundancy among its member features, but at the same time also has fewer features with strong relevance to the target class concept. At $\alpha = 0.5$, we get an equal-priorities scoring method. At $\alpha = 1$, the feature selection technique becomes rank-based.

We posit that different datasets will require different values of the DDP between maximizing relevance and maximizing antiredundancy in order to come up with the most efficacious predictor set. Therefore the optimal range of α (leading to the predictor set giving the best accuracy) is dataset-specific.

The linear incremental search is conducted as follows: The first member of S is chosen by selecting the gene with the highest $F(i)$ score. To find the second and the subsequent members of the predictor set, the remaining genes are screened one by one for the gene that would give the maximum $W_{A,S}$. Since the combination of our predictor set scoring method and this search method does not specify an output as to the final size of the predictor set to be used, the maximum size of the predictor set, P, will have to be predetermined by the user.

3 Toy Datasets Based on One-vs.-All (OVA) Model

In using toy datasets, we aim to provide simple but clear and demonstrative examples which highlight the importance of choosing the correct value of the DDP in forming the best predictor set. Furthermore, another advantage of toy datasets is the fact that we know exactly just how large a predictor set should be found for each case, facilitating the task of choosing the value of P.

It is widely accepted that over-expression or under-expression (suppression) of genes causes the difference in phenotype among samples of different classes. The categorization of gene expression is given as follows.

1. A gene is over-expressed: if its expression value is above baseline.
2. A gene is under-expressed: if its expression value is below baseline.
3. Baseline interval: the normal range of expression value.

Usually the mean of the expression across genes is taken as the middle of the baseline interval. In analyses of microarray data, the conventional data normalization procedures often set the mean across genes for each sample to zero. Hence, over-expression is represented by positive values and under-expression by negative values. With this categorization we next employ a well-known paradigm leading to the one-vs.-all (OVA) model, which is then used to generate toy datasets.

The crux of the OVA concept has gained wide, albeit tacit, acceptance among microarray and tumor gene expression researchers. The fact that particular genes are only over-expressed in tissues of certain type of cancer, and not any other types of cancer or normal tissues [13], is part of the entrenched domain knowledge. Hence the

term 'marker' – for genes that mark the particular cancer they are associated with. In the OVA model, certain groups of genes, also called the 'marker genes', are only over-expressed (or under-expressed) in samples belonging to a particular class and never in all samples of other classes. This model emphasizes that a group of marker genes is specific to one class. Therefore for a K-class dataset, there are K different groups of marker genes.

Let us denote as G the number of genes in each group of marker genes, $X_{\max}$ and $X_{\min}$ the maximum and minimum limits respectively to the absolute value of the class means for the whole dataset. Thus, for the g-th gene in a group of marker genes, the maximum limit to the absolute value of the class means is defined as:

$$x_{\max,g} = X_{\max} - (\Delta X)(g-1) \tag{5}$$

where $g = 1, 2, \ldots, G$, and

$$\Delta X = \frac{X_{\max} - X_{\min}}{G-1} \tag{6}$$

Among the K classes, the class means are made to vary such that there is an imbalance in terms of class means. The reasons are firstly to mimic a condition prevalent in multiclass microarray datasets (imbalance among classes in terms of class means even after normalization), especially in datasets with very large number of classes; and secondly, to present a challenge to the feature selection technique in choosing relevant but non-redundant genes. We will provide further elucidation on the second reason later in this section. For the g-th gene in a group of marker genes, the difference between the class means of subsequent classes is defined in the following manner.

$$\Delta x_g = \frac{2x_{\max,g}}{K-1} \tag{7}$$

Next, initialize a matrix $M:=(\mu_{i,k})_{N\times K}$ of zeros where N is the total number of genes in the dataset, and, in this case, is the product of G and K. This is the matrix of class means, whose element, $\mu_{i,k}$, represents the mean of gene i across samples belonging to class k ($k = 1, 2, \ldots, K$).

$$\mu_{(g-1)K+k,k} = \left(-1^g\right)\left[x_{\max,g} - (\Delta x_g)(k-1)\right] \tag{8}$$

The $[(g-1)K+k]$-th gene is the g-th member of the k-th group of marker genes and therefore has non-zero class mean for class k and zero class mean for all other classes – the archetypal OVA trait. The term $\left(-1^g\right)$ serves to change the sign of the class mean at different values of g so as to produce both over- and under-expressed marker genes. The strongest marker genes are composed of the first genes ($g = 1$) of each group of marker genes while the weakest marker genes consist of the last genes ($g = G$) of each group of marker genes.

Standard deviation among samples of the same class, or class standard deviation, is set to 1 for all instances, $\sigma_{i,k} = 1$ for all k and i. To produce a K-class toy dataset, a

Table 1. A 4-class example from the OVA model (μi,k represents the mean of gene i across samples belonging to class k)

g	k	$\mu_{i,k}$	$\mu_{i,1}$	$\mu_{i,2}$	$\mu_{i,3}$	$\mu_{i,4}$
1	1	$\mu_{1,k}$	$-X_{\max}$	0	0	0
1	2	$\mu_{2,k}$	0	$-0.5X_{\max}$	0	0
1	3	$\mu_{3,k}$	0	0	$0.5X_{\max}$	0
1	4	$\mu_{4,k}$	0	0	0	$X_{\max}$
2	1	$\mu_{5,k}$	$X_{\max}-\Delta X$	0	0	0
2	2	$\mu_{6,k}$	0	$0.5(X_{\max}-\Delta X)$	0	0
2	3	$\mu_{7,k}$	0	0	$0.5(\Delta X-X_{\max})$	0
2	4	$\mu_{8,k}$	0	0	0	$\Delta X-X_{\max}$
⋮	⋮	⋮	⋮	⋮	⋮	⋮
G	1	$\mu_{(G-1)K+1,k}$	$(-1^G)X_{\min}$	0	0	0
G	2	$\mu_{(G-1)K+2,k}$	0	$0.5(-1^G)\ X_{\min}$	0	0
G	3	$\mu_{(G-1)K+3,k}$	0	0	$-0.5(-1^G)\ X_{\min}$	0
G	4	$\mu_{(G-1)K+4,k}$	0	0	0	$-(-1^G)X_{\min}$

total of m samples are generated for class k (k = 1, 2, …, K) using Gaussian distribution of mean $\mu_{i,k}$ and standard deviation $\sigma_{i,k}$ for gene i.

In Table 1, an entry on the i-th row and k-th column represents the class mean of class k for gene i, where $i = \lfloor (g-1)K + k \rfloor$, and therefore gene i is the g-th member of the k-th group of marker genes. We can see that using relevance alone as a criterion, and with uniform class size, marker genes associated with class 1 and 4 will always be favored more than marker genes specific to any other classes, regardless of the value of g. Including antiredundancy as the second criterion will obviate this imbalanced predilection – therein lies the reason for us to use unequal values for class means among different classes. But how much weight is to be assigned to relevance, and how much to antiredundancy?

The apparent answer would be equal weights, which is the foundation of existing equal-priorities scoring methods. But as mentioned previously in Section 1, it has been shown that antiredundancy is not as important as relevance for the two-class case – this is obvious in the case of our OVA toy dataset; any subset of sufficiently relevant genes will be capable of differentiating between the two classes. Hence as the number of classes increases (an important theme in multiclass classification studies), will not the importance of antiredundancy (w.r.t. relevance) increase as well? This question is to be answered from the analyses in this study.

4 Experiment Settings

Ten values of α are tested from 0.1 to 1 with equal intervals of 0.1. G is set to 3, 5, 10, 20, and 30. $X_{\max}$ and $X_{\min}$ is set to 100 and 1 respectively, while the number of samples per class, or class size, m, is set to 100 uniformly for all classes. We test for $K = 2$ to $K = 30$. Since no inductions of classifiers are to be implemented in this study, whole datasets are used as training sets during feature selection.

The minimum predictor set size necessary to differentiate among the K classes is $K-1$. The optimal predictor set is actually any subset of $K-1$ genes from the first K of the marker genes (that is, at $g=1$) generated using the class means defined in Eq. 8. Thus, P is set to $K-1$.

5 Analyses of Predictor Sets

In this section we critically analyze the properties of the predictor sets produced from different values of α. The property of separation of classes is studied. Finally, we apply the analyses to real-life datasets for comparison to results from toy datasets.

5.1 Separation of Classes

The goodness of a predictor set can be measured by how well separated the classes of features in the predictor set are. If there are two predictor sets with the same relevance score, the set with better separated classes of features is deemed to be better as there is less redundancy among its features. A natural way to measure separation of classes is the distance between pairs of class means. In this study, we use the Euclidean distance metric. For the q-th pair of classes, $C_q = \{c_{1,q}, c_{2,q}\}$, the separation between classes given by the predictor set found through a DDP value of α, S_α, measured using the Euclidean metric is given below.

$$d_{E,\alpha}(c_{1,q}, c_{2,q}) = \sqrt{\sum_{i \in S_\alpha} \left| \bar{x}_{i,c_{1,q}} - \bar{x}_{i,c_{2,q}} \right|^2} \tag{9}$$

$\bar{x}_{i,k}$ is the average of the expression of gene i across samples belonging to class k. Averaging across all ${}^{K}C_2$ pairs of classes, we obtain the mean Euclidean distance between all pairs of classes as measured by S_α.

$$\bar{d}_{E,\alpha} = \frac{1}{{}^{K}C_2} \sum_{q=1}^{{}^{K}C_2} d_{E,\alpha}(c_{1,q}, c_{2,q}) \tag{10}$$

The value of the DDP leading to the best separation of classes in terms of the Euclidean metric is the one which gives the largest $\bar{d}_{E,\alpha}$.

$$\alpha_E^* = \arg\max_{\alpha} \left(\bar{d}_{E,\alpha} \right) \tag{11}$$

If there is more than one value of α satisfying Eq. 11, the mean among these values is taken as α_E^*. Since these values are generally adjacent to each other, taking the mean will still provide a good picture of how the DDP affects separation of classes.

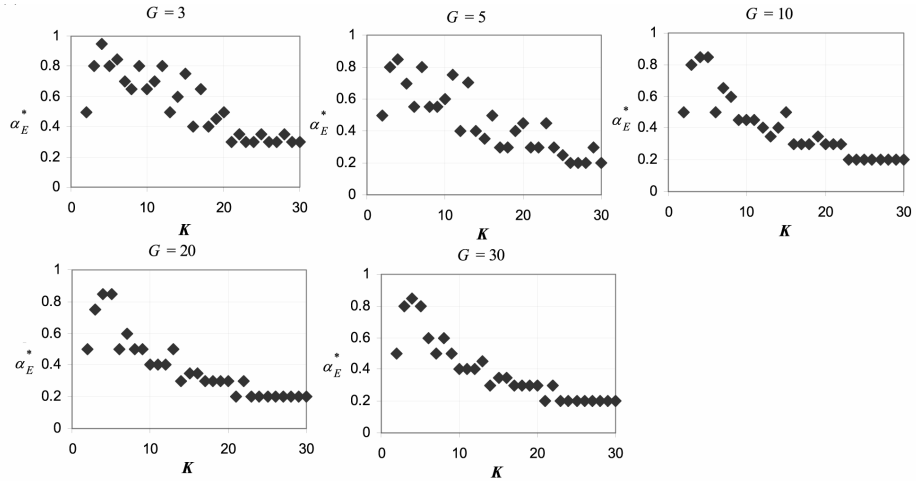


Fig. 1. The DDP producing the optimal separation of classes as measured by the Euclidean distance, α_E^*, as a function of K for toy datasets generated from (a) the OVA model and (b) the PW model

Fig. 1 shows that the number of classes, K, influences the value of α_E^*, regardless of the value set to G. Larger G tends to produce a more distinct α_E^*-K plot. As K increases beyond 20, α_E^* settles to a smaller value (around 0.2). Regardless of the parameter values of G, the number of classes in the dataset undoubtedly affects the value of the DDP which brings about the best separation of classes in terms of the Euclidean distance.

6 Real-Life Datasets

We now apply the previous analyses to real-life datasets. Descriptions of eight real-life microarray datasets are shown in Table 2. The Brown (BRN) dataset [14] includes 15 broad cancer types. Following a previous study [15], the skin tissue samples due to small class size (3 samples) are excluded from analysis. The GCM dataset [13] contains 14 tumor classes. For the NCI60 dataset [16], only 8 tumor classes are analyzed; the 2 samples of the prostate class are excluded due to the small class size.

The PDL dataset [17] consists of 6 classes, each class representing a diagnostic group of childhood leukemia. The SRBC dataset [18] consists of 4 subtypes of small, round, blue cell tumors (SRBCTs). In the 5-class lung dataset [19], 4 classes are subtypes of lung cancer; the fifth class consists of normal samples. The MLL dataset [20] contains 3 subtypes of leukemia: ALL, MLL, and AML. The AML/ALL dataset [21] also contains 3 subtypes of leukemia: AML, B-cell ALL, and T-cell ALL.

Table 2. Descriptions of real-life datasets. *N* is the number of genes after preprocessing. *K* is the number of classes in the dataset.

Dataset	Type	*N*	*K*	Training:Test set size
BRN	cDNA	7452	14	174:83
GCM	Affymetrix	10820	14	144:54
NCI60	cDNA	7386	8	40:20
PDL	Affymetrix	12011	6	166:82
Lung	Affymetrix	1741	5	135:68
SRBC	cDNA	2308	4	55:28
MLL	Affymetrix	8681	3	48:24
AML/ALL	Affymetrix	3571	3	48:24

Except for the BRN and SRBC datasets (which are only available as preprocessed in their originating studies), datasets are preprocessed and normalized based on the recommended procedures [12] for Affymetrix and cDNA microarray data. Except for the GCM dataset, for which the ratio of training to test set sizes used in the originating study [13] is maintained to enable comparison with previous studies, for all datasets we employ the standard 2:1 split ratio.

But before applying the analyses to real-life datasets, we investigate how close conditions in real-life datasets match those of toy datasets.

6.1 Investigating the Imbalance of Class Means in Real-Life Datasets

We have mentioned in the section on the generation of toy datasets that imbalance in terms of class means among classes is prevalent in highly multiclass microarray datasets. Investigation is conducted on whole datasets (no splitting) in order to determine the extent of the aforementioned imbalance. For class k ($k = 1,2,\ldots,K$), we choose the class mean with the greatest absolute value (equivalent to the absolute value of $\mu_{(g-1)K+k,k}$ from Eq. 8 or $\mu_{(g-1)\binom{K}{2}+q,c_{b,q}}$ from Eq. 9 at $g = 1$) among all *N* class means.

$$\bar{x}_{0,k} = \max_{i=1,2,\ldots,N} \left(\left| \bar{x}_{i,k} \right| \right) \tag{12}$$

Next, to illustrate the imbalance among classes in terms of class means, we compute the range of class means.

$$R\left(\bar{x}_{0,k}\right) = \max_{k}\left(\bar{x}_{0,k}\right) - \min_{k}\left(\bar{x}_{0,k}\right) \tag{13}$$

The result is shown in Table 3. We observe that the range $R\left(\bar{x}_{0,k}\right)$ for datasets with large *K* (such as BRN, GCM, and NCI60) is greater than the range $R\left(\bar{x}_{0,k}\right)$ for datasets with smaller *K*. Looking at the maximum and minimum values of $\bar{x}_{0,k}$ across *k* in Table 3, we can say with certainty that there is an imbalance among classes in terms of class means, especially in datasets containing more than 6 classes.

Table 3. Range of class means in real-life datasets

Dataset	$\max_k(\bar{x}_{0,k})$	$\min_k(\bar{x}_{0,k})$	$R(\bar{x}_{0,k})$
BRN	11.57	3.93	7.65
GCM	73.00	7.77	65.23
NCI60	7.64	4.20	3.44
PDL	2.77	2.67	0.10
Lung	9.62	8.39	1.23
SRBC	2.27	1.59	0.68
MLL	4.35	4.08	0.27
AML/ALL	7.14	6.76	0.38

We expect this imbalance to either increase or at least stay constant as K increases beyond 14 (which is the largest number of classes to be found among real-life datasets). Therefore the implementation of unequal maximum limits to the absolute value of the class means for different classes in Eqs. 8 and 9 is justified, particularly in analyses involving K as high as 30 for toy datasets, as is the case in this study.

6.2 Applying the Analyses to Real-Life Datasets

For real-life datasets, the analyses are implemented separately upon the training set of each split, there being a total of 10 splits of training and test sets. The mean across all splits is taken for the $\overline{d}_{E,\alpha}$ measured in the analyses, and then used to find the corresponding value of the DDP which optimizes $\overline{d}_{E,\alpha}$.

We will assume that the optimal P for each real-life dataset is directly proportional to K (as is the case for toy datasets). However, allowing for remnant noise (left even after data preprocessing), we assign larger values to P for real-life datasets ($30K$) than for toy datasets with similar K.

Fig. 2 shows that for the majority of real-life datasets, the trend regarding $\overline{d}_{E,\alpha}$ is similar to the trend for toy datasets. However, in the α_E^*–K plot, one dataset

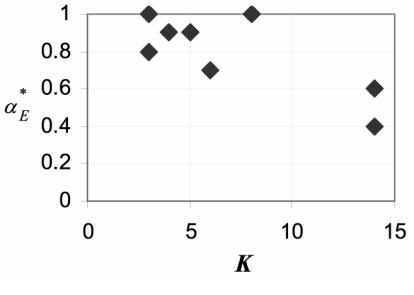


Fig. 2. α_E^* values of the DDP which optimize various predictor set characteristics as a function of K for real-life microarray datasets

(NCI60) produces a point ($\alpha_E^* = 1$ at $K = 8$) which diverges from the $\alpha_E^* - K$ plots observed in toy datasets. Despite this discrepancy (due to small class sizes and the heterogeneity of some of the classes), the overall picture provided by Fig. 2 indicates that the effect of *K* on the values of the DDP which optimize $\overline{d}_{E,\alpha}$ in real-life datasets is the same as the effect in toy datasets.

7 Conclusions

In this paper, we have proposed a systematic method to model toy datasets based on the OVA concept. We have used these toy datasets for analyzing the DDP concept in feature selection. The findings in this study have shown that DDP is necessary because it subsumes existing techniques such as equal-priorities scoring methods and rank-based selection. The optimal value of α is not always 0.5 (equal-priorities scoring methods) or 1 (rank-based selection). Instead, it is based on the number of classes in the datasets. By using this optimal value in the DDP-based feature selection technique, we can then find the predictor set with the best adjustment of maximum relevance and minimum redundancy pertinent to the number of classes in the dataset.

Finally, since all findings have been achieved based on analytical evaluations, not empirical evaluations involving classifiers, this study establishes the theoretical basis for the usefulness of the DDP.

References

1. Hall, M.A., Smith, L.A.: Practical feature subset selection for machine learning. In: Paper presented at the Proc. 21st Australasian Computer Science Conf. (1998)
2. Ding, C., Long, F., Peng, H.: Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy. IEEE Transactions on Pattern Analysis and Machine Intelligence 27(8), 1226–1238 (2005)
3. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. J. Machine Learning Research 3, 1157–1182 (2003)
4. Knijnenburg, T.A., Reinders, M.J.T., Wessels, L.F.A.: The selection of relevant and non-redundant features to improve classification performance of microarray gene expression data. In: Proc. 11th Annual Conf. of the Advanced School for Computing and Imaging, Heijen, NL (2005)
5. Li, T., Zhang, C., Ogihara, M.: A comparative study of feature selection and multiclass classification methods for tissue classification based on gene expression. Bioinformatics 20, 2429–2437 (2004)
6. Ramaswamy, S., Tamayo, P., Rifkin, R., Mukherjee, S., Yeang, C.H., Angelo, M., et al.: Multi-class cancer diagnosis using tumor gene expression signatures. Proc. Natl. Acad. Sci. USA 98, 15149–15154 (2001)
7. Chai, H., Domeniconi, C.: An evaluation of gene selection methods for multi-class microarray data classification. In: Paper presented at the Proc. 2nd European Workshop on Data Mining and Text Mining in Bioinformatics (2004)
8. Yu, L., Liu, H.: Redundancy based feature selection for microarray data. In: Paper presented at the Proc. of ACM SIGKDD 2004 (2004)

9. Ooi, C.H., Chetty, M., Gondal, I.: The role of feature redundancy in tumor classification. In: Zhang, D., Jain, A.K. (eds.) ICBA 2004. LNCS, vol. 3072. Springer, Heidelberg (2004)
10. Ooi, C.H., Chetty, M., Teng, S.W.: Relevance, redundancy and differential prioritization in feature selection for multiclass gene expression data. In: Oliveira, J.L., Maojo, V., Martín-Sánchez, F., Pereira, A.S. (eds.) ISBMDA 2005. LNCS (LNBI), vol. 3745. Springer, Heidelberg (2005)
11. Ooi, C.H., Chetty, M., Teng, S.W.: Modeling microarray datasets for efficient feature selection. In: Paper presented at the Proc. 4th Australasian Conf. on Knowledge Discovery and Data Mining (AusDM 2005) (2005a)
12. Dudoit, S., Fridlyand, J., Speed, T.: Comparison of discrimination methods for the classification of tumors using gene expression dat. J. Am. Stat. Assoc. 97, 77–87 (2002)
13. Ramaswamy, S., Tamayo, P., Rifkin, R., Mukherjee, S., Yeang, C.H., Angelo, M., et al.: Multi-class cancer diagnosis using tumor gene expression signatures. Proc. Natl. Acad. Sci. USA 98, 15149–15154 (2001)
14. Munagala, K., Tibshirani, R., Brown, P.: Cancer characterization and feature set extraction by discriminative margin clustering. BMC Bioinformatics 5, 21 (2004)
15. Park, M., Hastie, T.: Hierarchical classification using shrunken centroids. Department of Statistics, Stanford University. Technical Report (2005), http://www-stat.stanford.edu/~hastie/Papers/hpam.pdf
16. Ross, D.T., Scherf, U., Eisen, M.B., Perou, C.M., Rees, C., Spellman, P., et al.: Systematic variation in gene expression patterns in human cancer cell lines. Nat. Genet. 24, 227–235 (2000)
17. Yeoh, E.-J., Ross, M.E., Shurtleff, S.A., Williams, W.K., Patel, D., Mahfouz, R., et al.: Classification, subtype discovery, and prediction of outcome in pediatric lymphoblastic leukemia by gene expression profiling. Cancer Cell 1(2), 133–143 (2002)
18. Khan, J., Wei, J.S., Ringner, M., Saal, L.H., Ladanyi, M., Westermann, F., et al.: Classification and diagnostic prediction of cancers using expression profiling and artificial neural networks. Nat. Med. 7, 673–679 (2001)
19. Bhattacharjee, A., Richards, W.G., Staunton, J.E., Li, C., Monti, S., Vasa, P., et al.: Classification of human lung carcinomas by mRNA expression profiling reveals distinct adenocarcinoma subclasses. Proc. Natl. Acad. Sci. USA 98, 13790–13795 (2001)
20. Armstrong, S.A., Staunton, J.E., Silverman, L.B., Pieters, R., den Boer, M.L., Minden, M.D., et al.: MLL translocations specify a distinct gene expression profile that distinguishes a unique leukemia. Nat. Genet. 30, 41–47 (2002)
21. Golub, T.R., Slonim, D.K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J.P., et al.: Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. Science 286, 531–537 (1999)

Weighted Top Score Pair Method for Gene Selection and Classification

Huaien Luo[1], Yuliansa Sudibyo[2,⋆], Lance D. Miller[1], and R. Krishna Murthy Karuturi[1,⋆⋆]

[1] Genome Institute of Singapore, Singapore
[2] Nanyang Technological University, Singapore
{luoh2,gisv45,millerl,karuturikm}@gis.a-star.edu.sg

Abstract. Gene selection and expression profiles classification are important for diagnosing the disease using microarray technology and revealing the underlying biological processes. This paper proposes a weighted top scoring pair (WTSP) method which is a generalization of the current top scoring pair (TSP) method. By considering the proportions of samples from different classes, the WTSP method aims to minimize the error or misclassification rate. Results from several experimental microarray data have shown the improved performance of classification using the WTSP method.

Keywords: Microarray, Gene selection, Classification, Weighted Top Score Pairs, Cross-validation.

1 Introduction

By measuring the expression levels of thousands of genes, microarray techniques have been used to diagnose and explore the biologically relevant genes related to a disease. The obtained microarray data normally contains several thousands of genes and tens to hundreds of samples. The analysis of this data is challenged by the "small N, large P" problem, that is, the number of genes (P) is greatly larger than the number of samples (N). In order to deal with this high dimensional data and make the analysis feasible, dimension reduction (or gene selection) methods are used to choose the most informative genes by comparing the expression levels between the cancer tissue and normal ones, or between different tumor types. The purpose of the gene selection is to discard those genes which are least interesting to the classification and select the relevant genes which could provide the best ability to distinguish the samples from different classes and hence reveal the biomarkers or molecular signature for the disease. This purpose can be achieved by ranking the genes according to some relevance measurement and select those genes with the highest relevance scores. The commonly used genes selection methods can be categorized into three categories: i) choosing

⋆ This author is the co-first author.
⋆⋆ Corresponding author.

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 323–333, 2008.

single differentially expressed genes; ii) choosing gene pairs which co-regulate; and iii) choosing a set of genes or gene network. To discover the differentially expressed genes, t-statistic could be calculated for each gene and the genes with significantly different expression levels are chosen [1]. This single gene selection method considers the genes independently and may miss the functional relationships among genes due to the interaction/co-regulation of the genes. Some methods are proposed to investigate the information provided by the gene pairs. In [2], two-sample t-statistics are calculated for each gene pairs projected to the diagonal linear discriminant (DLD) axis in order to find the pairs with the highest score that together could discriminate the samples from different classes. In [3], a correlation-based method is developed to discover the gene pairs whose functional relationship changes across different conditions. This method is based on the assumption that gene pairs with largest differential correlation are more likely to be involved in the mechanisms of the disease. In [4], a feature construction method is proposed to find the synergic gene pairs which could enhance the accuracy of the classification. In this method, the mutual information contained in the interaction of the gene pairs is explored for the gene selection and these gene pairs are assumed to have biological significance for the underlying cellular processes. In addition to investigate the information contained in pairs of genes, the microarray data analysis can also be carried out with a list of genes (or gene networks). The information buried in this gene network could reveal the biological function or pathways of these genes related to the disease. In [5], the gene expression data is analyzed by integrating a priori the knowledge of the gene network to achieve a better classification. The hypothesis underlying this approach is that the genes close to the network are more likely to be co-expressed. In [6], a friendly neighbors (FNs) method for time-course microarray data analysis is proposed to find the genes whose induction-repression pattern are shared with other genes more often and these genes are considered to be the most informative for a certain cellular function. Based on this method, a differential friendly neighbors (DiffFNs) method is proposed to choose the genes in which the gain or loss of the relationships with other genes are most significant [7]. These genes could provide the biomarkers to distinguish the tumor from the healthy ones and signify the underlying pathways.

Besides the above methods, the common dimension reduction methods are also used to represent the information of the large number of genes with a set of gene components which could capture as much information of the original gene expression data as possible. These methods include: Q-mode Principle Component Analysis (PCA) which retain most of the variation [8]; Partial Least Squares (PLS) which constructs the components that maximize the covariance between classes and genes [9]; Sliced Inverse Regression (SIR) which regresses the gene expression data on the classes [9].

Having selected the most informative genes, the samples from different classes could be successfully identified. Many algorithms have been proposed to achieve this goal, such as Support Vector Machine (SVM) [10], nearest and k-nearest neighbors (kNN) [11][12], linear discriminant analysis (LDA) [11], Decision Trees

(DT) [13], naive Bayes (NB) [11], Prediction Analysis of Microarrays (PAM) [14] and so on.

Among these gene selection and classification methods, one of the simple and effective methods is Top Scoring Pair (TSP) based methods [15] [16] [17]. This method integrates the gene selection and classification based on a simple rule. It aims to find pairs of genes such that the expression level of gene A is greater than that of gene B in class 1, but smaller in class 2; and this rule is also used for the classification. Being a rank-based method, the TSP is invariant to the preprocessing steps such as normalization since it does not change the rank of a specified gene. Compared to the traditional methods which use more genes and a complex decision procedure, the TSP method is shown to have the ability to achieve comparably high accuracy of classification by using very few genes [18].

In this paper, a weighted TSP (WTSP) method is proposed as a generalization of the classical TSP method. Different from the TSP, the proposed WTSP method adjusts the scores of gene pairs by incorporating the information of the proportion of the samples belonging to different classes and/or the cost of misclassification. This weighted TSP method aims to minimize the cost of misclassifications and hence could achieve better performance compared to the classical TSP. This paper is organized as follows. In Section 2, the method of weighted TSP will be developed. Some implementation issues will also be given in this part. Section 3 presents the results of the proposed method as well as its comparison with the TSP classifier. This is followed by Section 4 where some discussion about the proposed method will be given.

2 Method

The gene expression data can be represented as a matrix $\mathbf{X}$ with dimension $P \times N$, where P is the number of genes and N is the number of samples (or gene expression profiles). Each column in $\mathbf{X}$ is an expression profile of P genes from a sample either in class 1 ($Y = 1$) or in class 2 ($Y = 2$). Normally, the number of genes is greatly larger than the number of samples ($P \gg N$) and this causes the problem of curse of dimensionality.

The TSP method aims to find the gene pairs whose relative relationship of expression levels change from one class to the other. That is, the marker gene pairs should be the ones that the expression level of gene A is greater than that of gene B in class 1, but smaller in class 2. Suppose there are N_1 samples from class 1 and N_2 samples from class 2 ($N_1 + N_2 = N$), and for a gene pair (i, j), there are respectively a_{ij} and b_{ij} samples from class 1 and class 2 with the expression level of gene i less than that of gene j (i.e., $X_i < X_j$). The TSP scheme order the gene pairs according to their scores defined as:

$$\begin{aligned} \Delta_{ij} &= |P(X_i < X_j | Y = 1) - P(X_i < X_j | Y = 2)| \\ &= |p_{ij}(C1) - p_{ij}(C2)| \\ &\approx \left| \frac{a_{ij}}{N_1} - \frac{b_{ij}}{N_2} \right| \end{aligned} \tag{1}$$

By choosing the gene pairs which achieve the top scores in the training data, a new gene expression profile $\mathbf{x}'$ could be classified according to the relation of the expression level X'_i of gene i and X'_j of gene j (or the rank of these two genes) according to the following rule:
If $p_{ij}(C1) > p_{ij}(C2)$,

$$Y' = \begin{cases} 1, & \text{if } X'_i < X'_j \\ 2, & \text{o.w.} \end{cases}, \tag{2}$$

else if $p_{ij}(C1) \leq p_{ij}(C2)$,

$$Y' = \begin{cases} 2, & \text{if } X'_i < X'_j \\ 1, & \text{o.w.} \end{cases} \tag{3}$$

2.1 Weighted TSP Method

The proposed weighted TSP method is based on the classical TSP with the incorporation of the probabilities of the samples belonging to each class and the cost of misclassification. It aims to minimize the cost of misclassification, that is, to minimize the following equation:

$$\text{Cost} = P(\text{error}|Y = 1)P_1\lambda_1 + P(\text{error}|Y = 2)P_2\lambda_2, \tag{4}$$

where, $P_1 = P(Y = 1)$ and $P_2 = P(Y = 2)$ are respectively the probability of the samples coming from class 1 and class 2; λ_1 and λ_2 represent the cost it may induce if a sample is misclassified.

If we specify the classification rule as: if $X_i < X_j$, the sample is classified to class 1 ($Y = 1$); else if $X_i > X_j$, it is classified to class 2 ($Y = 2$); and if $X_i = X_j$, the sample is assigned to the class with higher probability. Let a_{ij} be the number of samples correctly assigned to class 1 under this classification rule (i.e., either $X_i < X_j$ or $X_i = X_j$ with $P_1 > P_2$) and b_{ij} be the number of samples incorrectly assigned to class 2, Eq. (4) could be reduced to:

$$\text{Cost} = \frac{N_1 - a_{ij}}{N_1}P_1\lambda_1 + \frac{b_{ij}}{N_2}P_2\lambda_2 \tag{5}$$

$$= P_1\lambda_1 - (\frac{a_{ij}}{N_1}P_1\lambda_1 - \frac{b_{ij}}{N_2}P_2\lambda_2) \tag{6}$$

It can be easily observed from the above equation that the minimization of the cost of misclassification is actually equivalent to the maximization of the quantity $\frac{a_{ij}}{N_1}P_1\lambda_1 - \frac{b_{ij}}{N_2}P_2\lambda_2$. Therefore, for each gene pair, a weighted score is calculated according to:

$$\Delta'_{ij} = \frac{a_{ij}}{N_1}P_1\lambda_1 - \frac{b_{ij}}{N_2}P_2\lambda_2. \tag{7}$$

Compared to the original score, the weighted score Δ'_{ij} is a generalization of the original score Δ_{ij} by considering the proportion of the samples in each class as well as the cost of misclassification. Here, we consider two special cases of this weighted score.

1. If $P_1\lambda_1 = P_2\lambda_2$, Δ'_{ij} is reduced to a scaled version of the score Δ_{ij} calculated in the classical TSP as shown in Eq. (1). It can also be seen that the original score does not consider the proportions of samples from each class and hence the maximization of the original score is equivalent to minimizing the sum of misclassification probabilities over two classes instead of the probability of total misclassification.
2. If $\lambda_1 = \lambda_2$, minimization of the cost of misclassification in Eq. (4) is actually the minimization of the probability of total misclassification (or the error rate).

By ordering the scores of each pair, the gene pair with the largest score is chosen as the marker gene pair to classify the samples. And for a new expression profile $\mathbf{x}'$, the classification rule now is:

$$Y' = \begin{cases} 1, & \text{if } X'_i < X'_j; \text{ or } X'_i = X'_j \text{ and } P1 > P2 \\ 2, & \text{o.w.} \end{cases}. \tag{8}$$

It is to be noted that in the proposed WTSP method, the absolute sign is discarded compared to the original method and the classification rule is also accordingly simplified. This is because that in the weighted score Δ'_{ij}, the order of the genes in the pair is considered. That is, the scores of both the pair (i, j) and (j, i) are calculated and only the one which can achieve higher score is kept for further analysis. While in the original TSP, the order of the genes in the pair is not considered and hence the absolute sign is used and the classification rule depends on the relative value of $p_{ij}(C1)$ and $p_{ij}(C2)$.

In practice, several gene pairs may achieve the same top score. The original TSP method uses two schemes to deal with this situation: i) use all the top score gene pairs and a majority voting scheme to classify the test samples [17]; ii) find the rank of the genes in the pair and choose the pair whose rank difference of the two genes is largest as the marker gene pair for classification [15]. In the WTSP method, a different scheme is used. We treat the gene pairs whose scores are close to the top score as having the same power to classify the samples. This is because that the relative relationship of X_i and X_j may reverse due to noise and this may cause that the measured a_{ij} and b_{ij} are slightly different from the real ones (especially when the X_i and X_j are close to each other). Therefore, it is desirable to treat these gene pairs as potential pairs to be chosen for classifying the test samples. Among these gene pairs, the marker gene pair should have the property that their expression levels are most negatively correlated. And this marker gene pair is the one used in the proposed WTSP (w/ corr.) for classifying the test samples.

2.2 Cross-Validation

In this paper, leave-one-out cross-validation (LOOCV) is used to estimate the error or misclassification rate. For each sample in the available training data with known class, we select the gene pair and build the classifier from the remaining samples. The sample which is left out is treated as the test sample and the

classification is made according to the classifier established from the remaining training samples. The classification accuracy is then calculated as the correct classification divided by the number of samples.

Due to large number of genes, an efficient algorithm to perform the cross-validation is desired. To achieve this, an accelerated cross-validation scheme is utilized based on the idea that the gene pairs which possess very low scores can be ignored since they never have the chance to be chosen as the top scoring pair (or top two scoring pairs in the proposed WTSP (w/ corr.) method) no matter which sample is left out in the process of cross-validation. This can be realized by calculating the lower bound and upper bound of the weighted scores based on all samples for each gene pair. The following steps describe this procedure.

1. For each gene pair (i, j), first calculate the weighted scores Δ'_{ij} according to Eq. (7) by using all the samples, note down respectively the a_{ij} and b_{ij}.
2. Calculate the lower and upper bound of the weighted score of gene pair (i, j) when one sample is left out. This can be done by calculating the following four terms:

$$\Delta^1_{ij} = \frac{a_{ij}}{N_1 - 1} P_1\lambda_1 - \frac{b_{ij}}{N_2} P_2\lambda_2, \text{ if the sample is from class 1 and } X_i > X_j$$

$$\Delta^2_{ij} = \frac{a_{ij} - 1}{N_1 - 1} P_1\lambda_1 - \frac{b_{ij}}{N_2} P_2\lambda_2, \text{ if the sample is from class 1 and } X_i < X_j$$

$$\Delta^3_{ij} = \frac{a_{ij}}{N_1} P_1\lambda_1 - \frac{b_{ij}}{N_2 - 1} P_2\lambda_2, \text{ if the sample is from class 2 and } X_i > X_j$$

$$\Delta^4_{ij} = \frac{a_{ij}}{N_1} P_1\lambda_1 - \frac{b_{ij} - 1}{N_2 - 1} P_2\lambda_2, \text{ if the sample is from class 2 and } X_i < X_j$$

   It can be easily observed that $\Delta^2_{ij} < \Delta^1_{ij}$ and $\Delta^3_{ij} < \Delta^4_{ij}$.
   So the lower bound is then:

$$\Delta^L_{ij} = \min(\Delta^2_{ij}, \Delta^3_{ij}), \tag{9}$$

   and the upper bound is:

$$\Delta^U_{ij} = \max(\Delta^1_{ij}, \Delta^4_{ij}). \tag{10}$$

3. Find the lower bound of the top score pair based on all the samples, and discard those gene pairs whose upper bound is less than the lower bound of the top score pair since for these gene pairs, their weighted score cannot become the top one no matter which sample is left out.

By using this scheme, a list of gene pairs $\mathcal{L}$ is obtained. In each LOOCV loop, only the gene pairs in the list $\mathcal{L}$ are investigated and the weighted scores are updated as well. This procedure greatly reduces the number of gene pairs that we need to investigate and hence largely increases the time and space efficiency of the cross-validation.

3 Evaluation and Results

The proposed weighted TSP method was then tested on the data available from the public database as well as from our own side. These data sets are respectively Leukemia [19], Colon [20], Lung [21], DLBCL [22], GCM [23], CNS [24], Prostate [25], p53 [26]. Table 1 gives a summary of these data sets, such as the number of genes measured (P), total number of samples (N) and the number of samples in each class.

Table 2 shows the comparison of the performance of the proposed weighted TSP and the original TSP. The classification accuracy is estimated using the LOOCV. For the WTSP method, we choose $\lambda_1 = \lambda_2$ to calculate the weighted scores. In this table, "WTSP (w/o corr.)" means that all the gene pairs with the same top weighted scores are used to classify the samples and the classification result is based on the majority voting strategy. "WTSP (w/ corr.)" means weighted TSP with the consideration of the cross-correlation of the expression levels of the gene pair. The gene pairs with the weighted score at least second to the top ones are chosen as the potential gene pairs for classification and only

Table 1. Description of the Data Sets

Data sets	# genes (P)	# total samples (N)	# samples by class (N_1/N_2)
Leukemia	7129	72	47 ALL / 25 AML
Colon	2000	62	40 Tumor / 22 Normal
Lung	12533	181	150 ADCA / 31 MPM
DLBCL	7129	77	58 DLBCL / 19 FL
GCM	16063	280	190 Tumor / 90 Normal
CNS	7129	34	25 Classic / 9 Desmoplastic
Prostate	12625	88	50 Normal / 38 Tumor
p53	44928	257	59 p53+ / 198 p53-

Table 2. Classification Accuracy for 8 Data Sets

Data Sets	WTSP (w/o corr.)	WTSP (w/ corr.)	TSP (w/o rank)	TSP (w/ rank)
Leukemia	95.83%	**97.22%**	93.80%	94.44%
Colon	91.13%	90.32%	91.13%	**91.94%**
Lung	**99.17%**	95.58%	**99.17%**	98.30%
DLBCL	97.40%	94.80%	**98.05%**	97.40%
GCM	77.5%	**84.64%**	75.40%	75.40%
CNS	**83.82%**	79.41%	**83.82%**	79.41%
Prostate	65.34%	**75.00%**	55.68%	54.55%
p53	**79.76%**	79.00%	76.65%	76.65%
Average	86.24%	**87.00%**	84.21%	83.51%
Std	11.76 %	8.63%	14.66%	14.98%
Min	65.34%	75.00%	55.68%	54.55%
Max	99.17%	97.22%	99.17%	98.30%

the one which is most negatively correlated is chosen as the marker gene pair to classify the samples. Similarly, "TSP (w/o rank)" and "TSP (w/ rank)" represent the original TSP method respectively either using all the gene pairs having the same original top scores with majority voting strategy, or choosing the one whose average rank difference is largest [15].

From this table, it is clearly seen that on average, the WTSP-based methods work better than the original TSP-based methods. For 4 out of 8 cases, both WTSP (w/o corr.) and WTSP (w/ corr.) outperform either the TSP with or without rank (respectively Leukemia, GCM, Prostate and p53). For the data sets of Lung and CNS, the WTSP (w/o corr.) performs as well as the best of the TSP-based methods. Only in the DLBCL and Colon case, the WTSP method works slightly worse than the TSP method, but the difference is not significant (respectively, 0.65% and 0.81% difference). An obvious observation is that when the classification accuracies of TSP-based methods are high, the performance of the WTSP-based methods are comparable to the TSP-based methods. However, when the classification accuracies of TSP-based methods are low (such as in

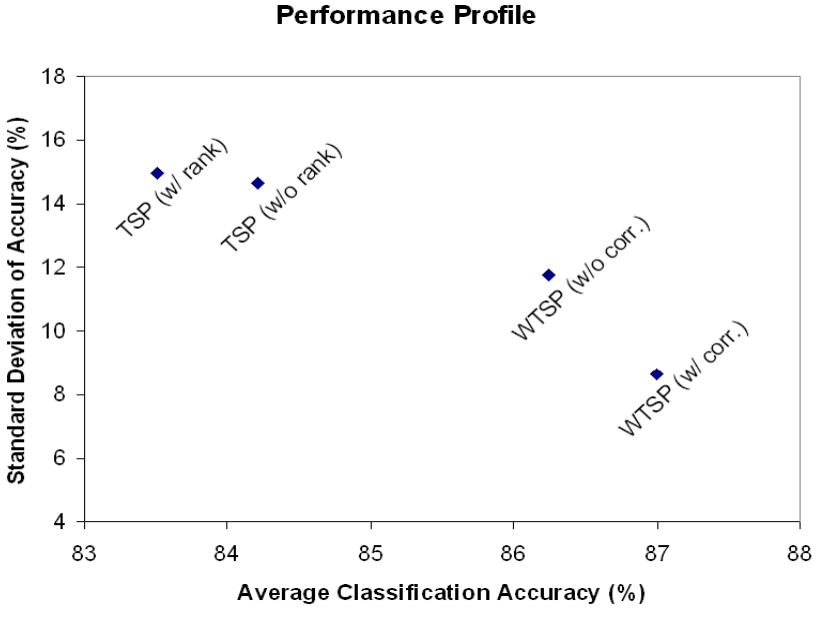


Fig. 1. Performance Comparison of WTSP and TSP methods. Each method is represented as a point in this figure with the coordinates composed of average classification accuracy and its standard deviation. The performance of WTSP (w/ corr.) is the best among these methods with highest average accuracy (87.00%) and smallest standard deviation (8.63%). The WTSP (w/o corr.) takes the second place with average accuracy (86.24%) and standard deviation (11.76%). The TSP-based methods perform worse compared to WTSP-based methods with lower average accuracies (84.21% for TSP without rank and 83.51% for TSP with rank) and relatively larger standard deviation (respectively, 14.66% and 14.98%).

GCM, Prostate and p53 data), the proposed WTSP-based methods significantly improve the performance. The increase of the accuracy are respectively 9.24% for GCM data, 19.32% for Prostate data and 3.11% for p53 data.

Figure 1 shows the performance comparison between the TSP-based and WTSP-based methods. The average accuracies of classification and their standard deviations for each method are calculated. The method which could achieve higher classification accuracy with lower standard deviation is desired (Ideally, a method with 100% accuracy and 0 standard deviation is the best). If the standard deviation of the accuracy is plotted against its average classification accuracy for each method as shown in this figure, it can be seen that when a method represented by a point in this figure is closer to the bottom-right corner, its performance will be better.

From this figure, it is clearly seen that the WTSP (w/ corr.) works best with the highest average accuracy of 87.00% and smallest standard deviation of 8.63%, followed by WTSP (w/o corr.), TSP (w/o rank) and TSP (w/ rank). The improvement of WTSP method comes from the fact that it minimizes the probability of total misclassification and hence it may choose different gene pairs than the ones chosen by the original TSP method for classification.

4 Discussion and Conclusion

Based on the classical TSP method, we proposed a weighted TSP (WTSP) method for the supervised gene selection and classification scheme. This WTSP method is a generalization of the original TSP method. Different from the TSP method which actually minimizes the sum of misclassification probabilities over two classes, the WTSP minimizes the cost of misclassification or the probability of total misclassification by incorporating the probability of each class in the data. The results obtained from experimental microarray data sets suggest that the WTSP could achieve higher classification accuracy compared to the TSP method. In addition, the WTSP method also simplifies the classification rule by considering the order of the genes in the gene pair. Besides that, the WTSP method possesses the advantages of TSP method such as achieving high classification accuracy with few genes and invariant to the preprocessing.

At this stage, it is difficult to arrive at a conclusion about at what specific conditions the proposed WTSP method can always work significantly better than the original TSP method. The proposed WTSP method aims to handle the problem of the unbalanced sample size in each class. This problem exists in all the 8 data sets we tested. However, for some data sets, the TSP still works comparatively as well as the WTSP. The possible reason is as follows. If the gene pair which achieves the top score in WTSP have $a_{ij} \approx N_1$ and $b_{ij} \approx 0$ (see Eq. (7)), the same gene pair is likely to be chosen in the TSP method. Thus, for those data sets, both methods can achieve a high classification accuracy as shown in the Table 2 for the cases of Leukemia, Colon, Lung and DLBCL. Whereas, if this scenario does not hold, that is, the TSP method works poorly, the proposed WTSP method should improve the performance more significantly.

This is consistent with the results from the GCM, CNS, Prostate and p53 data. Therefore, although it is difficult to pinpoint the exact situation at which the proposed WTSP works significantly better than TSP, a general conclusion is: the WTSP method performs significantly better than TSP method when the sample size in each class is unbalanced and TSP performs poorly.

Future work may include the investigation of other information contained in the gene-gene interaction. The methods based on TSP exploit one pattern of gene-gene interaction, that is, the relative expression levels of the gene pair revert from one class to another class. There may exist other possible gene-gene interaction patterns, such as the coexpression of two genes. By exploring these possible patterns contained in the microarray data, a more accurate classification method could be developed and the functional biological processes may be revealed.

References

1. Dudoit, S., Yang, Y.H., Callow, M.J., Speed, T.P.: Statistical Methods for Identifying Differentially Expressed Genes in Replicated cDNA Microarray Experiments. Statistica Sinica 12, 111–139 (2002)
2. Bo, T., Jonassen, I.: New Feature Subset Selection Procedures for Classification of Expression Profiles. Genome Biology 3, research0017.1–research0017.11(2002)
3. Kuo, W.P., et al.: Functional Relationships Between Gene Pairs in Oral Squamous Cell Carcinoma. In: Proceedings of American Medical Informatics Association (AMIA) 2003 Symposium (2003)
4. Hanczar, B., Zucker, J., Henegar, C., Saitta, L.: Feature Construction from Synergic Pairs to Improve Microarray-based Classification. Bioinformatics 23, 2866–2872 (2007)
5. Rapaport, F., Zinovyev, A., Dutreix, M., Barillot, E., Vert, J.: Classification of Microarray Data Using Gene Networks. BMC Bioinformatics 8 (2007)
6. Karuturi, R.K.M., Vinsensius, B.V.: Friendly Neighbors Method for Unsupervised Determination of Gene Significance in Time-course Microarray Data. In: Proc. of the Fourth IEEE Symposium on Bioinformatics and Bioengineering. IEEE Press, Los Alamitos (2004)
7. Karuturi, R.K.M., Wong, S., Sung, W.K., Miller, L.D.: Differential Friendly Neighbors Algorithm for Differential Relationship Based Gene Selection and Classification using Microarray Data. In: Proc. of the Intl. Conf. on Data Mining (DMIN 2006), USA (2006)
8. Xiong, M., Jin, L., Li, W., Boerwinkle, E.: Computational Methods for Gene Expression Based Tumor Classification. BioTechniques 29, 1264–1270 (2000)
9. Dai, J.J., Lieu, L., Rocke, D.: Dimension Reduction for Classification with Gene Expression Microarray Data. Stat. Appl. Genet. Mol. Biol. 5, 6 (2006)
10. Furey, T., et al.: Support Vector Machine Classification and Validation of Cancer Tissue Samples using Microarray Expression Data. Bioinformatics 16, 906–914 (2000)
11. Duda, R.O., Hart, P.E., Sork, D.G.: Pattern Classification. John Wiley & Sons, New York (2000)
12. Cover, T.M., Hart, P.E.: Nearest Neighbor Pattern Classification. IEEE Trans. Info. Theo. IT 13, 21–27 (1967)

13. Dudoit, S., Fridlyand, J., Speed, T.P.: Comparison of Discrimination Methods for the Classification of Tumors Using Gene Expression Data. J. Amer. Stat. Asso. 97, 77–87 (2002)
14. Tibshirani, R.O., et al.: Diagnosis of Multiple Cancer Types by Shrunken Centroids of Gene Expression. Proc. Natl Acad. Sci. 99, 6567–6572 (2002)
15. Tan, A.C., Naiman, D.Q., Xu, L., Winslow, R.L., Geman, D.: Simple Decision Rules for Classifying Human Cancers from Gene Expression Profiles. Bioinformatics 21, 3896–3904 (2005)
16. Xu, L., Geman, D., Winslow, R.: Large-scale Integration of Cancer Microarray Data Identifies a Robust Common Cancer Signature. BMC Bioinformatics 8 (2007)
17. Geman, D., d'Avignon, C., Naiman, D.Q., Winslow, R.: Classifying Gene Expression Profiles from Pairwise mRNA Comparisons. Stat. Appl. Genet. Mol. Biol. 3, 19 (2004)
18. Price, N.D., Trent, J., et al.: Highly Accurate Two-gene Classifier for Differentiating Gastrointestinal Stromal Tumors and Leiomyosarcomas. Proc. Natl Acad. Sci. 104, 3414–3419 (2007)
19. Golub, T.R., et al.: Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. Science 286, 531–537 (1999)
20. Alon, U., et al.: 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 (1998)
21. Gordon, G.J., et al.: Translation of microarray data into clinically relevant cancer diagnostic tests using gene expression ratios in lung cancer and mesothelioma. Cancer Res. 62, 4963–4967 (2002)
22. Shipp, M.A., et al.: Diffuse large B-cell lymphoma outcome prediction by geneexpression profiling and supervised machine learning. Nat. Med. 8, 68–74 (2002)
23. Ramaswamy, S., et al.: Multiclass cancer diagnosis using tumor gene expression signatures. Proc. Natl. Acad. Sci. USA 98, 15149–15154 (2001)
24. Pomeroy, S.L., et al.: Prediction of central nervous system embryonal tumour outcome based on gene expression. Nature 415, 436–442 (2002)
25. Stuart, R.O., et al.: In silico dissection of cell-type-associated patterns of gene expression in prostate cancer. Proc. Natl Acad. Sci. USA 101, 615–620 (2004)
26. Miller, L.D., et al.: From The Cover: An Expression Signature for p53 Status in Human Breast Cancer Predicts Mutation Status, Transcriptional Effects, and Patient Survival. Proc. Natl. Acad. Sci. USA 102, 13550–13555 (2005)

Identifying Conserved Discriminative Motifs

Jyotsna Kasturi[1,2], Raj Acharya[2], and Ross Hardison[3,4]

[1] Non-Clinical Biostatistics, Johnson & Johnson Pharmaceutical Research & Development, New Jersey
[2] Department of Computer Science and Engineering
[3] Center for Comparative Genomics and Bioinformatics, Huck Institutes of Life Sciences
[4] Department of Biochemistry and Molecular Biology; Pennsylvania State University
jkasturi@its.jnj.com, acharya@cse.psu.edu, rch8@psu.edu

Abstract. The identification of regulatory motifs underlying gene expression is a challenging problem, particularly in eukaryotes. An algorithm to identify statistically significant discriminative motifs that distinguish between gene expression clusters is presented. The predictive power of the identified motifs is assessed with a supervised Naïve Bayes classifier. An information-theoretic feature selection criterion helps find the most informative motifs. Results on benchmark and real data demonstrate that our algorithm accurately identifies discriminative motifs. We show that the integration of comparative genomics information into the motif finding process significantly improves the discovery of discriminative motifs and overall classification accuracy.

Keywords: Discriminative motifs, regulatory elements, comparative genomics, classification, Naïve Bayes, mutual information.

1 Introduction

One of the challenges of post-genomic molecular biology is to understand gene regulation and the complex mechanisms underlying gene expression. In protein-coding genes, gene regulation occurs by altering the rate of transcription of the gene thereby changing its expression levels. Transcriptional regulatory proteins exert control on the expression of genes in a cell by binding to specific DNA regulatory elements, in close proximity to the transcription start site of a gene. Computational methods have been quite successful in identifying patterns in DNA sequences as putative transcription factor binding sites, especially in bacteria and yeast. These methods typically make use of available gene expression measurements either to guide the search for motifs within DNA sequences [2], [3], [5], [13], or in a combined fashion by correlating gene expression data with the sequences [6], [7].

Eukaryotic binding site prediction remains a complex problem and a challenging one for several reasons. The first is that transcription factors often bind to regions of the DNA several kilobases from the gene's transcription start site. These sites may even be present within the introns or downstream of the gene. The factor MEF2 in humans, and GATA1 in mouse and human are examples. Another critical issue is that motif length is unknown and often highly variable between sites although they often

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 334–348, 2008.

have a common core site of smaller length (such as WGATAR). In addition, transcription factors may act cooperatively with other factors to control regulation. Since the length of sequences can be very long, computational identification of putative binding sites can lead to many false positives being detected. Functional regulatory elements are often conserved over evolution and true binding sites will often be present in highly conserved regions of the multi-species sequence alignments. Some motif finding algorithms have used this approach to identify putative binding sites [1], [9].

Motifs are represented using either position weight matrices (PWM) or string-based models, and identified as over-represented patterns common to a given set of sequences relative to a set of background sequences. Methods that model motifs with a probability matrix model [11], [12], [19] are capable of capturing variability between binding sites effectively, but use maximum likelihood and local search techniques to estimate parameters which are extremely dependent on starting conditions, number of iterations and stability of the model to converge to a local maximum/minimum. String-based motif finding algorithms [16], [17] on the other hand are based on basic counting schemes and have the advantage of being deterministic, in the sense that they produce identical results for every run with a fixed set of parameters. However, the motifs identified will vary depending on the choice and estimation method of the background distribution.

Discriminative methods find motifs that distinguish between two sets of input sequences or gene clusters, thereby avoiding the need for a background distribution. A probabilistic logistic regression model was proposed to identify discriminative motifs between two sets of genes, one expected to contain a *cis*-regulatory module (CRM) while the other did not [14]. The algorithm performs well but requires the estimation of an extremely large number of variables, with complexity exponential in the length of the sequences. Also, prior knowledge of CRMs is generally unavailable. DMotifs [17] uses a string-based approach to look for well-distributed motifs over individual promoters, identifying discriminative motifs between a set of positive and negative (background) sequences. The algorithm does not scale to large number of sequences. The DME algorithm [18] also uses an enumerative approach to perform an exhaustive search within a set of candidate matrices; refining the highest scoring matrices and erasing discovered motifs from the data iteratively. These methods cannot handle multiple clusters.

We present a supervised learning approach to identify conserved discriminative motifs between multiple cohorts of genes. An enumerative method is used to model each gene individually based on the distribution of words in the sequence and distinguishing between words with different counts. Words (putative motifs) that do not contribute to the discrimination as measured by a feature selection method are iteratively dropped, resulting in a drastic reduction of the search space and number of candidate motifs. We show that the discriminative power of the identified motifs is increased by 15%-20% when comparative genomics information is incorporated into the algorithm. The design and implementation of the algorithm is specifically targeted to handle very large amounts of data.

2 Methods

Let K be the number of clusters into which the N input genes have been grouped based on gene expression or function similarity. Let S be the set of DNA sequences corresponding to the genes consisting of upstream, intron and downstream regions for each gene. The objective of the discriminative algorithm is to identify a small set of motifs or word patterns that can discriminate between these input gene clusters (Fig. 1). For example, given a set of genes divided into an up-regulated cluster and a down-regulated cluster, we are interested in identifying those motifs that discriminate between the two clusters thereby gaining a better understanding of their individual regulatory mechanisms. Once a relatively small set of putative motifs has been identified using computational means, they may be validated through appropriate in-vivo and in-vitro assays.

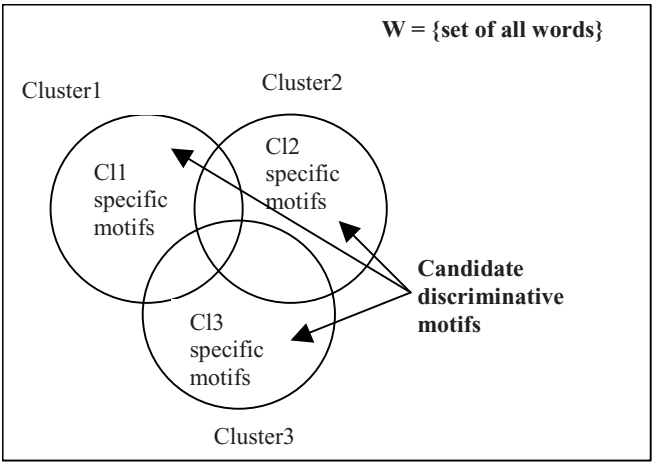


Fig. 1. Venn diagram showing candidate motifs to discriminate between 3 clusters

2.1 The Mathematical Motif Model

Let gene z correspond to a set of sequences (upstream, intron and downstream) denoted by s^z; $z = 1, 2, \ldots, N$. We use a string-based motif model that is capable of distinguishing between distinct words and their frequencies of occurrence. For a specified motif length l, the sequences are scanned using a sliding window of the corresponding size to obtain a list of all the word patterns. In defining the motif model, we distinguish between the terms *token* and *word*. A *token* is a DNA string pattern of specified length l. Let $V=\{t_1, t_2, \ldots, t_{|V|}\}$ denote the token vocabulary for the dataset. Given a sequence, the token t_i may appear with frequency count $n(t_i) >= 0$. Let the model parameter R (>= 1) denote the maximum frequency count considered by the model. The modified count x_i for token t_i is then given by

$$x_i = \begin{cases} n(t_i) & \text{if } n(t_i) \in \{0,1,...,R\} \\ R & \text{if } n(t_i) > R \end{cases} \quad (1)$$

A *word* (potential motif) is then defined as the tuple(t, x) consisting of a token and its modified frequency, where $t \in V$ and $x \in X = \{0,1,2,\ldots,R\}$. Let $W \subset V \times X$ denote the set of all words (vocabulary of words) present in the dataset. Any given sequence s^z may then be characterized using the word vocabulary as (s^z_1, s^z_2, ….,s^z_{Nz}); where $s^z_i \in W$ appears in the sequence (for i =1, 2, …, N_z). This representation of a word distinguishes between two or more clusters containing a common motif but differing in the exact number of occurrences in each of the clusters. Note that there is a word associated with a token not occurring within a gene sequence. Further, the model is invariant of the ordering of words within a sequence.

This model differs from the more commonly used Multivariate Bernoulli (MB) and Multinomial (MN) models frequently used in the context of document analysis [8], in the characterization of a sequence, and the way in which probabilities are assigned. In the MB model, a sequence is represented as a vector of binary attributes obtained as the presence or absence of tokens in the sequence from the token vocabulary. The number of times a word occurs within a sequence is not captured. Observe that the MB model is a special case of the motif model described here for R = 1. On the other hand, the MN model represents a sequence as a vector of token occurrences (frequency counts). Only the words that are present within a sequence are utilized for analyses, ignoring words that are absent. In comparison, we define here a motif model that imposes an upper bound on the frequency count, but takes non-occurring tokens into account. Both the MB and MN models assign a single probability value to each token irrespective of its frequency count, with the probability for non-occurrence calculated by subtracting from unity the probability of occurrence. This is in sharp contrast to our motif model, which allows assignment of any probability values to (token, frequency count) pairs, i.e., different frequency counts for the same token can have probabilities not restricted by any imposing model (the MN model imposes such restrictions). The design rationale behind our motif model is to capture meaningful frequency count information and also to treat tokens with different frequency counts independently to achieve high classification accuracy using a small number of words/tokens.

2.2 The Naïve Bayes Classifier

Let the K clusters be represented by the random variable C taking values $\{c_1, c_2,\ldots, c_K\}$, with prior probabilities denoted by $p(C_1)$, $p(C_2)$, …, $p(C_K)$. The Naïve Bayes model is *generative* in the sense that the word likelihood probabilities are first empirically estimated for each class using class prior probabilities and the dataset. It uses an easy to compute linear classifier to predict class membership for unknown sequences and has been shown to perform well in practice, especially suited to large datasets.

The likelihood probability of a word given a class, $p(w_i|C)$ based on the m-estimate [10] is estimated as $p(w_i \mid C_j) = (n_i^C + mp) / (|C| + m)$ where $n_i^{C_j}$ denotes the number of sequences in class C_j with word w_i occurring x_i times; $|C|$ is the total number of training examples in class C_j; p is the prior probability of words taken to be uniform = $1/|V|$;

and m is a constant called the 'equivalent sample size' which determines how heavily to weight p relative to the observed data. Imposing the Naïve Bayes conditional independence assumption, the conditional probability of $p(s^z|C_j)$ is a product of the individual probabilities $p(s^z \mid C_j) = p(s_1^z, s_2^z, ..., s_{N_z}^z \mid C_j) = \prod_i p(s_i^z \mid C_j)$. The posterior probabilities $p(C_j|s^z)$ can be calculated by a simple application of the Bayes theorem. Given a new sequence s^*, the classifier may be used in conjunction with the *maximum a posteriori* or *MAP* decision rule to assign it to a cluster.

$$classify(s^*) = \arg\max_j p(C_j) \prod_{i=1}^{|N_*|} p(s_i^* \mid C_j) \tag{2}$$

The independence assumption applied on the predictor variables, although not always accurate, simplifies the classification task dramatically by allowing the class conditional densities $p(x_k|C_j)$ to be calculated separately for each variable. In effect, Naive Bayes reduces a high-dimensional density estimation task to a one-dimensional kernel density estimation. The assumption does not seem to greatly affect the posterior probabilities, especially in regions near decision boundaries, thus leaving the classification task unaffected. If higher order interactions were to be examined, such as with Bayesian networks, the complete joint probability with size of vocabulary = $|V|*(R+1)$, $p(w_1, w_2, \ldots, w_n|Cj)$ has run-time exponential in the size of the vocabulary times the number of classes, $O(|S|^{|V|} * |C|)$. This is impractical for interactions of more than order 3, implying that it would be infeasible to identify CRMs with more than 3 combinatorial acting motifs. On the other hand, with the conditional independence assumption and assuming that the classification in independent of the positions of words (use same parameters for each position), the training time of the algorithm has order $O(|S|L_S + |C||V|)$ where L_S is the average length of each sequence. Even when the number of sequences examined is small, their lengths are usually at least 1kb or higher for eukaryotes and the number of unique words is quite high. Combining the two DNA strand orientations usually reduces the vocabulary size by $(1/3)^{rd}$, but is still higher than the number of sequences being examined. The test time for classification of new sequences takes $O(|C|L_t)$ time, where L_t is the average length of the test sequence.

2.3 Feature Selection to Identify Discriminative Cluster-Specific Motifs

The number of distinct words can be very large (in the order of $|V|*(R+l)$) depending on the number of genes and sequence lengths, the maximum stored token frequency and the motif length. Though good classification performance can often be achieved by considering all the words or even a large number (~1000) of words, it is not useful in understanding the regulatory mechanism underlying the differences in the gene clusters. Instead, we use a feature selection method to rank words by assigning scores to each word measuring its classification efficacy. The *Mutual Information MI* between two random variables measures the amount of information that the value of one variable gives about the value of the other [4] and has been shown to perform well in text

mining contexts [8]. $MI(X, Y) >= 0$ and equality holds if and only if the random variables X and Y are independent of each other. Discriminative motifs are selected from amongst the words that contribute significantly to the entire clustering. For each word $w \in W$, a new random variable $Y(w)=\{0,1\}$ is defined to represent the occurrence or non-occurrence of a word. The probabilities are estimated from the data as $p(y=1, C_j) = p(w, C_j)$ and $p(y=1) = p(w)$; while $p(y=0, C_j) = [1-p(w|C_j)]*p(C_j)$ and $p(y=0) = \sum_j p(y=0, C_j)$. The Mutual information between $Y(w)$ and C is then calculated as

$$MI(Y(w), C) = \sum_{j=1}^{K=|C|} \sum_{y=0,1} p(y, C_j) \log \frac{p(y, C_j)}{p(y) * p(C_j)} \tag{3}$$

2.4 Discriminative Algorithm

We now describe the discriminative algorithm that identifies cluster-specific motifs as the over-represented words within each cluster, that is words that are useful in discriminating a cluster from all the others. The mutual information feature selection criterion is used to score and sort all words in the vocabulary, the highest scores indicating the most informative words. The algorithm (described in Table 1) identifies discriminative motifs by iteratively dropping the least significant words (with zero score or below a specified threshold). Next, the relative score for each word, to the observed maximum score (maxScore), is calculated as [(maxScore – current word score) * 100/maxScore]%. Words with relative scores below a certain threshold of λ% (initialized to some λ_0) are removed from the complete word vocabulary W. If no word meets this condition indicating that the threshold is too stringent, the threshold is lowered by 5% and the removal step repeated. To avoid removing all the words in the vocabulary, user-defined value may be used to indicate the minimum number of words to be retained. The classifier is then re-trained using the new word vocabulary and the process repeated. The algorithm terminates when the desired number of discriminative motifs or a certain performance threshold has been reached.

Table 1. Algorithm to identify discriminative motifs

For each word w in the word vocabulary W,
(the occurrence of each word taken to be independent $\{0,1,\ldots,R\}$)
1. Get word counts from the data sequences.
2. Obtain estimates for likelihood and posteriors.
3. Calculate scores $MI(w_i, C_j)$.
4. Drop words with the lowest scores.
5. Use the remaining words to define the set of candidate motifs.
6. Classify using Naïve Bayes classifier.
7. Calculate classification accuracy (%) on a set of test sequences (with known classes).

Repeat steps 2 to 7 until error = (100-accuracy) < ε (small) or a small number of candidate features or motifs are obtained.

2.5 Constrained Enumeration Incorporating Sequence Conservation

The availability of diverse genomic databases allows the integration of these data into the motif finding process. Comparative genomics information has been used in some motif finding algorithms [1], [9], with the premise that functional regulatory elements are often conserved over evolution, thereby showing up in highly conserved regions of the alignments. These algorithms use as input all of the multiple sequence alignments to calculate some kind of scoring to measure the extent of conservation among the sequences. The calculations of these scores are data intensive and time consuming. DNA sequences do not change and can be considered a fixed data source, thereby eliminating the need to compute alignment scores dynamically within the motif finding algorithm. Instead, the same information can be accessed through pre-computed sequence conservation scores such as 'percent identity' or more comprehensive measures such as the PHYLOHMM scores [15]. To incorporate this quantitative measure into the algorithm, the following general procedure is followed.

Let Q denote any measure, such as conservation score for scoring the positions of a sequence, giving a higher score to regions of interest and a lower score to others. This information is incorporated into the motif finding process by considering sequence positions above a high preset threshold τ when scanning a sequence to extract tokens, $Q > \tau$. This constrains the enumeration of words, thereby influencing the likelihood probabilities based on the score information. The rest of the motif finding procedure remains the same.

2.6 Scalability and Implementation

Motif finding for higher eukaryotes like rodents and humans is a significantly greater challenge than for yeast or bacterial genomes in that transcription factor binding sites may be present at any distance from the transcription start site. Typically we will use a threshold of 3000bp to restrict the length of upstream and downstream sequences, but retain the complete intron sequences. The current implementation of the algorithm is designed to handle vast amounts of data - large number of genes and long sequences. The software is written in Java and connects to a MySQL database for input data and runtime storage. The PhyloHMM scores for the mouse genome require approximately 70GB of disk space. A typical run to enumerate about 360 genes with 4363 sequences (sequence lengths around 100kb) takes approximately 5 minutes without conservation and 15 minutes with conservation on an AMD Athlon XP 3000+ power with 1GB RAM.

3 Experimental Data and Results

3.1 Motif Assessment Data

The performance of our discriminative algorithm is assessed using benchmark data for mouse comparing the results with 13 other computational motif-finding tools [20]. The data consists of a total of 36 datasets with real binding sites from TRANSFAC planted within the sequences at their known positions and orientations, each dataset

consisting of one of three different types (12 datasets each) of background sequence namely (i)binding sites from real promoter sequences (called 'real'), (ii)randomly chosen promoter sequences from the same genome (called 'generic'), and (iii)sequences generated by a Markov chain of order 3 (called 'markov'). The prediction of only a single motif was allowed to be used in the comparisons for each dataset, using various statistical measures (*nSn, nPPV, nPC, nCC, sSn, sPPV,* and *sASP*) to assess the correctness of the predictions by comparing them with known binding sites (see [20] for more details). It was shown that the removal of the 'real' datasets resulted in an improvement in performance for nearly all the tools, with YMF being the most affected, while MotifSampler was the only tool to perform relatively better on the 'real' datasets than on the others, all species combined.

Discriminative motifs were predicted using the proposed algorithm using a set of background sequences generated from a third-order markov chain calculated with all the non-coding regions of the mouse genome as the second cluster. Three trials were performed for each dataset changing the background sequences. Fig. 2A summarizes the results obtained for all mouse data taken together and Fig. 2B compares the correlation measure *nCC* for each data type individually. Our method outperforms all the other tools on 'real' data with an *nCC* value of 0.114 (with nSN = 99) compared to the highest values of 0.1 and 0.08 (nSN = 50 and 34) achieved by MEME and MEME3 respectively and MotifSampler (nSN = 9) having a low negative value, indicating that it does not perform well on 'real' mouse data. Results on the 'generic' and 'markov' datasets could be poor due to the fact that the number of sequences within which to search for motifs is too few and the markov model probabilities were based on all noncoding sequences from the mouse genome rather than restricting it to use a maximum of 3000bp upstream of genes as was used to create the data. Hence, cluster discrimination is a better way to find motifs rather than relative over-representation over background, clearly sensitive to the model used to generate background sequences.

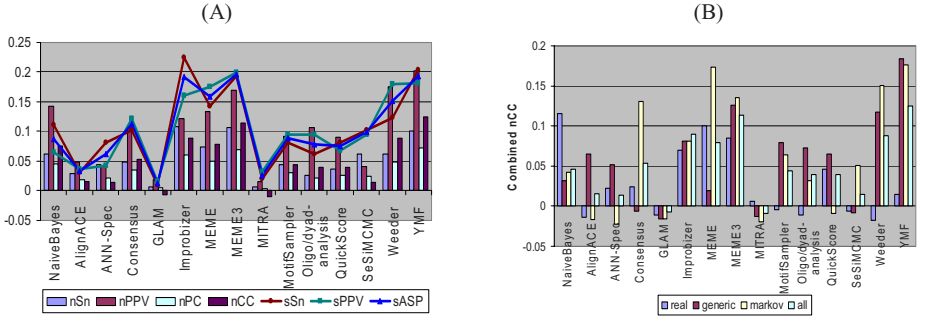


Fig. 2. Statistical measures of accuracy of the tools on the mouse benchmark data comparing 13 motif finding algorithms (MEME and MEME3 are considered one method with different parameters) (A) Combined measures of correctness over all 36 datasets for mouse. (B) Combined Correlation Coefficient (*nCC*) by data type.

3.2 Erythroid Differentiation in Mouse

The performance of the discriminative motif finding algorithm on real data is demonstrated using gene expression data of mouse genes studied with a late erythroid maturation model using the G1E line of Gata1-null cells, which are blocked at the proerythroblast stage because of the absence of the Gata1 transcription factor. The expression levels of over 9000 genes were measured at 6 time points after the restoration of Gata-1 [21], available via NCBI Gene Expression Omnibus database (accession GSE628). Genes were classified as being either up or down-regulated indicating that Gata-1 is important in gene repression as well as in the activation of a large number of genes. Two representative gene clusters (down and up-regulated) were picked after first clustering data using the kmeans algorithm with Pearson correlation distance to measure gene expression similarity. The chosen clusters are referred to as cluster1 and cluster2 respectively.

Many genes known to be induced by Gata-1 were present in the up-regulated group such as Alas2, Fog1, Vav2, Hbb-b1 and Hbb-b2. The Gata2 gene was down-regulated during maturation. The discriminative motif finding algorithm was used to identify transcription factors other than Gata-1, which may be responsible for the directional changes in expression levels. Repeat masked sequences from mm5 were used in the motif analysis comprised, taking as input all non-coding regions in and around genes (including UTRs and introns) and regions 3000bp flanking the gene upstream and downstream, gene positions as given by the KnownGenes table (UCSC table browser). Replicated genes (those for which the same Genbank Accession number was repeated in the KnownGenes table) were removed from the analysis and genes with overlapping positions were collapsed into a single gene complex. An examination of the distribution of sequence lengths showed a large amount of variability in the lengths of the introns, with the longest intron being approximately $12x10^4$ bp in length. It is important to note that existing motif finding tools cannot handle such large sequences.

The discriminative algorithm was used identify motifs specific to the up and down-regulated clusters, and classification performance compared for varying word lengths (l = 4, 6, and 8bp), and maximum word count, R from 1 to 10. Fig. 3 shows results obtained comparing classification accuracy with and without the use of conservation information using all words in W. For small values of R, shorter words (length l = 4) produce lowest classification accuracy ranging from 56% to 65% shown in Fig. 3A. Since the number of tokens occurring in the sequences (a sequence of length L has L-l+1 tokens) is large, there are many more common tokens between the sequences thereby producing less discrimination between the clusters. However as the word length increases (l=8) the number of tokens occurring in the sequences is fewer and classification accuracy is slightly better ranging from 60% to 65%. Although there are 4^8 possible distinct tokens that can be present we typically do not see so many in practice. The special case of our model when R=1 is the Multivariate Bernoulli model with lowest classification performance for all word lengths – 55% to 62%. Increasing the maximum word occurrence count, R from 10 to 50 resulted in a steady improvement in the classification performance of about 88% for any word length approximating the multinomial model, but resulting in an extremely large number of words $|V|*(l+1)$.

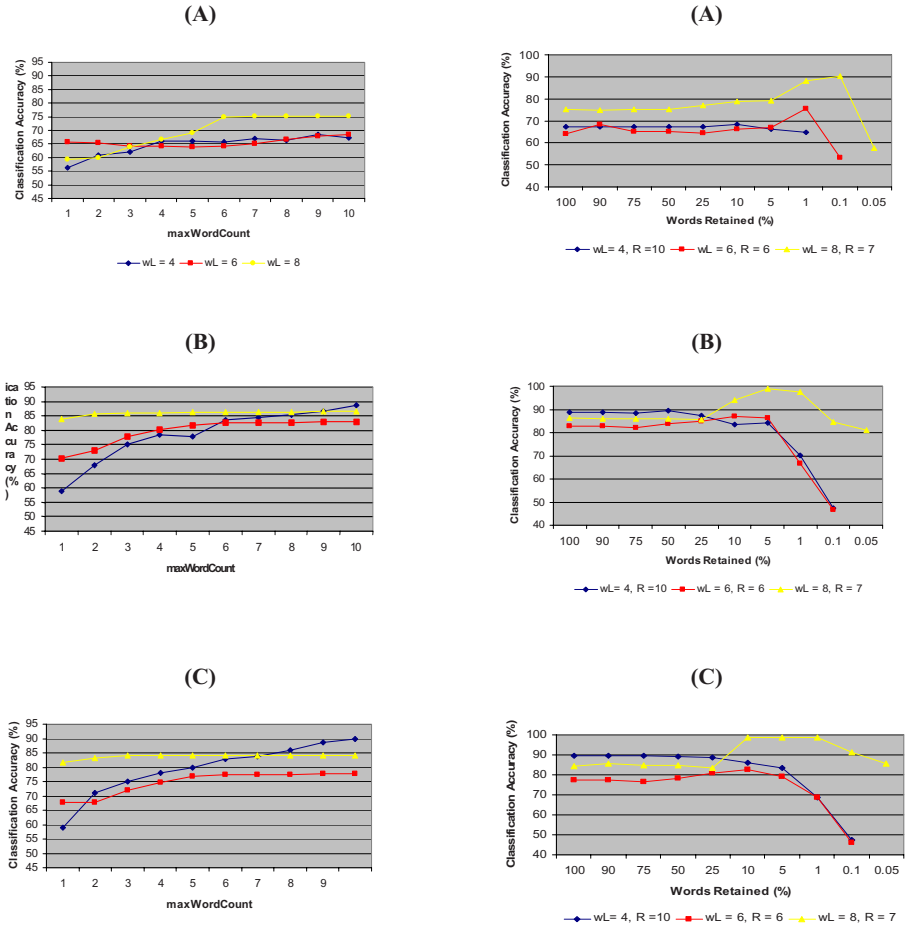


Fig. 3(left) and Fig. 4 (right). Classification accuracy of genes using the entire set of words (*W*) varying word length and maximum word count. (A)Unconstrained enumeration. (B) Constrained using conservation score threshold 0.7. (C) Constrained using conservation score threshold 0.8.

The use of multiple species sequence conservation in motif identification is evaluated next. We compare results with the constrained enumeration of words using thresholds 0.7 and 0.8 for PhyloHMM conservation scores for word lengths 4, 6 and 8 and varying *R* shown in Fig. 3B and C. A significant improvement by approximately 5% in classification accuracy is seen for word lengths 4 and 6, and nearly 20% for word length 8, for the worst case accuracy case (for *R*=1). As *R* increases, the accuracy is 90% for only *R*=10 when sequence conservation is used, while without conservation the maximum accuracy of 88% was achieved for *R*=50, clearly showing the sensitivity and performance quality of our word occurrence model. Even for a small word length of 4, the use of conservation scores achieves significant accuracy as the maximum word count increases. For word lengths 4 and 8 we observe that there is a

slight improvement in performance when a higher sequence conservation score threshold is used, while a word length of 6 has a reduction in accuracy from 84% to 78%. Using too few words for classification causes the reduction in accuracy. The same trend is observed when conservation information is used.

The effect of dropping words using the Mutual Information feature selection score is examined next. Fig. 4 shows that when words are dropped from 100% of the word vocabulary, W to 0.01% for a few word lengths and specific values of *R* the general trend is for the classification accuracy to increase significantly by steadily dropping words reaching a maximum value and then with a sharp drop when the number of words used for classification is further reduced. For example, for word length 8 and R=7, dropping 90% of the words (263160 distinct words) to 0.1% of words (263 distinct words) there is an improvement in accuracy of 15% from 75.28% to 90.28%.

Final motifs were returned for the top 50 scoring words. Typically, binding site sequences are variable, such that several words in the enumerative vocabulary may correspond to a single binding site. To consolidate the final list of motifs obtained and account for some of this variability, words differing in two or less positions were collapsed into a single unique motif (predicted binding site). For this, the similarity between every pair of words is required to be calculated, introducing the concept of a *distance* between words. The *Hamming distance* is one such measure of similarity between two strings, where it is calculated as "the number of positions in which the two strings differ, i.e., have different characters". This distance can be calculated only for strings of equal length. A more general and sophisticated measure of distance between strings is the *Levenshtein distance* and is defined for strings of arbitrary length. It counts the differences between two strings, where differences are counted not only when strings have different characters but also when one has a character whereas the other does not. The Levenshtein distance is what we use here to consolidate the final set of 50 words into a set of predicted motifs as listed here. Variable words are consolidated with other words only if they have equal word occurrence counts (described in the motif model). The consolidated motif is represented using the IUPAC nomenclature for nucleic acid representation.

The list of consolidated motifs is then matched against a list of known sites in human promoters [22] and shown in the following series of tables. Each consolidated motif is listed along with the motif for the matched known sites, the transcription factor to which it is known to bind, the word occurrence count, the corresponding mutual information score and the cluster for which the motif is discriminative. Table 2 lists the "most" discriminative motifs obtained using the discriminative algorithm with dropping words and without the use of conservation information. Tables 3 and 4 list the best set of motifs predicted when conservation information is incorporated into the feature selection model with conservation score (phylohmm) thresholds of 0.7 and 0.8 respectively. Conservation scores are based on mouse-centric multiple sequence alignments of human, mouse, rat and dog.

Several interesting aspects of the data emerge from the results of discriminative analysis. GATA1, which is a known transcription factor involved in erythroid differentiation and based on the experimental design, is most likely present in sequences of all genes irrespective of their cluster membership, is not identified. Clearly this binding site is not a candidate for discrimination between clusters and hence not detected by the algorithm. It can be observed that the integration of conservation information

Table 2. Discriminative Motifs without conservation

Factor	Known Site in Human	Predicted Site	WordCount	MI score	Discrimative for cluster
GABP	vCCGGAAGnGCR	CGGAAG	8	0.013196443	1
NCX	GTAAKTnG	GTAAAT	1	0.02081526	2
TBP	TATAAATW	GCTATAAA	5	0.0053224904	2
AREB6	WCAGGTGWnW, AbWCAGGTRnR	TCAGGTAA	5	0.0053224904	2
BACH2	SRTGAGTCAnC	TAGAGTCA	5	0.0053224904	2
NF-AT	WGGAAAnW	AGGAAA, CGGAAG	6, 8	0.010588359, 0.013196443	2, 1
CAC-BP	GRGGSTGGG	GGAGGTGG	6	0.008171844	2
LEF1	CTTTGA	CATTGA, CGTTGA	9, 6	0.0082450025, 0.007071697	1
MYC	SCACGTG	CATGTG	8	0.00951042	1
MYOD	RnCAGGTG	CATGTG	8	0.00951042	1
AP-4	GCAGCTGnY	CATGTG	8	0.00951042	1
SREBP-1	ATCACGTGAY	CATGTG	8	0.00951042	1
STAT5A,IY	AWTTCY, AWTTTCC	ATTTAC	8	0.02081526	2
AP-1, ATF-1	CTGASTCA, TGACGTCARRG	TAGAGTCA	5	0.0053224904	2
MAZ	GGGGAGGG	GGGAGGAT	4	0.0053224904	2
TAL-, ALPHA/E47	AACAGATGKT	CCAGATGT	5	0.0053224904	2

Table 3. Discriminative Motifs with conservation score threshold 0.7

Factor	Known Site in Human	Predicted Site	Word Count	MI score	Discrimative for cluster
NCX	GTAAKTnG	TGTAATTT	2	0.007647609	1
AP-4	GCAGCTGnY	CGAGCTGC	1	0.0117140515	1
		CAGCTG	2	0.0024077317	1
PU.1	WGAGGAAG	GAAGGAAG	2	0.019509021	2
FOXO1	RWAAACAA	CTAAACAG	1	0.0068486626	1
SF-1	TGRCCTTG	GACCTT	2	3.2390753E-6	2
MYOD	RnCAGGTG	GAGGTG	5	0.0024077317	1
AREB6	WCAGGTGWnW AbWCAGGTRnR	GAGGTG	5	0.0024077317	1

causes a smaller number of identified putative regulatory motifs to be matched with the list of “known binding sites” and with fewer word occurrence counts, when compared to the case where this data is not used in the analysis. It is generally believed that most “real” binding sites (TFBS) do not occur with high frequency within the gene’s non- coding sequence regions. It does appear to clearly indicate from our results that the addition of comparative genomics data causes a fewer number of false positives to be identified, keeping in mind the methods used to create the consensus sites and match with the “known sites” list.

Table 4. Discriminative Motifs with conservation score threshold 0.8

Factor	Known Site in Human	Predicted Site	Word Count	MI score	Discrimative for cluster
NCX	GTAAKTnG	GTAATTTT	1	0.010752671	1
		GTAATT	2	0.004678426	1
TBP	TATAAATW	GGTATAAA	1	0.009876572	1
AML1	ACCACA	ACCACA	3	0.0043065688	2
POU6F1	GCATAAWTTAT	ATTTAT	10	0.0049973438	1
DBP	GTdTGCT	TTTGCT	5	0.0047480455	2
CAC-BP	GRGGSTGGG	GGGTGG	4	0.0047480455	2
AP-4	GCAGCTGnY	CGAGCTGC	1	0.008132085	1
		CTAGCTGC	1	0.0072640753	1
NF-AT	WGGAAAnW	TGTAAA	10	0.0049973438	1
LEF1	CTTTGA	CTTTGT	4	0.0049973438	1
ER	RnnnTGACCT	GGACCT	2	0.004978211	1
STAT5A IY	AWTTCY AWTTTCC	GATTTC	1	0.00625898	2
TCF-4	WTCAAAGS	ACAAAG	4	0.0049973438	1
HNF-1	GGTTAATnWTTAMC	GTTA	6	0.003968242	1

There is only one predicted motif GTAAAT that matches the site for transcription factor NCX that is present in the list with a word occurrence count of 1 and is discriminative for the up-regulated cluster, cluster 2. Another NCX binding motif GTAATTTT is also identified when conservation information is used, but is however discriminative for the down-regulated cluster, cluster1. Literature indicates that NCX gene or *neural crest homeobox*, encodes a homeobox containing transcription factor that belongs to the Hox11 gene family, is involved in the activation of genes, and has been associated with diseases such as T-cell leukemia and Neuroblastoma. AP-4 or TFAP4 is another factor identified in all three results. This is a transcription factor of the basic helix-loop-helix-zipper (bHLH-ZIP) family contain a basic domain, which is used for DNA binding, and HLH and ZIP domains, which are used for oligomerization. Transcription factor AP4 activates both viral and cellular genes by binding to the symmetrical DNA sequence CAGCTG. Comparing the predicted motif within the three tables, it can be observed that the motifs CGAGCTGC and CTAGCTGC obtained when using conservation information are more similar to the known site in human with motif GCAGCTGnY when compared to that identified without the use of conservation scores – namely CATGTG. AP4 is also known to have functional interaction with AP-1 (a factor also identified by our algorithm).

Comparing results with and without sequence conservation, we see that factors AREB6 and MYOD are also identified at a conservation score threshold of 0.7, while TBP, NA-AT, and LEF1 are identified with a threshold of 0.8 implying higher degree of conservation over evolution. It is also interesting to see that some motifs that are discovered when conservation at a threshold value of 0.8 is used, such as AML1 and SF-1 do not appear in the motifs when conservation is not used. AML1 or runt-related transcription factor 1, RUNX1 is known to be associated with leukemia or as the name suggests, acute myeloid leukemia. The integration of conservation into the

analysis predicts many other interesting binding sites such as PU1, SF-1, and AML1 related to up-regulated genes while FOXO1, POU6F1, DBP, CAC-BP, ER, IY, TCF-4 and HNF-1 are associated with the down-regulated genes.

4 Conclusion

A new motif-finding algorithm is presented for multi-class discrimination. The motif model takes into account the frequency of token occurrence in individual sequences allowing for a very sensitive categorization of sequence clusters. Discriminative motifs are identified using an information-theoretic feature selection strategy and their prediction power examined with a supervised classifier. The algorithm is defined within a generic framework that allows the easy integration of additional genomic data, showing that comparative genomics information can be used to validate and evaluate the performance of the identified motifs. Results on benchmark and real data demonstrate the performance of our method in identifying true motifs. We have also provided strong empirical evidence to show that comparative genomics significantly improves the classification accuracy and achieves superior motif discovery.

References

1. Blanchette, M., Schwikowski, B., Tompa, M.: An exact algorithm to identify motifs in orthologous sequences from multiple species. In: Proc Eighth Intl Conf Intelligent Systems Mol Biol (ISMB), pp. 37–45. AAAI Press, Menlo Park (2000)
2. Bussemaker, H.J., Li, H., Siggia, E.D.: Regulatory element detection using correlation with expression. Nat. Gen. 27, 167–171 (2001)
3. Cardon, L., Stormo, G.: Expectation maximization for identifying protein-binding sites with variable lengths from unaligned DNA fragments. J. Mol. Biol. 223, 159–170 (1992)
4. Cover, T.M., Thomas, J.A.: Elements of information theory. Wiley, New York (1991)
5. Fickett, J.W., Wasserman, W.W.: Discovery and modeling of transcriptional regulatory regions. Curr. Opinion in Biotechnology 11, 19–24 (2000)
6. Holmes, I., Bruno, W.J.: Finding regulatory elements using joint likelihoods for sequence and expression profile data. Amer Assoc for Artificial Intelligence (2000)
7. Kasturi, J., Acharya, R.: Clustering of Diverse Genomic Data using Information Fusion. In: Proc ACM Sym Applied Computing (Bioinformatics Track) (2004)
8. McCallum, A., Nigam, K.: A comparison of event models for naïve bayes text classification. In: Proc. AAAI (1998)
9. McGuire, A.M., Church, G.M.: Predicting regulons and their cis-regulatory motifs by comparative genomics. Nucleic Acids Res. 28(22), 4523–4530 (2000)
10. Mitchell, T.: Machine Learning, ch. 10. McGraw Hill, New York (1997)
11. Lawrence, C., Reilly, A.: An expectation maximization (EM) algorithm for the identification and characterization of common sites in unaligned biopolymer sequences. Proteins 7, 41–51 (1990)
12. Liu, X., Brutlag, D.L., Liu, J.S.: Bioprospector: discovering conserved dna motifs in upstream regulatory regions of co-expressed genes. Pac. Sym. Biocomp., 27–38 (2001)
13. Roth, F.P., Hughes, J.D., Estep, P.W., Church, G.M.: Finding DNA regulatory motifs within unaligned noncoding sequences clustered by whole-genome mRNA quantitation. Nature Biotech. 16, 939–945 (1998)

14. Segal, E., Barash, Y., Simon, I., Friedman, N., Koller, D.: From promoter sequence to expression: a probabilistic framework. In: RECOMB (2001)
15. Siepel, A., Haussler, D.: Combining phylogenetic and hidden Markov models in biosequence analysis. In: Proc. Seventh Annual Intl. Conf. Comp. Mol. Biol. (RECOMB), pp. 277–286 (2003)
16. Sinha, S., Tompa, M.: A statistical method for finding transcription factor binding sites. Amer. Assoc. Artificial Intelligence (2000)
17. Sinha, S.: Discriminative motifs. J. Comput.Biol. 10(3-4), 599–615 (2003)
18. Smith, A.D., Sumazin, P., Zhang, M.Q.: Identifying tissue-selective transcription factor binding sites in vertebrate promoters. Proc. Natl. Acad. Sci. USA. 102(5), 1560–1565 (2005)
19. Thijs, G., Kathleen, M., Yves, M.: A Gibbs Sampling method to detect over-expressed motifs in the upstream regions of co-expressed genes. In: RECOMB (2001)
20. Tompa, M., Li, N., Bailey, T.L., Church, G.M., De Moor, B., Eskin, E., Favorov, A.V., Frith, M.C., Fu, Y., Kent, W.J., Makeev, V.J., Mironov, A.A., Noble, W.S., Pavesi, G., Pesole, G., Regnier, M., Simonis, N., Sinha, S., Thijs, G., Van Helden, J., Vandenbogaert, M., Weng, Z., Workman, C., Ye, C., Zhu, Z.: Assessing computational tools for the discovery of transcription factor binding sites. Nature Biotech. 23(1), 137–144 (2005)
21. Welch, J.J., Watts, J.A., Vakoc, C.R., Yao, Y., Wang, H., Hardison, R.C., Blobel, G.A., Chodosh, L.A., Weiss, M.J.: Blood. 104(10), 3136–3147 (2004)
22. Xie, X., Lu, J., Kulbokas, E.J., Golub, T.R., Mootha, V., Lindblad-Toh, K., Lander, E.S., Kellis, M.: Systematic discovery of regulatory motifs in human promoters and 3' UTRs by comparison of several mammals. Nature 434(7031), 338–345 (2005)

Exploratory Data Analysis for Investigating GC-MS Biomarkers

Ken McGarry[1], Kim Bartlett[1], and Morteza Pourfarzam[2]

[1] School of Pharmacy, City Campus, University of Sunderland, SR1 3SD, UK
[2] Royal Victoria Infirmary, Department of Clinical Biochemistry, Newcastle Upon Tyne, NE1 4LP, UK

Abstract. The detection of reliable biomarkers is a major research activity within the field of proteomics. A biomarker can be a single molecule or set of molecules that can be used to differentiate between normal and diseased states. This paper describes our methods to develop a reliable, automated method of detecting abnormal metabolite profiles from urinary organic acids. These metabolic profiles are used to detect Inborn Errors of Metabolism (IEM) in infants, which are inherited diseases resulting from alterations in genes that code for enzymes. The detection of abnormal metabolic profiles is usually accomplished through manual inspection of the chromatograms by medical experts. The chromatograms are derived by a method called Gas Chromatography Mass Spectrometry (GC-MS). This combined technique is used to identify presence of different substances in a given sample. Using GC/MS analysis of the urine sample of the patient, the medical experts are able to identify the presence of metabolites which are a result of an IEM.

1 Introduction

The recent advances in bio-medical screening technologies has enabled the development of the so-called "omics" fields, such as genomics (the study of genes sequences and their regulatory mechanisms); transcriptomics (RNA and gene expression); proteomics (protein expression) and metabolomics (metabolites and metabolic networks). In this paper we are primarily concerned with the generation and analysis of metabolomic data for the early diagnosis of disease. The motivation to diagnose disease states through the use of a non-invasive technique (avoids the need for surgery) such as profiling blood or urine samples for their metabolic contents is highly attractive and desirable. It is evident that metabolomics reflects the operation of the cell since each metabolite is the result of a bio-chemical process.

The main idea behind metabolic profiling is the identification of unique signal peaks to differentiate between two or more groups such as normal and diseased samples. Each peak represents the presence of a metabolite and the expectation is that the presence/absence of certain peaks is indicative between the groups. Although, the search for such biomarkers is highly motivated, there are many difficulties; most studies are performed on small sample sizes (20-100 samples)

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 349–358, 2008.

and contain complex mixtures of metabolites common to all samples. Furthermore the small sample sizes often lead to the "curse of dimensionality" problem so often encountered in computational statistics and machine learning [1].

The study reported in this paper is concerned with the identification of inborn errors of metabolism in infants. The errors are due to a faulty gene that encodes an enzyme which translates or transports a biological product into another [2]. A given IEM (there are several) will lead to a build up of a particular product which is invariably toxic [3]. Early detection of such conditions is critical since they lead to a number of highly delibating conditions symptoms in babies from poor feeding, vomiting and in older children can lead to autism, learning difficulties and mental retardation. Many IEM's can be treated with drugs and special diets. Fortunately, such conditions are rare, the occurrences of each IEM vary but perhaps they affect 1 in 5,000 individuals. GC-MS was first applied to identify diagnostic marker for IEMs by Tanaka [4]. Since then GC-MS has been widely used for separation of complex biological mixtures and identification of their components, and has greatly contributed to the study and characterizations of IEM's [5,6,7].

The aim of this work is to develop a hybrid intelligent system using various pattern recognition and machine learning algorithms which could potentially identify all known IEMs and also be able to state whether a given sample is normal i.e. all metabolite profiles present in the sample are results of a normal metabolism or if the sample has any abnormal metabolites present in it which could then be further analysed by medical experts to see if the abnormal metabolites are the cause of the problems in the patient.

The remainder of this is paper is structured as follows; section two discusses GC-MS data issues and biomarker detection; section three describes the data preprocessing issues specific to GC-MS problems and the computational techniques we use; section four presents related work; section five highlights the experimental setup and our results; finally section six presents the conclusions.

2 GC-MS Data Issues and Biomarker Detection

Metabolomics is one of the most promising technologies developed so far for the analysis of living systems. It is essentially a two-stage process with the GC (Gas Chromatography) stage using a capillary column to separate the molecules, this depends on time according to the molecular weight detecting the time at which a metabolite is released (retention time) and its relative abundance appearance). [8]. At the next we use the MS (Mass-Spectrometer) which breaks each molecule down into a set of fragments (peptides), each one has set of mass/charge ratios in which the peaks correspond to the chemical composition of the metabolites, which enables identification [9,10,11]. Combining the GC data with the MS data prevents metabolite identification problems which can often occur when the using GC or MS separately. GC-MS equipment is now a popular choice in many labs

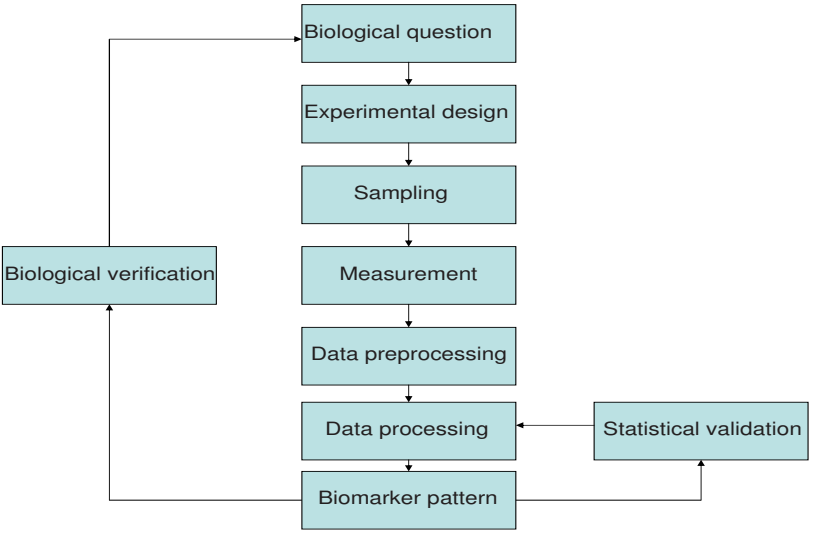


Fig. 1. The stages of biomarker detection (from Smit 2008)

worldwide because their cost have fallen in recent years but their accuracy and reliability has increased.

Irrespective of the hypernated mass spectra techniques used e.g. GC-MS, HPLC, CE-MS, or LC-MS they all suffer from the same difficulties. The main problem is the "curse of dimensionality" [12]. GC-MS is considered as one of the best available methods for forensic substance identification, but the data which is generated is one of very high dimensionality. Fortunately, there is a great deal of redundancy which can be partly managed by data reduction methods such as PCA and ICA. Automated analysis of this data for presence of metabolites is difficult because of this high-dimensionality and other complexities of the data; therefore the search for reliable biomarkers is of great importance [13,14]. A typical workflow model is presented in figure 1, here several stages are required, with data preprocessing playing a major part [15].

As with most data mining endeavors, the data cleansing and preprocessing stage consumes the majority of the effort [16]. Furthermore, if mass-spectra data are to be analysed, then considerable care must be taken to align the spectrograms [17,18]. This was not an issue in the study described in this paper, as the authors are dealing only with the GC-chromatograms, although they are similar visually, in the sense they are peaks displayed against time [19].

In table 1, some of the inborn errors of metabolism are presented (IEM). The first column is the chemical abbreviation of the disorder, the second column indicates a biomarker typically associated with its presence and the third column describes the symptoms if the error goes unrecognised and untreated.

Table 1. Diagnostic biomarkers for inborn errors of metabolism

IEM	Diagnostic Biomarker	Symptoms if untreated
PKU	Phenylalanine hydroxylase - PHE	mental retardation, autistic behavior
MCADD	medium chain acyl CoA dehydrogenase - C8	fasting intolerance, hypoglycemia
3-MCCD	3-Methylcrotonyl CoA carboxylase deficiency	metabolic acidosis and hypoglycemia
SCAD	short-chain acyl-Coa dehydrogenase	austic behaviour, mental retardation, impairment of neurophysical development

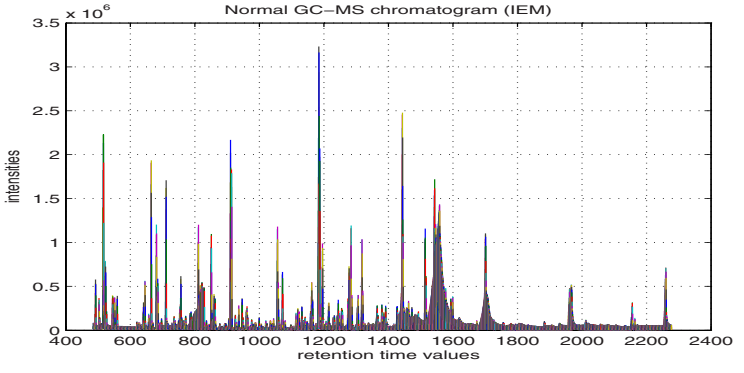


Fig. 2. GCMS Normal total chromatogram

3 Data Preprocessing and Analysis

The data were generated by a Agilent 6890N GC coupled to a mass selective detector and an Agilent ChemStation was used. Helium was used as carrier gas with an average linear velocity of 37cm/s. Inlet pressure was 8.44 psi. The injector split ratio was set to 1:40. Injector and interface temperatures were set 265C and 300C respectively. Samples were analysed using temperature programming (4min isothermal at 65C, 6C/min to 275C followed by 2min isotheraml at 275C). The mass spectra were acquired over the range 40-550 m/z at 3scans/min. The total run time was 38 min. The data set is composed of 22 normal samples and 13 abnormal samples and is in the netCDF le format. The data were converted from netCDF format into Matlab structures using the SNCTools package.

Principal components analysis (PCA), is useful in high-throughput proteomics experiments because of the intrinsic redundancy of the data. PCA involves the transformation of the original dataset into a smaller subset with fewer variables that are uncorrelated. The principal components identify where the maximum variance of the data occurs, each component is an axis in multidimensional space.

Equation 1 describes the first principal component of the data, y_1 is the linear combination.

$$y_1 = a_{11}x_1 + a_{12}x_2 + ... + a_{1p}x_p \tag{1}$$

Equation 2 describes the second principal component of the data, and the variance is constrained by $\mathbf{a_2'x}$ which ensures that y_1 and y_2 are uncorrelated.

$$y_2 = a_{21}x_1 + a_{22}x_2 + ... + a_{2p}x_p = \mathbf{a_2'x} \tag{2}$$

Prior to using PCA, we normalised the data by dividing each variable by its standard deviation. Normalisation is usually good practice, it is essential in this application because the variances of the raw data are considerable.

We used the `princomp` function from the Matlab statistics toolbox to process the data from the original 34 (samples) x 5221 (data elements) to 34 x 20 matrix. Figure 4 shows the Pareto chart with a scree plot of the percent variability explained by each principal component, for the first 10 principal components. The 1st component accounts for only 18% of the variance, and the next 9 components each account for perhaps 10-6% of the remaining variation.

Plotting the two main principal components, as shown in figure 5, we obtain the familiar horseshoe shape. The `biplot` function can help to visualize both the

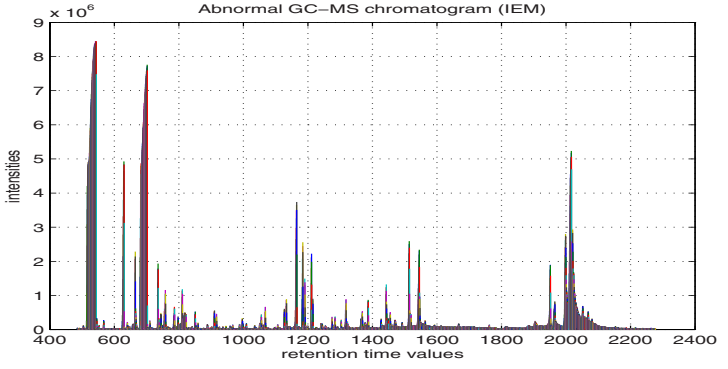


Fig. 3. GCMS Abnormal total chromatogram

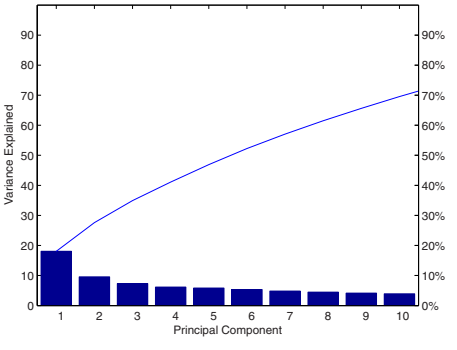


Fig. 4. Pareto diagram showing of the variances explained by the principal components

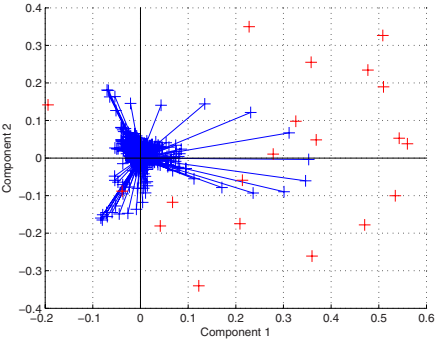


Fig. 5. Biplot principal component coefficients for each variable and the principal component scores for each observation

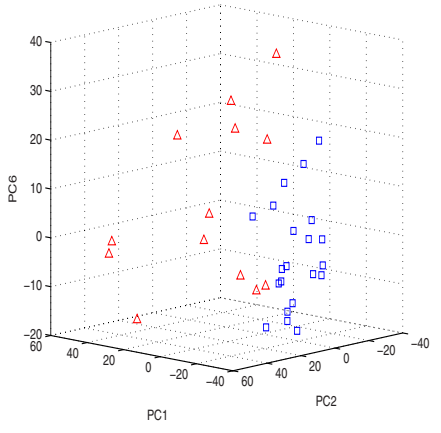


Fig. 6. K-means clustering with two clusters

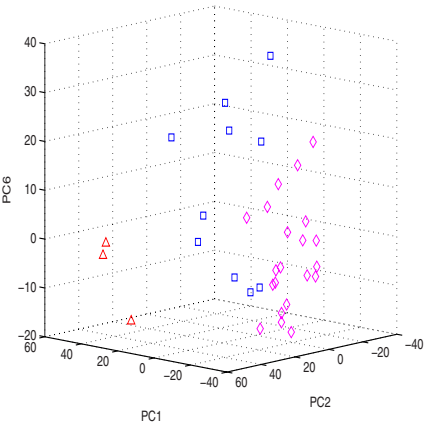


Fig. 7. K-means clustering with three clusters

principal component coefficients for each variable and the principal component scores for each observation in a unified diagram [20]. The length of the lines indicates the variances of the corresponding components and the angles between them show the size of their correlations i.e. small angles indicate high correlations.

K-means clustering was used to test the degree of discriminability between two classes (normal and abnormal), the 34 samples with 20 principal components were clustered as shown in figure 6. This shows the grouping for two clusters, which is the naturally occurring partition for our data set. We were interested in the groupings that would be formed and if each sample would be correctly allocated to the appropriate cluster. K-means clustering is a partitioning method, which operates by partitioning the objects into K mutually exclusive clusters, such that objects within each cluster are as close to each other as possible, and as far from objects in other clusters as possible.

The objective of K-means clustering is to divide the data into k groups so that the within group sum-of-squares is minimized [21]. The number of clusters used must be chosen in advance. We define the within-group data matrix by:

$$\mathbf{S}_W = \frac{1}{n} \sum_{j=1}^{g} \sum_{i=1}^{n} I_{ij}(x_i - x_j)^T \tag{3}$$

Where: Iij is unity when x_i belongs to group j and zero if not, g is the number of clusters. The sum of the diagonal elements S_W to be minimized is given by $S_W = \sum S_{W_i j}$.

Table 2. False Positive and True Positive rates for K-means clustering (2 clusters)

Abnormal TP	Abnormal FP	Normal FP	Normal TP
1, 3, 4, 6, 7 9, 10, 11 12, 13	15, 18, 20 21	1, 2, 5, 8	14, 16, 17, 19, 22, 23, 24, 25 26, 27, 28, 29, 30, 31, 32, 33

Figure 7 shows the grouping for three clusters, it was originally thought that the misgrouped data may reside in a separate cluster. However, in both cases, a number of samples were wrongly allocated.

In table 2, we can see that certain samples were allocated to the wrong clusters. We tried two and three clusters, two clusters is the natural grouping for this data.

4 Model Building and Experimental Results

The initial data analysis and preprocessing appeared encouraging and confirmed that it would be worthwhile to build a classifier model to detect the difference between normal and abnormal samples. From a biological and data mining viewpoint, the most satisfying models are produced by decision trees. The decision tree method is to divide the available data using the class labels into mutually disjoint sets. During the training process, hyperplanes (decision boundaries or surfaces) are generated based upon these data-class tuples, hyperplanes (nodes) are continually added until all of the training data is accounted for.

Structurally, the completed tree is composed of leaf nodes, which correspond to important variables where numerical tests are made, the final terminal nodes correspond to the class labels. The main advantage of the decision tree model is its transparency, it is very easy to see how each class is defined in terms of the variables used and numerical tests made. The training algorithms (there are several) generally use information theory or some form of entropy calculation which tests all the variables for their information bearing potential, thus the first node (variable) is the most important, followed by tests on the remaining variables. The tree is easily converted into rules for use in an expert system.

We developed a decision tree model using 10-fold crossover validation to compute the cost vector. This process partitions the sample into 10 subsamples, each chosen randomly, however with roughly equal size. The decision trees, also have the subsamples in approximately the same class proportions. The Matlab function `treefit` was used, and for each subsample, a tree was fitted to the

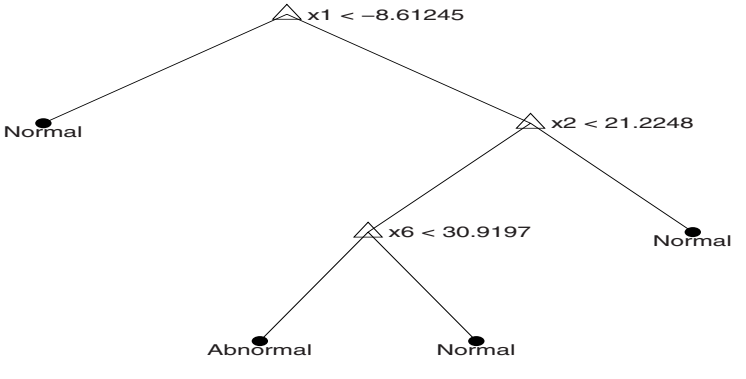


Fig. 8. Decision tree constructed with PC components

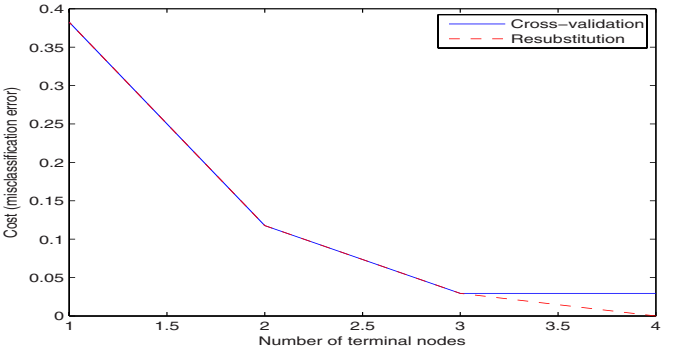


Fig. 9. Decision tree cross-validation

remaining data and then used to predict the subsample. This information is then pooled from all subsamples to compute the cost for the entire sample. The final tree is shown in figure 8.

The results of the cross-validation are shown in figure 9, here the cost (misclassification error) is displayed against the number of nodes used within the different models developed. The optimum, model used three nodes and was 100% accurate.

5 Related Work

A statistical model based on linear regression, augmented by expert rules, was demonstrated by Baumgartner et al to have highly accurate classification rate in terms of sensitivity (>95.2%) and a very low false positive identification rate (0.001%) [22]. This research did show that regression could potentially be used to identify the disorders very accurately but their method was employed on a database of already quantified metabolites detected using Modern Tandem Mass Spectrometry (MS/MS). Further work by Baumgartner *et al* involved the development of a biomarker identification algorithm which improved the discriminatory powers of their earlier regression model [23].

6 Conclusions

The exploratory analysis conducted has revealed for this particular IEM, that the abnormal sample were extreme cases and thus easy to differentiate from the normal samples. The initial analysis using k-means clustering, had an accuracy of 77% in determing, for a two class problem how the data should be divided. The more sophisticated decision tree model was very accurate at 100%, however, we expect this to drop as more IEM's are introduced into the model. Future work will involve the collection of more data from normal and abnormal sources. The

difficulties, for data collection arise from the rarity of the abnormal cases. The intention is to build into our model more IEM's to enable a methodology for fast, automated screening.

Acknowledgments

We wish to thank the developers of the Matlab SNC Tools package.

References

1. Humphrey-Smith, I., Dracup, W.: The search for validated biomarkers in the face of biosystems complexity. Drug Discovery World, 49–56 (Spring 2005)
2. Kumps, A., Duez, P., Mardens, Y.: Metabolic, nutritional, latrogenic, and artifactual sources of urinary organic acids: a comprehensive table. Clinical Chemistry 48(5), 708–717 (2002)
3. Chu, C., Xiao, X., Zhou, X., Lau, T., Rogers, M., Fok, T., Law, L., Pang, C., Wang, C.: Metabolomic and bioinformatic analyses in asphyxiated neonates. Clinical Biochemistry 39, 203–209 (2006)
4. Tanaka, K., Budd, M., Efron, M., Isselbacher, K.: Isovaleric acidemia: a new genetic defect of leucine metabolism. Proc. Natl. Acad. Sci. USA 56(1), 236–242 (1966)
5. Kimura, M., Yamamoto, T., Yamaguchi, S.: Automated metabolic profiling and interpretation of GC/MS data for organic acidemia screening: a personal computer-based system. Journal of Experimental Medicine 188, 317–334 (1999)
6. Halket, J., Przyborowska, A., Stein, S., Mallard, W., Down, S., Chalmers, R.: Deconvolution gas chromatography/mass spectrometery of urinary organic acids - potential for pattern recognition and automated identification of metabolic disorders. Rapid Communications in Mass Spectrometry 13, 279–284 (1999)
7. Ho, S., Lukacs, Z., Hoffmann, G., Linder, M., Wetter, T.: Feature construction can improve diagnostic criteria for high-dimensional metabolic data in newborn screening for medium-chain acyl-coa dehydrogenase deficiency. Clinical Chemistry 53(7), 1330–1337 (2007)
8. Duran, A., Wang, L., Yng, J., Sumner, L.: Metabolomics spectral formatting, alignment and conversion tools MSFACTS. Bioinformatics 19(17), 2283–2293 (2003)
9. Hanson, M., Andersen, B., Smedsgaard, J.: Automated and unbiased classification of chemical profiles from fungi using high performance liquid chromatograph. Journal of Microbiological Methods 61, 295–304 (2005)
10. Guillo, C., Barlow, D., Perrett, D., Hanna-Brown, M.: Micellar electrokinetic capillary chromatography and data alignment analysis: a new tool in urine profiling. Journal of Chromatography A 1027, 203–212 (2004)
11. Baran, R., Kochi, H., Saito, N., Suematsu, M., Soga, T., Nishioka, T., Robert, M., Tomita, M.: MathDAMP: a package for differential analysis of metabolite profile. BMC Bioinformatics 7, 1–9 (2006)
12. Broadhurst, D., Kell, D.: Statistical strategies for avoiding false discoveries in metabolomics and related experiments. Metabolomics 2(4), 171–196 (2006)
13. Damian, D., Oresic, M., Verheij, E., Meulman, J., Friedman, J., Adourian, A., Morel, N., Smilde, A., Van Der Greef, J.: Applications of a new subspace clustering algorithm (COSA) in medical systems biology. Metabolomics 3(1), 69–77 (2007)

14. Goodacre, R., Vaidyanathan, S., Dunn, W., Harrigan, G., Kell, D.: Metabolomics by numbers: acquiring and understanding global metabolite data. TRENDS in Biotechnology 22(5), 245–252 (2004)
15. Smit, S., Hoefsloot, H., Smilde, A.: Statistical data processing in clinical proteomics. Journal of Chromatography B866(1-2), 77–88 (2008)
16. Obuchowshi, N., Lieber, M., Wians, F.: ROC curves in /it Clinical Chemistry: uses, misuses, and possible solutions. Clinical Chesmitry 50(7), 118–1125 (2004)
17. Leibermeister, W., Klipp, E.: Bringing metabolic networks to life: integration of kinetic, metabolic and proteomic data. Theoretical Biology and Medical Modelling 42(3), 1–15 (2006)
18. Yeang, C., Vingron, M.: A joint model of regulatory and metabolic networks. BMC Bioinformatics 332(7), 1–5 (2006)
19. Hilario, M., Kalousis, A., Prados, J., Binz, P.: Data mining for mass spectra-based cancer diagnosis and biomarker discovery. Drug Discovery Today 2(5), 214–222 (2004)
20. Gower, J., Hand, D.: Biplots. Chapman and Hall, London (1996)
21. Martinez, W., Martinez, A.: Exploratory data analysis with Matlab. Chapman and Hall, New York (2000)
22. Baumgartner, C., Bohm, C., Baumgartner, D.: Modelling of classification rules on metabolic patterns including machine learning and expert knowledge. Journal of Biomedical Informatics 38(2), 89–98 (2005)
23. Baumgartner, C., Baumgartner, D.: Biomarker discovery, disease classification, and similarity query processing on high-throughput MS/MS data of inborn errors of metabolism. Journal of Biomolecular Screening 11(1), 90–99 (2006)

Multi-relational Data Mining for Tetratricopeptide Repeats (TPR)-Like Superfamily Members in *Leishmania spp.*: Acting-by-Connecting Proteins

Karen T. Girão, Fátima C.E. Oliveira, Kaio M. Farias, Italo M.C. Maia, Samara C. Silva, Carla R.F. Gadelha, Laura D.G. Carneiro, Ana C.L. Pacheco, Michel T. Kamimura, Michely C. Diniz, Maria C. Silva, and Diana M. Oliveira

Núcleo Tarcisio Pimenta de Pesquisa Genômica e Bioinformática – NUGEN, Faculdade de Veterinária, Universidade Estadual do Ceara – UECE, Av. Paranjana, 1700 – Campus do Itaperi, Fortaleza, CE 60740-000 Brazil
diana.magalhaes@uece.br

Abstract. The multi-relational data mining (MRDM) approach looks for patterns that involve multiple tables from a relational database made of complex/structured objects whose normalized representation does require multiple tables. We have applied MRDM methods (relational association rule discovery and probabilistic relational models) with hidden Markov models (HMMs) and Viterbi algorithm (VA) to mine tetratricopeptide repeat (TPR), pentatricopeptide (PPR) and half-a-TPR (HAT) in genomes of pathogenic protozoa *Leishmania*. TPR is a protein-protein interaction module and TPR-containing proteins (TPRPs) act as scaffolds for the assembly of different multiprotein complexes. Our aim is to build a great panel of the TPR-like superfamily of *Leishmania*. Distributed relational state representations for complex stochastic processes were applied to identification, clustering and classification of *Leishmania* genes and we were able to detect putative 104 TPRPs, 36 PPRPs and 08 HATPs, comprising the TPR-like superfamily. We have also compared currently available resources (Pfam, SMART, SUPERFAMILY and TPRpred) with our approach (MRDM/HMM/VA).

Keywords: Multi-relational data mining; hidden Markov models, Viterbi algorithm, tetratricopeptide repeat motif, *Leishmania* proteins.

1 Introduction

Early efforts in bioinformatics concentrated on finding the internal structure of individual genome-wide data sets; with the explosion of the 'omics' technologies, comprehensive coverage of the multiple aspects of cellular/organellar physiology is progressing rapidly, generating vast amounts of data on mRNA profiles, protein/metabolic abundances, and protein interactions encompassing a systems-level approach that requires integrating all of the known properties of a given class of components (e.g., protein abundance, localization, physical interactions, etc.) with computational methods able to combine large and heterogeneous sets of data [1]. A

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 359–372, 2008.

technique for generation of unified mechanistic models of cellular/organellar processes (a major challenge for all who seek to discover functions of many yet unknown genes) is the multi-relational data mining (MRDM) approach, which looks for patterns that involve multiple input tables (relations) from a relational database (db) made of complex/structured objects whose normalized representation requires multiple tables [2]. MRDM extends association rule mining to search for interesting patterns among data in multiple tables rather than in one input table [3]. We have applied MRDM methods (relational association rule discovery – RARD and probabilistic relational models - PRMs) combined with hidden Markov models (HMMs) [4-5] and the Viterbi algorithm (VA) [6] to mine the tetratricopeptide repeat (TPR) [7-9] and related motifs (pentatricopeptide repeat (PPR) [10-11] and half-a-TPR (HAT) [12] in pathogenic protozoa *Leishmania spp*.. Our aim is to build a great panel of the TPR-like superfamily of proteins, whose members can be further assigned functional roles in terms of containing motifs. TPR motifs were originally identified in yeast as protein-protein interaction (PPI) modules [7], but now they are known to occur in a wide variety of proteins (over 12,000 as included in SMART nrdb) present in prokaryotic and eukaryotic organisms [8], being involved in protein-protein and protein-lipid interactions in cell cycle regulation, chaperone function and post-translation modifications [7-9]. TPRs exhibit a large degree of sequence diversity and structural conservation (two antiparallel alpha-helices separated by a turn) that might act as scaffolds for the assembly of different multiprotein complexes [13] including the peroxisomal import receptor and the NADPH oxidase [14]. Similar to TPR, PPR and HAT motifs also have repetitive patterns characterized by tandem array of repeats, where the number of motifs seems to influence the affinity and specificity of the repeat-containing protein for RNA [12,15-16]. PPR-containing proteins (PPRPs) occur predominantly in eukaryotes [10] (particularly abundant in plants), while it has been suggested that each of the highly variable PPRPs is a gene-specific regulator of plant organellar RNA metabolism. HAT repeats are less abundant and HAT-containing proteins (HATPs) appear to be components of macromolecular complexes that are required for RNA processing [10-12,15-16].

TPR-containing proteins (TPRPs) have recently attracted interest because of their versatility as scaffolds for the engineering of PPIs [17-18] and, since they are characterized by homologous, repeating structural units, which stack together to form an open-ended superhelical structure, such an arrangement is in contrast to the structure of most proteins, which fold into a compact shape [19]. The curvature created by the superhelical nature predetermines the target proteins that can bind to them [20]. TPRs, PPRs and HAT (all together referred as TPR-like motifs), form a large superfamily or the clan TPR-like [7-16]. Homologous structural repeat units are often highly divergent at the sequence level, a feature that makes their prediction challenging. Currently, several web-based resources are available for the detection of TPRs, including Pfam [21], SMART [22], and SUPERFAMILY [23], which use HMM profiles constructed from the repeats trusted to belong to the family (from closely homologous repeats); therefore, divergent repeat units often get a negative score and are not considered in computing the overall statistical significance, even though they are individually significant [18]. For this reason Pfam, SMART, and SUPERFAMILY perform with limited accuracy in detecting remote homologs of known TPRPs and in delineating the individual repeats within a protein [18]. A new

profile-based method [18], TPRpred, uses a P-value- dependent score offset to include divergent repeat units and to exploit the tendency of repeats to occur in tandem. Although TPRpred indeed performs significantly better in detecting divergent repeats in TPRPs, and finds more individual repeats than the afore mentioned methods, we have noticed that it still fails to detect some particular groups of members of TPR-like superfamily, such as now we demonstrate for *Leishmania spp*. Since the characterization of proteins of a given family often relies on the detection of regions of their sequences shared by all family members, while computing the consensus of such regions provides a motif that is used to recognize new members of the family, our approach of HMMs/VA with MRDM was suitable to detect 104 TPRPs, 36 PPRPs and 08 HATPs in *Leishmania spp*. genomes, a greater number than Pfam, SMART, SUPERFAMILY are able to yield (Table 1) and slightly higher than TPRpred.

2 Methods

2.1 Data Sources and Bioinformatics Tools

We have used publicly available datasets of individual or clusters of gene/protein data on *Leishmania spp*., mainly *L. major, L. braziliensis*, *L. infantum* and related trypanosomatids (GeneDB [24] and NCBI/Entrez - www.ncbi.nlm.nih.gov/sites/gquery). Variants of BLAST [25] and GlimmerHMM [26] were widely used for sequence similarity searches, comparisons and gene predictions. External db searches were performed against numerous collections of protein motifs and families. Gene ontology (GO) terms were assigned, based on top matches to proteins with GO annotations from Swiss-Prot/trEMBL (www.expasy.org/sprot) and AMIGO after GeneDB (www.genedb.org/amigo/perl) access. Functional assignment of genes/gene products was inferred using the RPS-BLAST search against conserved domain db (CDD) [27]. For protein domain identification and analysis of protein domain architectures, Simple Modular Architecture Research Tool (SMART) [22], Pfam [21], SUPERFAMILY [23] and TPRpred [18] were used. For multiple alignments we used MUSCLE [28].

2.2 Finding a TPR-Like Regular Expression

TPR motif sequence is loosely based around the consensus residues -W-LG-Y-A-F-A-P-. TPRs are minimally conserved (degenerate and variable) regions of 34-residue long extension (with exceptions accepted to the range of 31 residues [14]). Three-dimensional structural data have shown that tandem arrays of 3-16 TPR motifs generate a right-handed helical structure with an amphipathic channel that might accommodate the complementary region of a target protein [7, 9, 14]. The PPR motif is a degenerate 35-residue sequence, closely related to the 34-residue TPR motif. On the basis of the solved structure of a TPR domain [9] as well as modeling approaches [10], each PPR domain is though to be configured also as two distinct antiparallel alpha-helices, helices A and B. In PRPPs, 2-26 tandem repeats of these alpha-helical pairs are predicted to form a superhelix that encloses a central spiral groove with a positively charged ligand-binding surface [10]. Although there exists no position characterized by an invariant residue, a consensus sequence pattern of small and large

hydrophobic residues has been defined: small hydrophobic residues are commonly observed at positions 8, 20, and 27, while large ones are at 4, 17, and 24 [14]. The consensus sequence for TPR-like motif is given below (1) and it has been used as a regular expression, which defines the most probable amino acid (aa) at each position within this core, to fully exploit the TPR motif finding in *Leishmania spp*. genomes. As reported in [29], we systematically solved inconsistencies in the motif annotation by manual expertise. Since motif occurrences are adjacent in sequences, we could define the motif sequence of a protein as the succession of motifs read from the N toward the C terminus.

```
[WLF]-X(2)-[LIM]-[GAS]-X(2)-[YLF]-X(8)-[ASE]-X(3)-[FYL]-X(2)-[ASL]-X(4)-[PKE]
```

(1)

2.3 Definition of a TPR-Like Protein

We have defined a TPR-like protein as any protein sequence containing a TPR-like motif that fits in our regular expression (1), which also, by reference, confirms to a set of known bona fide domains contained in TPR-like superfamily [a.118.8] of SCOP (v.1.69) [18,30], SMART (v.5.0) [22], TPRpred [18] and SUPERFAMILY [23]. Classification criteria are supported by structural/sequence similarity, plus searches with remote homology prediction.

2.4 Profile Generation After Querying TPR-Like Motifs

Aware that performance dependence on any sequence profiles relies on either the selectivity or sensitivity of its regime, respectively depending on the number of close or remote homologs used [18], we have established a fixed threshold value to include a minimum number of remote homologs (to avoid having too many false positives). Initial profiles were generated by iterative searches against non-redundant dbs (nrdbs) at NCBI and GeneDB, filtered to a maximum pairwise sequence identity of 60% (nr-60) by CD-HIT [31-32], slightly modified after [18] in a sense that we have extracted sequences conservatively with PSI-BLAST through multiple iterations using the TPR-like regular expression (1) as a query sequence. We, then, performed iterative searches to convergence on nr-60 minus TPRPs (detected by Pfam, SMART, SUPERFAMILY and TPRPred) with various threshold parameters to test the resulting profiles on a positive (TPR-like) or negative set (non TPR-like). Best profiles were selected based on its performance on a predicted family assignment, as illustrated on Figure 1a.

2.5 TPR-Like Superfamily Assignment

To provide structural (and hence implied functional) assignments to TPR-like proteins at the superfamily level, structured sequences from available *Leishmania* genomes were randomly selected and parsed into unique 24,708 sequences. Each sequence was a labeled input to a multi-class motif classifier. To pick the best method to represent one or more of the three target motifs, we compared the results of motif classifiers when the sequence was presented as a (I) TPR-containing, (II) PPR-containing (III) HAT-containing, (IV) combination of any two or three motifs, and (V) not-containing

target motifs. Performance was measured by classification precision, recall and F1 measure (a composite measure of classification precision and recall).

2.6 Hidden Markov Models (HMMs) and the Viterbi Algorithm (VA)

A HMM is a probabilistic network of nodes, so called states. One state q_i is connected to another state q_j by a transition probability ij. Non-silent states are able to emit an alphabet of symbols [4-5]. A special topology of HMMs, termed pHMM, is frequently used in homology detection of protein families [33]. Transition and emission probabilities are estimated by a maximum likelihood approach combined with a standard dynamic programming algorithm for decoding HMMs, the Viterbi (VA) [6]) to get site and path dependent probabilities for every hidden state in the posterior decoding. In a first validation step we used the feature of trained HMMs to emit domain-specific sequences according to their model parameters. Sequences were compared with generated state paths in the same way as described earlier [29]. The process of generation was repeated 10 times for every TPR-like motif. To fully exploit the sequential ordering of motifs in a set, we used pHMMs to label motif types. We have transformed the motif categorization problem into a HMM sequence alignment problem. The HMM states correspond to the motif types. Labeling motifs in a sequence is equivalent to aligning the sequences to HMM states. There are five states in our HMM model: (I) TPR-containing, (II) PPR-containing (II) HAT-containing, (IV) combination of any two or all motifs, and (V) not-containing target motifs. Transition probabilities between these states were estimated from the training data by dividing the number of times each transition occurs in the training set by the sum of all the transitions. The state emission probabilities were calculated from the score output reported by the multi-class classifiers. Given the HMM model [33], state emission probabilities and state transition probabilities, VA was used to compute most-likely sequence of states that emit (any of the target) motifs in sequences. Subsequently, the state associated with the motif was extracted from the most-likely sequence of states [34].

2.7 Multi-relational Data Mining (MRDM) Method

Algorithms for RARD are well suited for exploratory data mining due to the flexibility required to experiment with examples more complex than feature vectors and patterns more complex than item sets [35], such as the case with TPR-like motifs. An adequate approach of machine learning [36] focuses on learning a complex web of relationships among a collection of diverse objects rather than supervised learning from independent and identically distributed training examples (a classifier f that given an object x would produce as output a classification label $y = f(x)$). Such formalism, developed as PRMs [36-37], can represent these webs of relationships and support learning and reasoning with them [38]. PRMs are a multi-relational form of Bayesian networks that allow descriptions of a template for a probability distribution. This, together with a set of motif objects, defines a distribution over the attributes of the objects. Such a model can then be used for reasoning about an entity using the entire rich structure of knowledge encoded by the relational representation [37,39]. For each PRM, we were interested in constructing a model whose trades off fit to data

with the TPR-like motif model complexity. This tradeoff allows us to avoid fitting the training data too closely, which would reduce our ability to predict unseen data.

3 Results and Discussion

3.1 TPR-Like Motif Localization Task

As illustrated on Fig. 1, we have applied two MRDM methods (RARD and PRMs) after HMMs/VA to mine TPR, PPR and HAT repeats in protein sequences of *Leishmania spp*. Provided six variants of the data set for the TPR-like motif localization task (considering four TPRs, one PPR and one HAT in the TPR-like clan), the first version consisted of a single table with 24,708 attributes and the second consisted of two tables with 26 attributes in total. We used a normalized version of the data set with two tables. The names of the two original tables are *motifs_relation* and *interactions_relation*. The *motifs_relation* table contained 120 different motifs but there could be more than one row in the table for each motif. The attribute *motif_id* identifies a motif is uniquely. Since our current implementation of MRDM requires that the target table must have a primary key, it was necessary to normalize the *motifs_relation* table before we could use it as the target table. This normalization was achieved by creating the tables named *motif*, *interaction*, and *composition* as follows: Attributes in the *motifs_relation* table that did not have unique values for each motif were placed in *composition table* and the rest of attributes were placed in *motif* table. The *motif_id* attribute is a primary key in the *motif* table and as a foreign key in *composition* table. The *interaction* table is identical to the original *interactions_relation* table. This represents one of several ways of normalizing the original table and renormalization of the relational db has an impact on the entity-relation diagram for the renormalized version of *Motif Localization* db. Thus, for

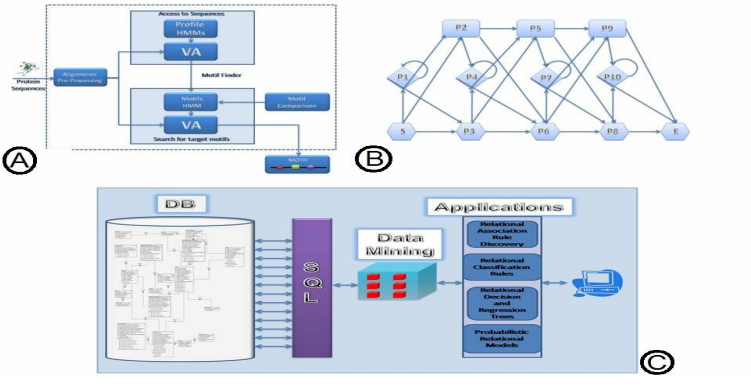


Fig. 1. Overall schema for multi-relational data mining (MRDM) approach (C) comprising the search for TPR, PPR and HAT motifs in available *Leishmania* genomes after (A) hidden Markov models (HMMs) powered by (B) the Viterbi algorithm (VA), a combined method for superfamily assignment on searches among *Leishmania* fully sequenced species and trypanosomes [24]. The input to VA is a HMM in sequences of length *L*. The output is the highest probability path through the HMM that could generate the input sequence.

TPR-like motif localization task, the target table is *motif* and the target attribute is *localization*. From this point of view, the training set consists of 120 motifs and the test set 68. The experiments described here focused on building a classifier for predicting the localization of motif-containing proteins by assigning the corresponding instance to one of six possible localizations. For this motif localization task, we have chosen to construct a classifier using all training data and test the resulting classifier on the test set provided by *Leishmania* sequences. This task presents significant challenges because many attribute values in training instances corresponding to the 120 training motifs are missing. Initial experiments using a special value to encode a missing value for an attribute resulted in classifiers whose accuracy is around 40% on the test data. This prompted us to investigate incorporation of other approaches to handling missing values. Replacing missing values by the most common value of the attribute for the class during training resulted in an accuracy of around 68%. This shows that providing reasonable guesses for missing values can significantly enhance the performance of MRDM on our data sets. However, in practice, since class labels for test data are unknown, it is not possible to replace a missing attribute value by the most frequent value for the class during testing. Hence, there is a need for better ways of handling missing values (e.g., predicting missing values based on values of which attributes?).

3.2 Identification of the TPR-Like Superfamily in *Leishmania* spp. Genomes

The percentage of repeat-containing proteins, such as TPR-like, grows with the complexity of the organism, with repeat proteins being particularly abundant in multicellular organisms [40]. Genomes of unicellular eukaryotes, as *Leishmania*, usually possess a relatively high number of putative encoding genes (around 8,000 genes in *L. major*, e.g.) [24]. Analyses of such a large number of coded proteins require that the characterization of a given family of proteins be dependent on detection of regions of their sequences shared by all family members. Computing the consensus of such regions provides a motif that is used to recognize new members of the family [41]. With the sequencing completion of 03 *Leishmania* and several trypanosomes genomes [24], we were able to search for all TPR-like genes in *Leishmania* using the defining characteristic of a TPR-like protein. As depicted by Tab. 1, numbers of members detected through different tools (GeneDB, Pfam and

Table 1. Comparative results of TPR-like motif finding in *Leishmania* genes obtained with three standard tools (Superfamily, GeneDB and Pfam) and with our method (MRDM/ HMM/VA). Numbers are shown in terms of TPR, PPR and HAT-containing proteins in three species (*L. major*, *L. infantum* and *L. braziliensis*).

	SUPERFAMILY			GENEDB			PFAM			MRDM		
	TPR	PPR	HAT	TPR	PPR	HAT	TPR	PPR	HAT	TPR	PPR	HAT
L. major	95	-	-	62	12	3	51	12	1	104	34	3
L. infantum	98	-	-	37	11	2	42	11	-	104	36	3
L. braziliensis	99	-	-	53	8	1	55	8	1	103	33	2

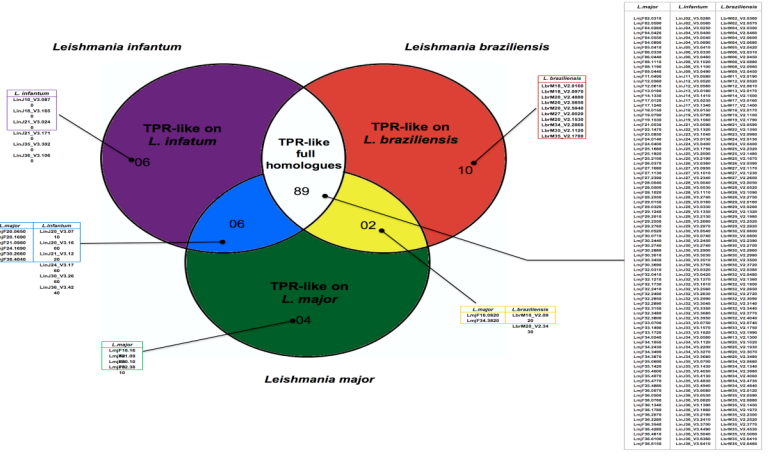


Fig. 2. Schematic diagram of TPR-containing genes in genomes of *Leishmania major, L. infantum* and *L. braziliensis*, detected after a multi-relational data mining and hidden Markov model/Viterbi algorithm approach (MRDM/HMM/VA). Shared ortologues among the three species are illustrated as colored circles intersections and individual identifiers (GeneDB IDs) for putative TPRPs are given as lateral tables.

Superfamily) are shown in comparison to our method (MRDM/HMM/VA), which is able to assign a significantly larger number of TPR-like motif-containing proteins in *Leishmania*: 104 TPRs and 36 PPRs at the most and 08 HATs in total. These members are elements putatively involved in several key cellular processes, such as glycosome biogenesis (PEX5 and PEX14) and flagellar pathways (IFT subunits, cyclophilins, phosphatases), besides binding partners of either motor or cargo proteins (kidins220/ARMS and other members of the KAP family of P-loop NTPases) or those involved with assembly/disassembly of protein complexes. The resulting descriptions of the families and its members, a good example of relevant patterns found along with reasonable assignment of family members with our approach, should provide a solid and unified platform on which future genetic and functional studies regarding *Leishmania* TPRPs can be based.

3.3 TPR-Encoding Genes in *Leishmania spp*. Genomes

We first used the alignment of 275 sequences previously identified as putative strict TPR-containing motifs (obtained from Superfamily, SMART, Pfam and TPRpred) to obtain the consensus model (Fig. 3). This TPR signature matrix was subsequently used to search for TPR motifs in the six reading frames of whole *Leishmania* genomes. Multiple alignment of *Leishmania* TPRP sequences revealed that most of substitutions in the TPRs occur at nonconsensus positions; consensus residues are selectively conserved between ortologues (particularly in *Trypanosoma spp*). Because TPR motifs are highly degenerate, a fairly large number of false positive hits were expected. However, because TPR motifs appear usually as tandem repeats, we could remove most random uninteresting matches by omitting all orphan TPR motifs that

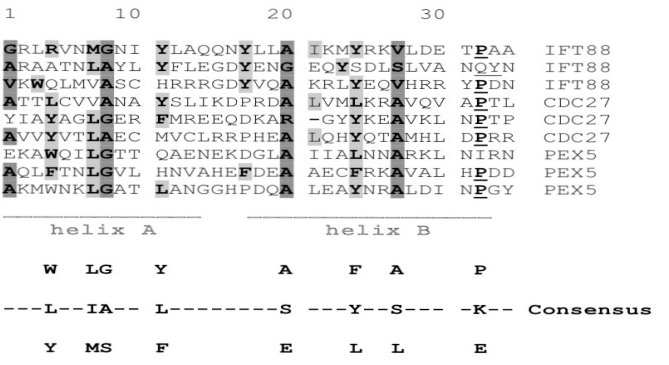


Fig. 3. Multiple sequence alignment of typical TPR motifs present in IFT88, CDC27 and PEX5 proteins of *Leishmania* (LmjF27.1130, LmjF05.0410 and LmjF35.1420). TPR motif residues are shown with *dark* and *light gray shaded boxes* for small and large hydrophobic residues, respectively. Small hydrophobic residues are commonly observed at positions 1, 8, 20 and 27. Position 32 is frequently proline (*bold underlined*), located at the C terminus of helix B, and large hydrophobic residues are also located at particular positions, especially 4, 17, and 24. Schematic consensus for TPR is illustrated.

were found farther than 200 nucleotides from any other TPR motif. The 465 TPR motifs retained formed 132 clusters, each of which comprised a putative TPR gene. Each TPR motif cluster was then investigated in detail by manually analyzing the positions and reading frames of the TPR motifs compared with available *Leishmania* genomes (1) open reading frame (ORF) models and (2) predicted protein sequences within potential coding sequences. From this analysis, 104 putative TPR ORF models were constructed (i.e., 28 motif clusters were discarded or fused with other clusters). TPR genes are fairly evenly distributed throughout the 36 mini-chromosomes of *L. major*, with little in the way of obvious clusters. The densest grouping of TPR genes lies on chromosomes 30, 32 and 36, the latter which contains 13 genes, the maximum number found in any isolate chromosome of *L. major*.

3.4 Functional Features of TPR-Like Motifs

The preliminary functional predictions of a range of family members performed here, together with the sparse data on these proteins in *Leishmania* published so far, allows us to propose putative models in which TPR-like proteins might play the role of sequence-specific adaptors for a variety of other RNA-associated proteins. Such models, yet requiring further testable hypothesis, can surround a testable prediction: that TPR-like proteins in *Leishmania* might be directly or indirectly associated with specific RNA sequences and with defined effector proteins, as previously suggested in *Arabidopsis* [15]. Future work needs to be directed toward the identification of these factors to elucidate the precise functions of one of the largest and least understood protein families in *Leishmania*, the TPR-like. For now, our MRDM approach may be also relevant for other families of proteins with repeated motifs, in a similar way to what was reported by [42]. We must recall that the *L. major* genome contains 708 predicted proteins annotated with the term *repeat* in their descriptions [24], including

only 62 out of the 104 TPRPs and 12 out of the 36 PPRPs that we have identified here. For instance, there are 18 proteins containing repeats of the Kelch motif often associated to a F-box domain, 169 WD40 repeat-containing proteins and 121 proteins with Leu-rich repeats frequently associated to a protein kinase domain. Others cases are armadillo (61) and ankyrin (45) repeats-containing proteins. In some of these protein families, the region containing the repeats is a large part of the proteins that can, and should be, a valuable target for applying MRDM methods.

3.5 PPR- and HAT-Encoding Genes in *Leishmania spp*. Genomes

The name PPR was coined based on its similarity to the better-known TPR motif [10]. PPRPs make up a significant proportion of the unknown function proteins in many organisms, but only few of them have functional roles ascribed, although a putative RNA-binding function is widely accepted [15] and one PPRP is involved in RNA editing [11]. The existence of a large family of PRPPs only became apparent with the *Arabidopsis* Genome Initiative that revealed 446 PPR coding genes – 6% of its entire genome [15]. The PPRP family has been divided, on the basis of their motif content and organization, into two subfamilies: the PPRP-P and the exclusive plant combinatorial and modular proteins (PCMPs). PPR motifs have been found in all eukaryotes analyzed to date, but with an extraordinary discrepancy in numbers between plant and nonplant organisms (the human genome encodes only six putative PPRPs). Trypanosomatids and other flagellated organisms are expected to have an intermediate number (around one hundred PPR genes), a number still far from the 36 PPR-encoding genes we have found here (12 of them annotated at GeneDB as conserved hypothetical proteins of *L. major*). Recent reports [43-44] mention more than twenty (respectively 23 and 28) PPRPs identified in *Trypanosoma brucei* (with at least 25 ortologues found in *L. major*) and with a predicted indication that most of these proteins are targeted to mitochondria. As of Release 2.1 of GeneDB [24] with curated annotations of *Leishmania* genes, 13 of the GeneDB ORFs are annotated as conserved hypothetical proteins that contain PPR motifs based on matches with the PFAM profile PF01535 or SMART profile IPR002885. None of *Leishmania* GeneDB models are annotated as homologs of known PPRPs. Of the two sets of ORF models (ours and GeneDB's), 12 are identical (i.e., our analysis agreed with the GeneDB model). The 13th GeneDB model does not have an equivalent in our set because we did not consider it to be a PPRP by our criteria (lacking tandem motifs matching our HMM profiles) Twenty-one of our models have no GeneDB equivalent and correspond to genes apparently overlooked during annotation or considered to be pseudogenes. In all, 22 of our 34 models differ in at least some respects from the corresponding GeneDB model, but correspond quite well to the 28 PPRPs identified in *T brucei* [44], what reinforces how well conserved PPR genes seem to be in trypanosomatids. It should be noted that in very few of these cases are molecular data available that can be used to decide between discordant models. Our choice has been generally made by comparison with other genes in the family and a general familiarity with these proteins. A noticeable characteristic of PPR genes is that they rarely contain introns within coding sequences even in higher eukaryotes (more than 80% of known PPR genes of plants unexpectedly do not contain introns), what is also true for *Leishmania* ORF models (an obvious extension for trypanosomatid genes that usually

do not contain introns anyway). This characteristic might explain why PPR genes are relatively short (on average <2 kb) despite the fact that PPRPs are comparatively large proteins (680 aa on average).

HAT repeats have three aromatic residues with a conserved spacing, being structurally and sequentially similar to TPRs/PPRs, although they lack the highly conserved alanine and glycine residues found in TPRs. The number of HAT repeats found in different proteins varies between 9 to 12. HATPs appear to be components of macromolecular complexes that are required for RNA processing and the HAT motif has striking structural similarities to HEAT repeats (IPR000357), being of a similar length and consisting of two short helices connected by a loop domain, as in HEAT repeats [10-12, 15-16]. Our survey identified a total of 08 putative HATPs (Table 1) in the three species of *Leishmania* analyzed, but the lack of general information on HATPs does not allow any further indication on their definite significance on the protozoan genome. The detection of such a small, but significant, presence of HATPs in *Leishmania* is certainly an issue for future investigation.

4 Conclusions

We have performed bioinformatics analyses of *Leishmania* TPR, PPR and HAT proteins with an integrated MRDM/HMM/VA approach that, in contrast to other currently available resources (PFAM, SMART, SUPERFAMILY, TPRpred), seeks to capture as much model information as possible in the pattern matching heuristic, without resorting to more standard motif discovery methods. TPR genes are ubiquitous, whereas PPRs and HATs are mostly found in eukaryotes, but, in common, they have the fact of being largely unexplored in *Leishmania* parasites. Diffusion of new developments and applications of MRDM techniques to data-driven knowledge discovery problems in bioinformatics is a future direction towards better power of biological inference after sequence and structural analyses.

References

1. Ideker, T., Bafna, V., Lemberger, T.: Integrating scientific cultures. Mol. Syst. Biol. 3, 105–112 (2007), doi:10.1038/msb4100145
2. Getoor, L.: Multi-relational data mining using probabilistic relational models: research summary. In: Knobbe, A.J., van der Wallen, D.M.G. (eds.) Proceedings 1st Workshop in Multi-relational Data Mining, KDD (2001)
3. Dehaspe, L., De Raedt, L.: Mining association rules in multiple relations. In: Džeroski, S., Lavrač, N. (eds.) ILP 1997. LNCS, vol. 1297. Springer, Heidelberg (1997)
4. Eddy, S.R.: Profile hidden Markov models. Bioinformatics 14, 755–763 (1998)
5. Winters-Hilt, S.: Hidden Markov Model Variants and their Application. BMC Bioinformatics 7, 14 (2006)
6. Forney Jr., G.D.: The Viterbi algorithm. Proc. IEEE 61, 268 (1973)
7. Blatch, G.L., Lässle, M.: The tetratricopeptide repeat: a structural motif mediating protein-protein interactions. Bio. Essays 21, 932–939 (1999)
8. D'Andrea, L.D., Regan, L.: TPR proteins: the versatile helix. Trends Biochem. Sci. 28, 655–662 (2003)

9. Das, A.K., Cohen, P.W., Barford, D.: The structure of the tetratricopeptide repeats of protein phosphatase 5: implications for TPR-mediated protein-protein interactions. EMBO. J. 17, 1192–1199 (1998)
10. Small, I.D., Peeters, N.: The PPR motif – a TPR-related motif prevalent in plant organellar proteins. Trends Biochem. Sci. 25, 46–47 (2000)
11. Kotera, E., Tasaka, M., Shikanai, T.: A pentatricopeptide repeat protein is essential for RNA editing in chloroplasts. Nature 433, 326–330 (2005)
12. Preker, P.J., Keller, W.: The HAT helix, a repetitive motif implicated in RNA processing. Trends Biochem. Sci. 23, 15–16 (1998)
13. Scheufler, C., Brinker, A., Bourenkov, G., et al.: Structure of TPR domain-peptide complexes: critical elements in the assembly of the Hsp70-Hsp90 multichaperone machine. Cell 101, 199–210 (2000)
14. Koga, H., Terasawa, H., Nunoi, H., et al.: Tetratricopeptide Repeat (TPR) Motifs of p67phox Participate in Interaction with the Small GTPase Rac and Activation of the Phagocyte NADPH Oxidase. Biol. Chem. 274, 25051–25060 (1999)
15. Lurin, C., Andrés, C., Aubourg, S., et al.: Genome-Wide Analysis of Arabidopsis Pentatricopeptide Repeat Proteins Reveals Their Essential Role in Organelle Biogenesis. The Plant Cell 16, 2089–2103 (2004)
16. Rivals, E., Bruyère, C., Toffano-Nioche, C., Lecharny, A.: Formation of the Arabidopsis Pentatricopeptide Repeat Family. Plant Physiol. 141, 825–839 (2006)
17. Main, E.R.G., Lowe, A.R., Mochrie, S.G.J., Jackson, S.E., Regan, L.: A recurring theme in protein engineering: the design, stability and folding of repeat proteins. Curr. Opin. Struct. Biol. 15, 464–471 (2005)
18. Karpenahalli, M.R., Lupas, A.N., Söding, J.: TPRpred: a tool for prediction of TPR-, PPR- and SEL1-like repeats from protein sequences. BMC Bioinformatics 8, 2 (2007), doi:10.1186/1471-2105-8-2
19. Groves, M.R., Barford, D.: Topological characteristics of helical repeat proteins. Curr. Opin. Struct. Biol. 9, 383–389 (1999)
20. Kobe, B., Kajava, A.V.: When protein folding is simplified to protein coiling: the continuum of solenoid protein structures. Trends Biochem. Sci. 25, 509–515 (2000)
21. Sonnhammer, E.L., Eddy, S.R., Durbin, R.: Pfam: a comprehensive database of protein domain families based on seed alignments. Proteins 28, 405–420 (1997)
22. Schultz, J., Milpetz, F., Bork, P., Ponting, C.P.: SMART, a simple modular architecture research tool: identification of signaling domains. Proc. Natl. Acad. Sci. USA 95, 5857–5864 (1998)
23. Madera, M., Vogel, C., Kummerfeld, S.K., Chothia, C., Gough, J.: The superfamily database in 2004: additions and improvements. Nucleic Acids Res. 32, 235–239 (2004)
24. Hertz-Fowler, C., Peacock, C.S., Wood, C., et al.: GeneDB: a resource for prokaryotic and eukaryotic organisms. Nucleic Acids Res. 32, D339–D343 (2004) The Pathogen Sequencing Unit - Wellcome Trust Sanger Institute – GeneDB – (2004), `http://www.genedb.org`
25. Altschul, S.F., Madden, T.L., Schäffer, A.A., et al.: Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic Acids Res. 25, 3389–3402 (1997)
26. Majoros, W.H., Pertea, M., Salzberg, S.L.: TigrScan and GlimmerHMM: two open source ab initio eukaryotic gene-finders. Bioinformatics 20, 2878–2879 (2004)
27. Marchler-Bauer, A., Anderson, J.B., Derbyshire, M.K., et al.: CDD: a conserved domain database for interactive domain family analysis. Nucleic Acids Res. 35, D237–240 (2007)

28. Edgar, R.C.: MUSCLE: multiple sequence alignment with high accuracy and high throughput. Nucleic Acids Res. 32, 1792–1797 (2004)
29. Pacheco, A.C.L., Araujo, F.F., Kamimura, M.T., et al.: Following the Viterbi Path to Deduce Flagellar Actin-Interacting Proteins of Leishmania spp.: Report on Cofilins and Twinfilins. In: Pham, T. (ed.) AIP Proceedings of Computer Models for Life Sciences, CMLS 2007, vol. 952, pp. 315–324. American Institute of Physics, Australia (2007)
30. Murzin, A.G., Brenner, S.E., Hubbard, T., Chothia, C.: SCOP: a structural classification of proteins database for the investigation of sequences and structures. J. Mol. Biol. 247, 536–540 (1995)
31. Li, W., Jaroszewski, L., Godzik, A.: Clustering of highly homologous sequences to reduce the size of large protein databases. Bioinformatics 17, 282–283 (2001)
32. Li, W., Jaroszewski, L., Godzik, A.: Tolerating some redundancy significantly speeds up clustering of large protein databases. Bioinformatics 18, 77–82 (2002)
33. Karplus, K., Barrett, C., Hughey, R.: Hidden Markov models for detecting remote protein homologies. Bioinformatics 14, 846–856 (1998)
34. Friedrich, T., Pils, B., Dandekar, T., Schultz, J., Müller, T.: Modelling interaction sites in protein domains with interaction profile hidden Markov models. Bioinformatics 22, 2851–2857 (2006), doi:10.1093/bioinformatics/btl486
35. Friedman, N., Getoor, L., Koller, D., Pfeffer, A.: Learning probabilistic relational models. In: Proceedings of the International Joint Conference on Artificial Intelligence, pp. 1300–1307. Morgan Kaufman, Stockholm (1999)
36. Getoor, L., Friedman, N., Koller, D., Pfeffer, A.: Learning probabilistic relational models. In: Dzeroski, S., Lavrac, N. (eds.) Relational Data Mining, pp. 307–335. Kluwer, Dordrecht (2001)
37. Getoor, L., Taskar, B., Koller, D.: Using probabilistic models for selectivity estimation. In: Proceedings of ACM SIGMOD International Conference on Management of Data, pp. 461–472. ACM Press, New York (2001)
38. Craven, M., Page, D., Shavlik, J., Bockhorst, J., Glasner, J.: A probabilistic learning approach to whole-genome operon prediction. In: Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology, pp. 116–127. AAAI Press, La Jolla (2000)
39. Segal, E., Taskar, B., Gasch, A., Friedman, N., Koller, D.: Rich probabilistic models for gene expression. Bioinformatics 1, 1–10 (2001)
40. Bjorklund, A.K., et al.: Expansion of protein domain repeats. PLoS Comput. Biol. 2, 114 (2006)
41. Servant, F., Bru, C., Carrère, S., et al.: ProDom: automated clustering of homologous domains. Brief. Bioinform. 3, 246–251 (2002)
42. Rivals, E., Bruyere, E., Toffano-Nioche, C., Lecharny, A.: Formation of the Arabidopsis pentatricopeptide repeat family. Plant Physiol. 141, 825–839 (2006)
43. Mingler, M.K., Hingst, A.M., Clement, S.L., et al.: Identification of pentatricopeptide repeat proteins in Trypanosoma brucei. Mol. Biochem. Parasitol. 150, 37–45 (2006)
44. Pusnik, M., Small, I., Read, L.K., Fabbro, T., Schneider, A.: Pentatricopeptide Repeat Proteins in Trypanosoma brucei Function in Mitochondrial Ribosomes. Mol. Cell. Biol. 27, 6876–6888 (2007)
45. NCBI (National Center for Biotechnology Information / Entrez / Cn3D (All Databases), `http://www.ncbi.nlm.nih.gov/sites/gquery`
46. Swiss-Prot/trEMBL, `http://www.expasy.org/sprot`
47. AMIGO after GeneDB access, `http://www.genedb.org/amigo/perl`

48. SMART, http://smart.embl.de
49. Superfamily, http://supfam.cs.bris.ac.uk
50. TPRpred, http://toolkit.tuebingen.mpg.de/tprpred
51. Arabidopsis Genome Initiative (AGI, 2000), http://www.arabidopsis.org/portals
52. Pfam, http://pfam.wustl.edu/hmmsearch.shtm

Heuristic Non Parametric Collateral Missing Value Imputation: A Step Towards Robust Post-genomic Knowledge Discovery

Muhammad Shoaib B. Sehgal[1,*], Iqbal Gondal[2], Laurence S. Dooley[3], and Ross Coppel[4,5]

[1] ARC Centre of Excellence in Bioinformatics at IMB, University of Queensland, St Lucia, QLD 4067, Australia
Shoaib.Sehgal@gmail.com
[2] Faculty of Information Technology, Monash University, Churchill, VIC. 3842, Australia
[3] Department of Communications and Systems, The Open University, Milton Keynes, MK7 6AA, United Kingdom
[4] Department of Microbiology
[5] Victorian Bioinformatics Consortium, Clayton, VIC. 3800, Australia

Abstract. Microarrays are able to measure the patterns of expression of thousands of genes in a genome to give profiles that facilitate much faster analysis of biological processes for diagnosis, prognosis and tailored drug discovery. Microarrays, however, commonly have missing values which can result in erroneous downstream analysis. To impute these missing values, various algorithms have been proposed including Collateral Missing Value Estimation (CMVE), Bayesian Principal Component Analysis (BPCA), Least Square Impute (LSImpute), Local Least Square Impute (LLSImpute) and K-Nearest Neighbour (KNN). Most of these imputation algorithms exploit either the global or local correlation structure of the data, which normally leads to larger estimation errors. This paper presents an enhanced Heuristic Non Parametric Collateral Missing Value Imputation (HCMVI) algorithm which uses CMVE as its core estimator and Heuristic Non Parametric strategy to compute optimal number of estimator genes to exploit optimally both local and global correlations.

1 Introduction

Microarrays are used to measure expression levels of a myriad of genes under a variety of conditions and the resulting expression profiles have been utilized in a wide range of biological applications from diagnosis to drug discovery [1]. Depending on the application, this expression data may be analyzed by statistical, mathematical and machine learning algorithms [2-4] such as data dimension reduction, class prediction [5] and clustering [6]. Despite its pervasive usage, microarray data frequently contains at least 5% erroneous spots and in most datasets, at least 60% of genes have either one

* Corresponding author.

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 373–387, 2008.

or more erroneous values [7]. These spots are identified as missing values for a variety of reasons, including slide scratches, spotting problems, blemishes on the chip, hybridization error, image corruption or simply dust on the slide [8]. Sometimes for instance, a background colour has a higher intensity than a foreground colour due to hybridization failure or bleeding from neighboring spots, while background subtraction may also produce negative values which are subsequently marked as missing. These missing values can seriously impact upon subsequent data analysis methods such as significant gene selection and clustering algorithms [9, 10].

Several approaches to solving the missing data problem have been proposed, with the simplest being either the repetition of the experiment, though this is often not feasible for economic reasons or ignoring samples containing missing values, but again this is not recommended due to limited number of samples. Other alternatives include, row average/median impute (replacement by the corresponding row average/median) and zero impute (replacing the missing values by zero) though both these approaches are high variance approaches as neither takes advantage of inherent data correlations, so leading to higher estimation errors [11]. It has been well accepted that a better strategy is to attempt to accurately estimate the missing values by exploiting the underlying correlation structure of the data [10, 12]. This has been the catalyst for a number of imputation techniques including *Collateral Missing Value Imputation* (CMVE) [13], *K-Nearest Neighbor* (KNN), *Least Square Imputation* (LSImpute) [12], *Local LSImpute* (LLSImpute) [10] and *Bayesian PCA* (BPCA) [8]. The resulting estimation errors can still be high however, as some algorithms focus mainly on global data correlation (BPCA), while others exploit local correlations in the data (KNN) by using a fixed number of predictor genes. This provided the motivation for the development of new generic techniques that minimise prediction errors by optimising the number of predictor genes. Moreover, the comparative imputation performances of CMVE, BPCA, LSImpute, LLSImpute and KNN has traditionally been numerically evaluated using the *Normalized Root Mean Square Error* (NRMSE) measure, which is partial indicative of the estimation impact on any subsequent biological analysis.

This paper presents a *Heuristic Non Parametric Collateral Missing Value Imputation* (HCMVI) algorithm that employs a combination of correlated genes to estimate missing values by multiple imputation matrices. The basis of HCMVI is CMVE technique that has been demonstrated both theoretically and empirically, to be better than established algorithms including KNN, LSImpute and BPCA [14]. However, like KNN and LSImpute, CMVE does not automatically determine the optimal number of predictor genes k from the dataset and this can lead to higher estimation errors. For data with a local correlation structure, if a large k value is used then it may include genes which have no correlation with the gene that has missing values. Similarly, if data has a global correlation structure, then a small value of k ignores correlated genes in the prediction again resulting in a higher estimation error. It is therefore intuitive to try and calculate the best value of k, based upon the underlying correlation structure of the data. LLSImpute automatically determines k using computational intensive exhaustive search method, hence provides improved results than other LS regression based methods [10], though since this approach is based upon LS regression, therefore, estimation error is still high because LS regression is sensitive to outliers [15, 16] (See Section 2). HCMVI uses CMVE as its

core kernel together with a heuristic non-parametric estimator, to automatically determine *k*, thereby combining the intrinsic benefits of heuristics and CMVE with a strategy to automatically estimate the optimal number of predictor genes. The estimation performance of HCMVI has been rigorously tested and compared with four other well-established imputation techniques, namely CMVE, KNN, LLSImpute (An enhanced version of LSImpute [12]) and BPCA in predicting randomly introduced missing values with probabilities ranging from 0.01 to 0.2 for six different ovarian and breast cancer datasets [17, 18]. To cross validate the performance of the different imputation strategies, six separate biological and statistical (both parametric and non-parametric) measures have been used to eliminate any bias towards a particular metric for a certain imputation methodology. The study in particular compared the impact of estimation on significant gene selection where HCMVI clearly demonstrated improved capability for both the breast cancer (locally correlated) and ovarian cancer (globally correlated) datasets. For instance, the KIAA1025 gene which is expressed in breast cancer cell lines and is co-regulated with several cancer causing genes such as estrogen receptors [19] was not selected when missing values were imputed using KNN, LLSImpute, BPCA and CMVE, but was correctly identified across a range of missing values when gene selection was preceded by HCMVI imputation (See supplementary materials[1]). For completeness, results are also compared using the conventional NRMSE [20] and *Wilcoxon Ranksum Significance Test* metrics to quantitatively assess the estimation performance of each imputation method, with results again consistently demonstrating the improved accuracy and robustness of HCMVI over the entire missing value range. The next Section presents an overview of the existing imputation strategies with their respective merits and demerits.

2 Overview of Existing Imputation Methods

The following nomenclature is adopted in describing different imputation methods. A microarray gene expression matrix *Y*, contains *m* genes and *n* samples. In *Y*, every gene *i* is represented by g_i. A missing value in gene *i* for sample *j* is thus expressed as $Y(i, j) = g_i(j) = \Xi$. A short overview is now provided of the main features of the four imputation methods (KNN, LLSImpute, BPCA and CMVE) which are used in this paper to compare the performance of HCMVI.

KNN [11] estimates missing values by searching for the *k* nearest genes normally using a Euclidean distance function, and then taking the weighted average of the *k* nearest genes. The method however, does not consider negative correlations [21] and has the drawback of using a predetermined value of *k* regardless of the dataset being used. Kim et al [10], introduced an improved Least Square regression based algorithm called *Local Least Square Impute* (LLSimpute), which automatically selects the number of predictor genes *k* using computational intensive exhaustive search method and then regresses using LS techniques to impute the missing values, though this regression makes the technique highly sensitive to outliers [15, 16] which leads to higher estimation errors (Section 1). BPCA [8] uses *Bayesian Principal Component*

[1] Supplementary Material: http://hcmvi.wiki.sourceforge.net

Analysis to impute missing values, though this only exploits global correlations within the data structure, which can lead to erroneous estimates if data possesses a strong local correlation [8]. CMVE algorithm generates multiple estimation matrices using *Non-Negative Least Square*s (NNLS), *Linear Programming* (LP) and LS regression techniques to approximate missing values, however despite its enhanced estimation capability, it still relies upon a preset parametric value of k, which limits its applicability. Importantly, despite their respective merits these imputation algorithms have not been analyzed on *a-priori* biological knowledge, which is ultimately a true evaluation for comparing imputation performance. This was the motivation to develop a new strategy to automatically determine the best value of k directly from the correlation structure of the data, while concomitantly providing significant improvement on both biological and statistical grounds. The next Section presents the HCMVI imputation technique which combines the estimation capability of CMVE with a strategy for deriving the optimal value of k directly from the correlation structure of the data.

3 Heuristic Non Parametric Collateral Missing Value Imputation Method

The HCMVI algorithm, which is formally presented in Fig. 1, imputes missing values in three stages. Firstly, the number of estimator genes k is computed using a *Heuristic Non-Parametric* algorithm that exploits data correlation structures. Secondly, the k most correlated genes with the gene (g_i) containing missing value are selected from a given dataset, before g_i is approximated using the CMVE algorithm and finally value is imputed using *Non Negative Least Square* and *Linear Programming*.

To select the number of estimator genes k, the set of sub-matrices SM is chosen (Step 1, Fig. 1) which has the highest correlation with the rest of data, since this best represents the underlying correlation structure for the entire data Y. To construct such correlated sub-matrix which optimally represents the correlation of the entire data Y is

Pre Condition: Gene expression matrix $Y(m,n)$ where m and n are the number of genes and samples respectively; actual missing value location δ.

STEP 1 Select a set of sub-matrices $SM \in \mathbb{R}^{m \times n}$ from Y using Monte Carlo simulation with uniform distribution.
STEP 2 FOR $i \leftarrow 1$ to R_w
  2.1 Compute mean G_w of gene expression vectors in sub-matrix SM_i
  2.2 Calculate mean for all corresponding gene expression vectors G_Y from Y selected in SM_i
  2.3 Determine Pearson correlation r_i between G_w and G_Y for sub-matrix SMi using (1)
STEP 3 Rank the sub-matrices SM based on the magnitude of the correlation coefficients r.
STEP 4 Select the sub-matrix SM_c with the highest r.
STEP 5 Select the expression locations v in Y which are present in SM_c
STEP 6 FOR $k \leftarrow 1$ to m
  6.1 Call *Estimate* using expression locations v and k as parameters
  6.2 Calculate NRMS error in (5) and save the corresponding k in θ
STEP 7 Sort θ in ascending order and select corresponding k as k_{opt} for actual missing value estimation.
STEP 8 Compute missing values using CMVE using δ *and* k_{opt} as parameters.
END

Post Condition: Y with no missing values.

Fig. 1. The complete HCMVI Algorithm

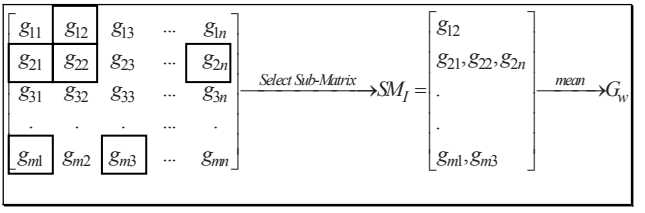


Fig. 2. A selected Sub-Matrix *SM* for determining the optimal value of *k*

an NP hard problem, so a pseudo-random generation strategy is adopted to select gene expressions from *Y* (Step 1, Fig. 1). A statistically conservative selection probability of 0.05 [22] is chosen for the objective function, such that there will be no missing values in the *SM* (See Fig. 2). In each sub-matrix SM_i, genes *G* that are present in SM_i (Step 2) are selected, such as $G = [G_1 ... G_n]^T \in \mathbb{R}^{m \times n}$ (Step 2). For example in Fig. 2, the SM_1 contains one gene expression value g_{12} from gene 1, three expression values $\{g_{21}, g_{22}, g_{2n}\}$ from gene 2, and two expression values from gene *m* $\{g_{m1}, g_{m3}\}$, while no expression was selected from gene 3 (which is normally very rare). All selected genes are represented by boxes in Fig. 2. This example highlights two key limitations for computing the correlation between *Y* and SM_i: 1) The number of columns of SM_i and *Y* are not equal 2) The number of rows of SM_i and *Y* are unequal because not all the genes in *Y* are present in SM_i, so in order to determine the correlation between *Y* and any sub-matrix SM_i, only those genes that are selected from *Y* and also present in SM_i, are chosen. Thus, in the Fig. 2 example, genes *1, 2* and *m* would be selected while gene *3* is ignored. The mean values G_{SM} and G_Y of the genes vector in both *Y* and SM_i are then computed (Step 2.1). The Pearson correlation of the data is calculated (Step 2.3) to determine the maximum correlation between each selected SM_i and *Y* from:

$$r_i = \frac{\sum G_w G_y - \frac{\sum G_w G_y}{N}}{\sqrt{\left(G_w^2 - \frac{(\sum G_w)^2}{N}\right)\left(G_y^2 - \frac{(\sum G_y)^2}{N}\right)}} \quad (1)$$

The reason for selecting Pearson correlation is that it has been consistently proven to provide better performance for microarray data compared with other similarity measures [23].

The sub-matrix SM_c with the maximum absolute correlation with *Y* represents the best correlation of *Y* (Step 4). Each individual value of SM_c is then treated as a missing value and iteratively estimated for a range of different *k* values (Step 5). Since, these values are already known; the NRMSE can be computed for these estimations, so the *k* value which generates the minimum *Root Mean Square Error* (RMSE) is designated as the optimal value (k_{opt}). This is subsequently used in the actual estimation of missing value Y_{ij} of gene *i* and sample *j*, which involves three separate estimates Φ_1, Φ_2 and Φ_3 being generated, and the final estimate χ computed by their fusion using CMVE [13]. The CMVE technique is explained in the context of HCMVI in Appendix A (See supplementary material).

4 Analysis and Discussion of Results

4.1 Test Data

To analyze and compare the performance of the proposed HCMVI algorithm with CMVE, BPCA, LLSImpute and KNN, six microarray cancer datasets from two different studies on breast and ovarian cancer tissues were used. The data was log transformed and normalized to $\bar{x}=0$ and $\sigma^2=1$ to remove experimental variations. The rationale behind selecting these particular datasets is that in general, cancer data lacks molecular homogeneity in tumour tissues so missing values are hard to predict in cancerous data [24].

The locally correlated breast cancer data set contained 7, 7, 8 samples of BRCA1, BRCA2 and Sporadic mutations (neither BRCA1 nor BRCA2) respectively [18], while the globally correlated ovarian cancer dataset contained 16, 16 and 18 samples respectively of BRCA1, BRCA2, Sporadic mutations [17]. Each breast cancer data sample contained microarray data of 3226 genes and there were 6445 genetic expressions per sample for the ovarian dataset.

To compare the performance of the HCMVI algorithms with CMVE and KNN, $k=10$ was used throughout the experiments, since the insensitivity of KNN to values of k in the range from 10 to 20 was observed by Troyanskaya et al, [11] who confirmed that the best estimation results were achieved in this interval and using a similar rationale CMVE [13] employed k=10. In contrast, LLSImpute determines the value of k using computational intensive exhaustive search method, while HCMVI automatically determines the optimal value of k_{opt} using a non-parametric heuristic algorithm (Fig. 1), which exploits the underlying correlation structure of the data, thereby reducing the computational complexity and avoiding problems of large and small values of k highlighted in Section 1. Following six metrics are used to evaluate the performance of the new HCMVI algorithm.

4.2 Gene Regulatory Network Reconstruction

To further evaluate the influence of missing values on GRN reconstruction, the ARACNe has been employed because a study in [25] demonstrated its improved performance compared to the commonly used algorithms, like Bayesian networks [26]. Moreover the method has been tested for mammalian gene network reconstruction and compared with other techniques that are normally applied to simple eukaryotes, such as *Saccharomyces cerevisiae* [27].

ARACNe firstly computes gene-gene co-regulation using mutual information. The method then prunes indirect regulatory relationships that are co-regulated by one or more intermediate genes using data processing inequality. To comparatively evaluate the respective imputation performances on GRN reconstruction, the number of *Conserved Links* was determined, which represents whether a particular co-regulation link is present in both GRN_{org} and $GRN_{imputed}$. The gene network GRN_{org} was initially constructed from the original data Y with no missing values using ARACNe. Iteratively, up to 20% missing values were randomly introduced and then respectively estimated using imputation methods. The corresponding gene networks $GRN_{imputed}$

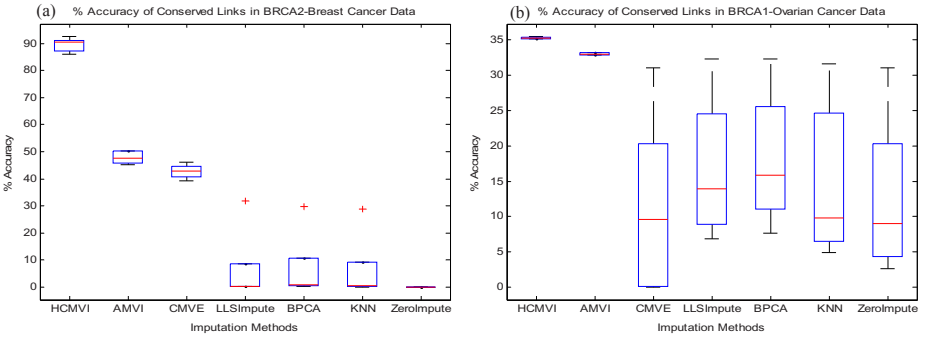


Fig. 3. Comparison of original GRN and the GRN constructed after imputation where missing values were randomly introduced in a) BRCA2-breast cancer data b) BRCA1-ovarian cancer data

were then constructed from the imputed data and GRN_{org} and $GRN_{imputed}$ compared, to ascertain the conserved links.

Fig.3 shows that the ARACNe method, which was reported to be robust [28] for GRN construction, could not maintain its performance in the presence of missing values, especially for ZeroImpute. In contrast, when HCMVI was applied, ARACNe conserved the number of links even at higher missing value probabilities. For example, in BRCA1 breast cancer data, the transcriptional link between *ADP-Ribosylation Factor 3* (ARF3) and *General Transcription Factor II, I, Pseudogene 1*(GTF2IP1) was overlooked when missing values were imputed by comparative methods (Fig. 3(a)), but was correctly inferred when values were imputed using HCMVI. Similarly, the link between *HS1 Binding Protein* and *Mitogen-Activated Protein Kinase 3* in BRCA2 breast cancer data was reconstructed when values were imputed using HCMVI but was neglected by all other imputation techniques. The results of Sporadic breast cancer data revealed similar observations. For example, the interaction between ARF3 and EST, which is similar to NSAP1 protein, was found when data was imputed using the HCMVI method. But it was missed by the other imputation strategies. These results further highlight the importance of accurate imputation in improving GRN reconstruction performance (See supplementary materials for details).

4.3 Gene Selection

This Section provides rigorous analytic review of gene selection results. Since, different gene selection methods produce different sets of significant genes [29] therefore; we compared the performance of imputation methods using two widely used methods namely: standard t-test and *Between Sum of Squares to Within Sum of Squares* (BSS/WSS).

4.3.1 Gene Selection Using t-Test

To investigate the impact of each estimation algorithm upon significant gene selection, genes were selected from both breast and ovarian cancer datasets using

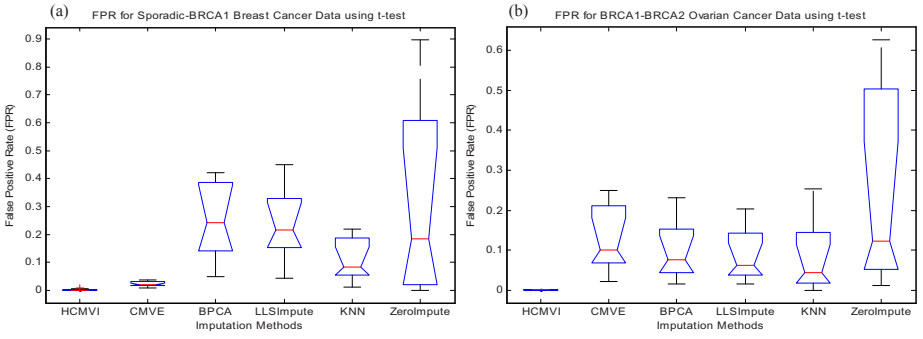


Fig. 4. False Positive Rate of Gene Selection in a) Breast and b) Ovarian Cancer Data

t-test [30] with the established statistical P-value of 0.05 [24]. These selected genes were then, marked as *Standard Genes*. After that the values were randomly removed from the data with the probability range of 0.01 to 0.20 and were marked as missing. These missing values were then, imputed using above mentioned imputation methods and followed by gene selection using t-test using the same P-value of 0.05. The selected genes were then compared with the *Standard Genes* to compute *False Positive Rate* (FPR) [31] using:

$$FPR = \frac{False\ Positive}{True\ Positive + False\ Positive} \tag{2}$$

Fig.4 demonstrates FPR of gene selection results after imputation by the above mentioned estimation methods (See supplementary materials for the rest of the results). The results show that HCMVI had minimum FPR for both the datasets, in all the selected groups, while most of the imputation methods could not retain their performance for all the datasets. For instance, CMVE showed better FPR for Breast cancer data (Fig. 4(a)) but since the method doesn't consider global correlations it could not hold similar performance for ovarian cancer data (Fig. 4(b)). Similarly, BPCA showed better performance for ovarian cancer data (Fig. 4(b)) due to its ability to exploit global correlation but couldn't retain the same performance for breast cancer data (Fig. 4(a)). Both the LLSImpute and KNN methods also, showed mixed performance while not surprisingly, ZeroImpute method had highest FPR due to its inability to exploit latent correlation of the data.

4.3.2 Gene Selection Using BSS/WSS

To investigate the impact of each estimation algorithm upon significant gene selection, a set of p genes (G_{org}) was selected from the original data Y using the BSS/WSS method [32], which identified those genes that concomitantly had large inter-class and small intra-class variations. For any gene i in $Y \in \mathbb{R}^{m \times n}$, BSS/WSS is calculated as follows:

$$BSS(i)/WSS(i) = \frac{\sum_{t=1}^{T}\sum_{q=1}^{Q} F(L_t = q)(\bar{Y}_{qi} - \bar{Y}_i)^2}{\sum_{t=1}^{T}\sum_{q=1}^{Q} F(L_t = q)(Y_{it} - \bar{Y}_{qi})^2}, \tag{3}$$

where T is the training sample size, Q the number of classes and $F(\bullet)$ is a Boolean function having value = 1 if the condition is TRUE and zero otherwise. Y_i denotes the average expression level of gene i across all samples and $\bar{Y}_{qi}$ is the average expression level of gene i across all samples belonging to class q. Genes are then ranked from the highest to the lowest BSS/WSS ratio to form a significant gene expression matrix ϑ, where the first p genes are selected for subsequent analysis.

To fully test the robustness of the HCMVI algorithm, experiments were performed for missing values up to 20%, with values being iteratively removed from the original gene expression matrix Y. Missing values were then estimated using KNN, LLSImpute, BPCA, CMVE and HCMVI to form Y_{est}, before respective sets of p genes G_{est} were selected using BSS/WSS, for each estimated matrix. Finally, these selected genes were compared with G_{org} to give the *%Accuracy*.

To eliminate performance variations with respect to the number of selected genes in the BSS/WSS method, all imputation techniques were tested for 1000 and 50 significant genes with the graphs in Fig. 5 displaying the gene selection performance for 50 significant genes (Supplementary materials for the results on 1000 significant genes). The results reveal the consistent better performance of HCMVI over the other imputation methodologies because of its ability to exploit both local and global correlations within the data. HCMVI performed equally well for both types of data with the average overall improvement being 60% and 48% for breast and ovarian cancer datasets respectively. The results also highlighted some other noteworthy points: even though HCMVI consistently performed better than comparative algorithms, its performance was better for the breast cancer dataset than ovarian cancer because the latter contained some actual missing values which influence gene selection. The CMVE algorithm using a fixed k performed better than BPCA, LLSImpute, KNN and ZeroImpute (Fig. 5(a)) for the locally correlated Breast cancer data, but was unable to maintain this performance for the ovarian cancer (globally correlated) data (Fig. 5(b)). Similarly, BPCA performed better than the aforementioned algorithms for the ovarian cancer data because of its inherent ability to exploit global correlations, though in contrast its performance deteriorated significantly for breast cancer data. The next Section focuses on non-parametric significance test results.

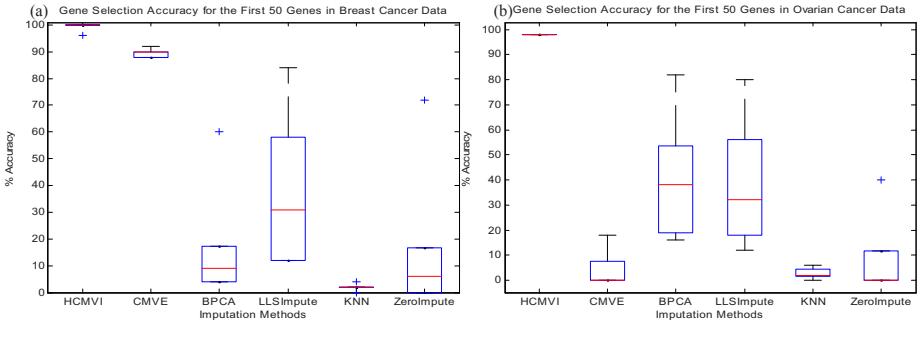


Fig. 5. Gene Selection Accuracy for 50 Significant Genes in a) Breast and b) Ovarian Cancer

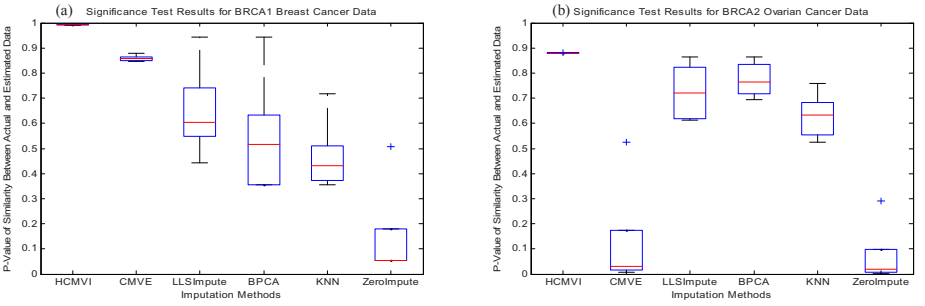


Fig. 6. Significance Test Results for a) BRCA1-Breast and b) BRCA-2 Ovarian Cancer Data

4.4 Wilcoxon Rank Sum Significance Test

To evaluate the estimation performance of all the imputation algorithms on empirical grounds and variance stability, the *two-sided Wilcoxon Rank sum statistical significance* test was applied. The motivation for using this particular test is that compared to some other parametric significance tests such as *t-test* [33], it does not mandate the data has to have equal variance, which is vital given the variance of data can be disturbed due to erroneous estimation, especially for ZeroImpute. To test the hypothesis H_0, $Y = Y_{est}$ where Y and Y_{est} are the actual and estimated matrices respectively, the *P-Value* of the hypothesis is calculated as:

$$H_0,\ P\text{-}Value = 1 - 2P_r(R \leq y_r) \tag{4}$$

where y_r is the sum of the ranks of observations for Y and R is the corresponding random variable. Fig. 6 plots the *P-Value* of similarity between the actual and estimated matrices. The results again corroborate that HCMVI performed better than all the other comparative algorithms for both locally and globally correlated datasets because of its better estimation capability. A similar trend is observed by the imputation strategies in terms of the statistical significance test results as witnessed for significant gene selection in the above Section. It is interesting to note that HCMVI performed consistently better for all the datasets, as shown in Fig. 6, where the performance of other algorithms was highly data dependent.

Interestingly, HCMVI proved to be robust for both cancer datasets which is certainly not the case for other imputation techniques who performed well for one data type, but failed for the other (Fig.3-6). For instance, CMVE, generally performed better than BPCA, KNN, ZeroImpute, and LLSImpute for breast cancer data (Figs. 3(a) - 6(a)), but this was not sustained for ovarian cancer data (Figs. 3(b) - 6(b)), where BPCA proved a better choice for this globally correlated dataset. Not surprisingly, ZeroImpute exhibited the widest disparity on statistical grounds, so inculcating the importance of estimating any missing values rather than simply imputing zeros.

4.5 Normalized Root Mean Square Error

For completeness the estimation performance of HCMVI and comparative imputation methods was also analyzed using the traditional parametric *Normalized Root Mean*

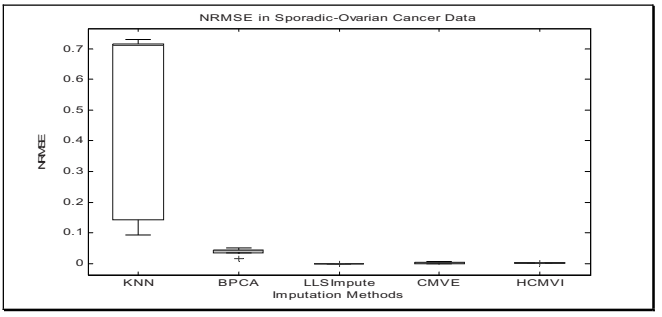


Fig. 7. NRMSE in Sporadic-Ovarian Cancer Data

Square Error (NRMSE) measure, despite its limitations in reflecting the true impact of missing values on subsequent biological analysis. NRMSE is defined as:

$$\Theta = \frac{RMS(Y - Y_{est})}{RMS(Y)} \tag{5}$$

where Y is the original data matrix and Y_{est} is the estimated matrix using HCMVI, CMVE, BPCA, LLSImpute and KNN respectively. This particular measure has been used by Sehgal et al, [13], Ouyang et al, [20] and Tuikkala et al [7] for error estimation because $\Theta = 1$ for zero imputation.

Fig.7 shows boxplot of NMRS Error for different imputation algorithms (See supplementary material for the rest of the results). It again confirms the better performance of HCMVI and reiterates the value of accurately exploiting information about the underlying correlation structure of the data instead of using a preset value. Interestingly LLSImpute exhibited similar performance to HCMVI so justifying the merit of using other metrics to dispassionately compare the performance of different imputation strategies. As highlighted earlier, accurate imputation plays a crucial role in selecting the correct set of genes for a given biological process, so an analysis of the biological significance of the various imputation results has been undertaken with the key finings presented in the next Section.

4.6 Biological Significance

As alluded to earlier, different analytical methods will by virtue of their underlying assumptions generate differing gene lists, so an attempt has been made to assess the significance of the results for HCMVI from a biological perspective. Any superior imputation technique can be reasonably expected to return genes that have been implicated in the biological process when independent experiments are studied. Indeed, as microarray experiments effectively serve as a hypothesis generation step, it is constructive to ascertain whether a method not only identifies known genes, but also novel genes including hypothetical ones, about which little is known so that appropriate additional experiments can be performed. This may provide not only valuable information for the design of basic mechanistic, diagnostic and biomarker studies, but also further data for use in the construction of gene networks and pathways involved in processes like oncogenesis and resistance to tumour induction.

Table 1. KIAA1025 (KIAA) and Plakophilin2 (PKP2) Selection in Breast Cancer Dataset and MHC Class II=DQ alpha (MHCα) and MHC Class II=DQ beta (MHCβ) Selection in Ovarian Cancer across the Range of Missing Values Across the Range of Missing Values

% MV	HCMVI	CMVE	LLSImpute	BPCA	KNN	ZeroImpute
1	KIAA PKP2 MHCα MHCβ	KIAA PKP2 MHCα	KIAA MHCα	 MHCα		KIAA MHCα
5	KIAA PKP2 MHCα MHCβ	KIAA PKP2 MHCβ	KIAA			KIAA
10	KIAA PKP2 MHCα MHCβ	KIAA PKP2				
15	KIAA PKP2 MHCα MHCβ	KIAA PKP2				
20	KIAA PKP2 MHCα MHCβ					

While the final validation of HCMVI as an imputation strategy will only be truly achieved when the role of newly predicted genes are validated in biological experiments, it is instructive to examine the list of candidate genes to determine whether any are independently validated.

In examining both the breast and ovarian cancer datasets, HCMVI identified a number of genes overlooked by all the other algorithms and which, independent experiments [34] confirm, alter expressions in tumor lines and so could be important in oncogenesis. This set of genes has not only been selected by BSS/WSS algorithm but has been revalidated using the *modified t-test with greedy pairs* method [35] which minimizes the bias of the gene selection strategy towards either a particular imputation technique or a set of genes.

For example, as the results in Table 1 reveal, the KIAA1025 protein has not been selected when values are imputed using KNN, LLSImpute, BPCA and CMVE, but has been identified when gene selection is preceded by HCMVI imputation (See also supplementary material). This is an important protein which is co-regulated with estrogen receptors for both in vivo and clinical data, which are expressed in more than 66% of human breast tumors [19]. Another gene selected by HCMVI across the range of missing values is plakophilin 2 (PKP2) which is a common protein and exhibits a dual role, appearing as both a constitutive karyoplasmic protein and a desmosomal

plaque component for all the desmosome-possessing tissues and cell culture lines. The gene is found in breast carcinoma cell lines [36] and furthermore, because of its significance it can serve as a marker for the identification and characterisation of carcinomas derived either from or corresponding, to simple or complex epithelia [37] (See Table 1).

Similar observations can be made in the study of significant genes in the ovarian cancer dataset. For instance, MHC Class II=DQ alpha (MHCα) and MHC Class II=DQ beta (MHCβ) genes are linked to the immune system and have been shown to be down-regulated for ovary syndrome [38]. Also, the allele gene is present at a higher frequency in patients with malignant melanoma than in Caucasian controls. These genes help in particular to diagnose melanoma patients in the relatively advanced stages of the disease and/or patients who are more likely to have a recurrence [39]. The results reveal that these genes have been correctly identified by HCMVI while being consistently missed by other imputation methods, especially for higher numbers of missing values (See Table 1 and supplementary material).

For both cancer datasets, in every case these regulated genes have been correctly identified when gene selection followed imputation by HCMVI for the full range of missing values from 1% to 20% as confirmed in Tables 1 and 2. Summarizing, these biological significance results demonstrate the robustness of the HCMVI algorithm in correctly estimating missing values for different data types by adapting to both the causal global and local correlation structures of the data in contrast to all other imputation algorithms, especially for higher numbers of missing values.

5 Conclusions

This paper has presented a new *Heuristic Non Parametric Collateral Missing Value Imputation* (HCMVI) algorithm based upon the concept of constructing an optimal sub-matrix of the most correlated genes to determine the optimal value of k predictor genes to be applied in the imputation process. HCMVI has demonstrated an ability to adapt to both the local and global correlations, with experimental results for gene selection, statistical significance tests, biological significance and the *Normalized Root Mean Square Error* measure proving that it provided lower estimation errors compared to existing missing value imputation algorithms including CMVE, KNN, LLSImpute, ZeroImpute and BPCA. In particular, GRN reconstruction results showed the improved performance of 90% and 35% for both breast and ovarian cancer data. Similarly, gene selection results revealed an overall improved selection performance of 60% and 48% respectively for breast and ovarian cancer data, while the biological significance results upon selected genes demonstrated that key breast cancer genes like plakophilin2, KIAA1025 and MHC Class II=DQ are consistently correctly identified by HCMVI for the full range of missing values, while being overlooked by other imputation methods. The HCMVI strategy of exploiting a combination of underlying correlations in a dataset together with the automatic selection of the optimal k using a *Heuristic Non Parametric* approach has proven to be more effective, less computational intensive and robust than using either a preset k value or determining its value by exhaustive search.

References

[1] Furey, T.S., Cristianini, N., Duffy, N., Bednarski, D.W., Schummer, M., Haussler, D.: Support vector machine classification and validation of cancer tissue samples using microarray expression data. Bioinformatics 16(10), 906–914 (2004)
[2] Gustavo, B., Monard, C.M.: An analysis of four missing data treatment methods for supervised learning. Applied Artificial Intelligence 17(5-6), 519–533 (2003)
[3] Ramaswamy, S., Tamayo, P., Rifkin, R., et al.: Multiclass cancer diagnosis using tumour gene expression signatures. Proc. Natl. Acad. Sci. 98(26), 15149–15154 (2001)
[4] Shipp, M.A., Ross, K.N., Tamayo, P., et al.: Diffuse large B-cell lymphoma outcome prediction by gene expression profiling and supervised machine learning. Nat. Med. 8(1), 68–74 (2002)
[5] Golub, T.R., Slonim, D.K., Tamayo, P., et al.: Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. Science 286(5439), 531–537 (1999)
[6] Munagala, K., Tibshiran, R., Brown, P.O.: Cancer characterization and feature set extraction by discriminative margin clustering. BMC Bioinformatics 5, 21 (2004)
[7] Tuikkala, J., Elo, L., Nevalainen, O.S., Aittokallio, T.: Improving missing value estimation in microarray data with gene ontology. Bioinformatics, 566–572 (2005)
[8] Oba, S., Sato, M.A., Takemasa, I., Monden, M., Matsubara, K., Ishii, S.: A Bayesian Missing Value Estimation Method for Gene Expression Profile Data. Bioinformatics 19, 2088–2096 (2003)
[9] Acuna, E., Rodriguez, C.: The treatment of missing values and its effect in the classifier accuracy. Classification, Clustering and Data Mining Applications, 639–648 (2004)
[10] 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)
[11] Troyanskaya, O., Cantor, M., Sherlock, G., Brown, P., Hastie, T., Tibshirani, R., Botstein, D., Altman, R.: Missing Value Estimation Methods for DNA Microarrays. Bioinformatics 17, 520–525 (2001)
[12] Bø, T.H., Dysvik, B., Jonassen, I.: LSimpute: Accurate estimation of missing values in microarray data with least squares methods. Nucleic Acids Res. 32(3), 34 (2004)
[13] Sehgal, M.S.B., Gondal, I., Dooley, L.: Collateral Missing Value Imputation: a new robust missing value estimation algorithm for microarray data. Bioinformatics 21(10), 2417–2423 (2005)
[14] Sehgal, M.S.B., Gondal, I., Dooley, L.: Missing Value Imputation Framework for Microarray Significant Gene Selection and Class Prediction. In: Li, J., Yang, Q., Tan, A.-H. (eds.) BioDM 2006. LNCS (LNBI), vol. 3916, pp. 131–142. Springer, Heidelberg (2006)
[15] Stevens, J.P.: Applied Multivariate Statistics for the Social Sciences. LEA, Inc (2001)
[16] Voelker, D.H., Orton, P.Z., Adams, S.: Statistics. Cliffs Notes (2001)
[17] Amir, A.J., Yee, C.J., Sotiriou, C., et al.: Gene Expression Profiles of Brca1-Linked, Brca2-Linked, and Sporadic Ovarian Cancers. Journal of the National Cancer Institute 94(13) (2002)
[18] Hedenfalk, I., Duggan, D., Chen, Y., Borg, A., Trent, J., et al.: Gene-expression profiles in hereditary breast cance. N. Engl. J. Med. 22;344(8), 539–548 (2001)
[19] Harvell, D.M.E., Richer, J.K., Allred, D.C., Sartorius, C.A., Horwitz, K.B.: Estradiol Regulates Different Genes in Human Breast Tumor Xenografts Compared with the Identical Cells in Culture. Endocrinology 147, 700–713 (2006)

[20] Ouyang, M., Welsh, W.J., Georgopoulos, P.: Gaussian Mixture Clustering and Imputation of Microarray Data. Bioinformatics 20(6), 917–923 (2004)
[21] Sehgal, M.S.B., Gondal, I., Dooley, L.: A Collateral Missing Value Estimation Algorithm for DNA Microarrays. In: IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), USA, pp. 377–380 (2005)
[22] Abelson, R.P.: Statistics as Principled Argument. Lawrence Erlbaum Associates, Mahwah (1995)
[23] Yona, G., Dirks, W., Rahman, S., Lin, D.M.: Effective similarity measures for expression profiles. Bioinformatics 22, 1616–1622 (2006)
[24] Jornsten, R., Wang, H.-Y., Welsh, W.J., Ouyang, M.: DNA microarray data imputation and significance analysis of differential expression. Bioinformatics 21, 4155–4161 (2005)
[25] Basso, K., Margolin, A.A., Stolovitzky, G., Klein, U., Dalla-Favera, R., Califano, A.: Reverse engineering of regulatory networks in human B cells. Nature Genetics 37, 382–390 (2005)
[26] Jensen, F.V.: Bayesian Networks and Decision Graphs, 2nd edn. Springer, Heidelberg (2002)
[27] Ihmels, J., Levy, R., Barkai, N.: Principles of transcriptional control in the metabolic network of Saccharomyces cerevisiae. Nature Biotechnology 22, 86–92 (2003)
[28] Margolin, A.A., Nemenman, I., Basso, K., et al.: ARACNE: An Algorithm for the Reconstruction of Gene Regulatory Networks in a Mammalian Cellular Context. BMC Bioinformatics 7 (2006)
[29] Jeffery, I.B., Higgins, D.G., Culhane2, A.C.: Comparison and evaluation of methods for generating differentially expressed gene lists from microarray data. BMC Bioinformatics 7 (2006)
[30] Eschrich, S., Yeatman, T.J.: DNA Microarrays and Data Analysis: An Overview. Surgery, ELSEVIER 136, 500–503 (2004)
[31] Jornsten, R., Wang, H.-Y., Welsh, W.J., Ouyang, M.: DNA microarray data imputation and significance analysis of differential expression. Bioinformatics 21, 4155–4161 (2005)
[32] 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, 77–78 (2002)
[33] Sidak, Z., Sen, P.K., Hajek, J.: Theory of Rank Tests (Probability and Mathematical Statistics). Academic Press, London (1999)
[34] Salceda, S., Drumright, C., DiEgidio, A., et al.: Identification of differentially expressed genes in breast cancer. Nature Genetics 27, 83–84 (2001)
[35] Bø, T.H., Jonassen, I.: New feature subset selection procedures for classification of expression profiles. Genome Biology 3(4), research0017.1–research0017.11 (2002)
[36] Mertens, C., Kuhn, C., Franke, W.: Plakophilins 2a and 2b: constitutive proteins of dual location in the karyoplasm and the desmosomal plaque. J. Cell Biol. 135, 1009–1025 (1996)
[37] Mertens, C., Kuhn, C., Moll, R., Schwetlick, I., Franke, W.W.: Desmosomal plakophilin 2 as a differentiation marker in normal and malignant tissues. Differentiation 64, 277–290 (1999)
[38] Jansen, E., Laven, J.S.E., Dommerholt, H.B.R., et al.: Abnormal Gene Expression Profiles in Human Ovaries from Polycystic Ovary Syndrome Patients. Mol. Endocrinol 18, 3050–3063 (2004)
[39] Lu, M., Thompson, W.A., Lawlor, D.A., Reveille, J.D., Lee, J.E.: Rapid direct determination of HLA-DQB1 * 0301 in the whole blood of normal individuals and cancer patients by specific polymerase chain reaction amplification. Journal of Immunological Methods 199, 61–68 (1996)

Protein Expression Molecular Pattern Discovery by Nonnegative Principal Component Analysis

Xiaoxu Han[1] and Joseph Scazzero[2]

[1] Department of Mathematics and Bioinformatics Program
[2] Department of Accounting and Finance
Eastern Michigan University, Ypsilanti MI 48197, USA
{xiaoxu.han,jscazzero}@emich.edu

Abstract. Identifying cancer molecular patterns robustly from large dimensional protein expression data not only has significant impacts on clinical ontology, but also presents a challenge for statistical learning. Principal component analysis (PCA) is a widely used feature selection algorithm and generally integrated with classic classification algorithms to conduct cancer molecular pattern discovery. However, its holistic mechanism prevents local data characteristics capture in feature selection. This may lead to the increase of misclassification rates and affect robustness of cancer molecular diagnostics. In this study, we develop a nonnegative principal component analysis (NPCA) algorithm and propose a NPCA-based SVM algorithm with sparse coding in the cancer molecular pattern analysis of proteomics data. We report leading classification results from this novel algorithm in predicting cancer molecular patterns of three benchmark proteomics datasets, under 100 trials of 50% hold-out and leave one out cross validations, by directly comparing its performances with those of the PCA-SVM, NMF-SVM, SVM, k-NN and PCA-LDA classification algorithms with respect to classification rates, sensitivities and specificities. Our algorithm also overcomes the overfitting problem in the SVM and PCA-SVM classifications and provides exceptional sensitivities and specificities.

Keywords: Nonnegative principle component analysis, sparse coding, support vector machine (SVM).

1 Introduction

Molecular diagnostics has been challenging traditional cancer diagnostics in oncology by generating gene/protein expression data from a patient's tissue, serum or plasma samples through the DNA and protein array technologies. In clinical oncology, the gene/protein expressions are molecular patterns of cancers, reflecting gene/protein activity patterns in different types of cancerous or precancerous cells. However, robustly classifying cancer molecular patterns to support clinical decision making in early cancer diagnostics is still a challenge because of the special characteristics of gene and protein expression data. In this study, we focus on the mass spectrometry based protein expression data (MS data).

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 388–399, 2008.

Similar to general gene expression data, MS data can have large or even huge dimensionalities. It can be represented by a n×m matrix, each row of which represents the intensity values of a measured data point at a mass charge ratio (m/z) across different biological samples; each column of which represents the intensity values of all measured data points at different m/z values in a sample. Generally, the total number of measured data points is in the order of $10^5 \sim 10^6$ and the total number of biological samples is in the magnitude of hundreds, i.e., the number of variables is much greater than the number of biological samples. Although there are a large number of variables in these data, only a small set of variables have meaningful contributions to the data variations. Actually, these high-dimensional data are not noise-free. This is because the raw data contains systematic noise and the preprocessing algorithms can not remove it completely.

1.1 Principal Component Analysis Is a Holistic Feature Selection Algorithm

Many feature selection algorithms are employed to reduce protein expression data dimensions and decrease data noise before further classification or clustering [1,2]. Principal component analysis (PCA) is a commonly used approach among them [3,4]. It projects data in an orthogonal subspace generated by the eigenvectors of the data covariance or correlation matrix. The data representation in the subspace is uncorrelated and the maximum variance direction-based subspace spanning guarantees the least information loss in the feature selection. However, as a holistic feature selection algorithm, PCA can only capture the global characteristics of data instead of local characteristics of data. This leads to difficulty in interpreting each principal component (PC) intuitively, because each PC contains some levels of global characteristics of data. In the cancer pattern analysis of proteomics data, the holistic mechanism will prevent the following supervised/unsupervised learning algorithm from capturing the local behaviors of proteomics data in the clustering/classification. This would lead to the increase of misclassification rates and finally affect the robustness of the cancer molecular diagnostics.

One main reason for the holistic mechanism of the PCA is that data representation in the classic PCA is not '*purely additive*', i.e. the linear combination in the PCA contains both positive and negative weights and each PC consists of both negative and positive entries. The positive and negative weights are likely to cancel each other partially in the data representation. In fact, it is more likely that weights contributing from local features are partially cancelled out because of their frequencies. This directly leads to the holistic feature selection characteristics in the PCA.

Imposing nonnegative constraints on the PCA can remove the likelihood of the partial cancellation and make data representation consists of only additive components. In addition, it also contributes to sparse data representation. In the context of feature selection, adding nonnegative constraints on the PCA can improve the data locality in feature selection and make the data latent structure explicit.

Adding nonnegativity on the PCA is also motivated by the cancer molecular pattern discovery itself, i.e., protein expression data generally are represented as positive or nonnegative matrices naturally or after simple preprocessing. It is reasonable to require their corresponding dimension reduction data to be positive or at least

nonnegative to maintain data locality in order to catch more subtle or local behaviors in the following clustering or classification-based pattern discovery.

In this study, we present the nonnegative principal component analysis (NPCA) algorithm and demonstrate the superiority of the NPCA-based SVM classification algorithm (NPCA-SVM) with sparse coding, for three benchmark mass spectral serum datasets, by directly comparing it with five other similar classification algorithms, i.e., SVM, PCA-SVM, NMF-SVM, k-NN and PCA-LDA. This paper is organized as follows. Section 2 presents the nonnegative principal component analysis (NPCA) and NPCA-based SVM classification. Section 3 gives the experimental results of the NPCA-based SVM algorithm with sparse coding under 100 trials of 50% holdout cross validations for each dataset. It also compares the NPCA-SVM algorithm with the other five classification algorithms for the same training and test datasets. Finally, Section 4 concludes the paper.

2 Nonnegative PCA-Based Classification

Nonnegative PCA can be viewed as an extension of classic PCA by imposing PCA with nonnegativity constraints to capture data locality in the feature selection. Let $X = (x_1, x_2, \cdots x_n)$, $x_i \in \Re^d$, be a zero mean dataset, i.e., $\sum_{i=1}^{n} x_i = 0$. Then, the nonnegative PCA can be formulated as a constrained optimization problem to find maximum variance directions under nonnegative constraints as follows.

$$\begin{aligned} &\max\ J(U) = \frac{1}{2}\left\|U^T X\right\|_F^2, \quad s.t. \\ &U^T U = I,\ U \geq 0 \end{aligned} \tag{1}$$

where $U = [u_1, u_2, \cdots u_k]$, $k \leq d$, is a set of nonnegative PCs. The square Frobenius norm for a matrix A is defined as $\|A\|_F^2 = \sum_{i,j} a_{ij}^2 = trace(AA^T)$.

In fact, the rigorous orthonormal constraint under non-negativity is too strict for the practical cancer molecular pattern analysis, because it requires only one nonnegative entry in each column of U. The quadratic programming problem with the orthonormal-nonnegativity condition can be further relaxed as

$$\max_{U \geq 0} J(U, \alpha) = \frac{1}{2}\left\|U^T X\right\|_F^2 - \alpha\left\|I - U^T U\right\|_F^2 \tag{2}$$

where $\alpha \geq 0$ is a parameter to control the orthonormal degree of each column of U. After relaxation, matrix U is a near-orthonormal nonnegative matrix, i.e., $U^T U \sim I$. Computing the gradient of the objective function with respective to U, we have

$$U(t+1) = U(t) - \eta(t)\nabla_U J(t), \quad U \geq 0 \tag{3}$$

where $\nabla_U J(U,\alpha) = (U^T X)X^T + 2\alpha(I - U^T U)U$ and $\eta(t)$ is the iteration step size in the t time level iteration. For convenience, we select the step size in the iteration as 1. In fact, this is equivalent to finding the local maximum of a function $f(u_{sl})$ under the conditions: $u_{sl} \geq 0$, $s = 1,2\cdots d; l = 1,2\cdots n$, in the scalar level.

$$\max_{u_{sl}\geq 0} f(u_{sl}) = -\alpha u_{sl}^4 + c_2 u_{sl}^2 + c_1 u_{sl} + c_0 \tag{4}$$

where c_2 and c_1 are the coefficients of the u_{sl}^2 and u_{sl}; c_0 is the sum of the constant items independent of u_{sl}. The local maximum finding of the equation (4) is actually a set of cubic polynomial nonnegative root finding. Computing the stationary points for the scalar function $f(u_{sl})$, we have a set of cubic function root finding problems: $p(u_{sl}) = df(u_{sl})/du_{sl} = 0$ (see the appendix for details). The final U matrix is a set of nonnegative roots of the equation. By collecting the coefficients of u_{sl} and u_{sl}^2, we have

$$c_2 = \frac{1}{2}\sum_{i=1}^{n} x_{si}^2 - \alpha \sum_{j=1,\, j\neq l}^{k} u_{sj}^2 - 2\alpha \sum_{t=1, t\neq s}^{d} u_{tl}^2 + 2\alpha \tag{5}$$

$$c_1 = \sum_{i=1}^{n} \sum_{t=1, t\neq s}^{d} x_{sl} u_{tl} x_{ti} - 2\alpha \sum_{j=1,\, j\neq l}^{k} \sum_{t=1, t\neq s}^{d} u_{sj} u_{tl} u_{tj} \tag{6}$$

Actually, the constant term $c_0 = -k\alpha$ does not affect the entries of the matrix U. Only coefficients c_1 and c_2 are involved in the nonnegative root finding. The algorithm complexity of NPCA is $O(dkn \times N)$, where N is the total iteration number used in the algorithm. The detailed parameter derivations about equation 5 and 6 can be found in the appendix. Some authors also proposed a similar approach to solve the nonlinear optimization problem induced by a nonnegative sparse PCA [5]. However, their results lack technical soundness in the key parameter derivations.

2.1 NPCA-Based Classifications

The NPCA-based cancer molecular pattern classification employs the nonnegative principal component analysis (NPCA) to obtain a nonnegative representation of each sample in a low-dimensional, purely additive subspace spanned by the meta-variables first. A meta-variable is a linear combination of the intensity values of the measured data points for the MS data. The nonnegative representation for each sample is called a meta-sample, which is the prototype of the original sample with small dimensionalities. Then, a classification algorithm π_a, which is the SVM algorithm in this study, is applied to the meta-samples to gain classification information.

Theoretically, NPCA-based classification is rooted from a special nonnegative matrix factorization (NMF) [6] that we propose in this study: the nonnegative principal component induced NMF. We brief the principle of the NPCA-induced NMF as follows.

Let $X \in \Re^{d \times n}, d \ll n$, be a nonnegative matrix, which is a protein expression dataset with d number of samples for n number of measured points. Let $U \in \Re^{d \times d}$ be the nonnegative PCs, a near-orthogonal matrix for X before any further dimension selection. Projecting X^T into the purely additive subspace generated by U, we obtain the nonnegative projection $X^T U = P$. Alternatively, considering the PC matrix U is a near-orthogonal matrix, we can view it as an orthogonal matrix to decompose the data matrix, i.e., $X^T \sim PU^T$, where the nonnegative matrix P is equivalent to the basis matrix W and matrix U^T is equivalent to the feature matrix H in the classic NMF: $X \sim WH$. Similarly, the decomposition rank r in the NMF is the corresponding selected dimensionality in the nonnegative principal component analysis.

The NPCA-induced NMF can be also explained as follows. Each row of U is the corresponding meta-sample of each sample of X in the meta-variable space: $X_i^T \sim PU_i^T$. The meta-variable space is a subspace generated by columns of the basis matrix P, where each column/basis is a meta-variable. The meta-variable space is a purely additive space where each variable can be represented as the nonnegative linear combination of meta-variables as shown below.

$$X_i^T = \sum_{j=1}^{r} U_{ij}^T P_j, 1 \le r \le d \tag{7}$$

Based on the observation that proteomics data are nonnegative data or can be converted to corresponding nonnegative data easily, we have the NPCA-based SVM classification algorithm for proteomics data, i.e., starting from the NPCA-induced NMF for the protein expression dataset X, we input the corresponding normalized meta-samples $U = U / \|U\|_2$ to the SVM algorithm to conduct classification.

2.2 Sparse-Coding

To improve the generality of the NPCA-SVM classification algorithm, we conduct a sparse coding for the nonnegative PC matrix U. The sparseness of a nonnegative vector v with n tuples is a ratio between 0 and 1, which is defined in the equation (8) according to the relationship of two norms [7].

$$sparseness(v) = \frac{\sqrt{n} - \|v\|_1 / \|v\|_2}{\sqrt{n} - 1} \tag{8}$$

The sparse coding of the nonnegative PC matrix U finds the corresponding nonnegative vector satisfying the specified sparseness degree for each row $U_i^T, i = 1, 2 \cdots k$. In other words, for each row vector $x \ge 0$, the nearest vector $v \ge 0$ in the Euclidean sense is found that achieves a specified sparseness s. For convenience, we first normalize the nonnegative vector x such that $\|x\|_2 = 1$ before the sparse coding. Then, we project x into the hyperplane: $\sum v_i = \|x\|_1$ and compute the nonnegative

intersection point with the hypersphere $\sum v_i^2 = 1$ under the condition $s = (\sqrt{n} - \|x\|_1) / (\sqrt{n} - 1)$ in real time to finish its sparse coding.

2.3 Cross Validations

Since different training sets will affect classification results of a classification algorithm, we conducted the NPCA-SVM classification under the *50% holdout* cross validation 100 times, i.e., 100 sets of training and test datasets are generated randomly for each cancer dataset in the classification, to evaluate the expected classification performances. To improve computing efficiency, the PC matrix *U* in the nonnegative principal component analysis (NPCA) is cached from the previous trial and used as the initial point to compute the next principal component matrix in the computation.

3 Experimental Results

Our experimental data consists of three mass spectral serum profiles: *Ovarian*, *Ovarian-qaqc* (quality assurance/quality control) and *Liver* [8,9]. These datasets include one low resolution dataset and two high resolution datasets. They are generated from the Surface-Enhanced Laser Desorption/Ionization Time-Of-Flight (SELDI-TOF) and SELDI-QaTOF (a hybrid quadrupole time-of-flight mass spectrometry) technologies respectively. The detailed information about the datasets is given in the Table 1.

Table 1. Three Mass Spectral Serum Profiles.

Dataset	Data type	#M/Z	#Samples
Ovarian	SELDI-TOF Low resolution	15142	91 controls 162 cancers
Ovarian-qaqc	SELDI-TOF High resolution	15000	95 controls 121 cancers
Liver	SELDI-QqTOF High resolution	6710	181 controls 176 cancers

3.1 Preprocessing and Basic Feature Selection

We conducted the basic preprocessing steps for each mass spectrometry dataset: spectrum calibration, baseline correction, smoothing, peak identification, intensity normalization, and peak alignments. In addition, we employed the two-sided t-test to conduct basic feature selection for the three proteomics datasets before classifications. After the basic feature selection, *3780*, *2000* and *3000* most significant features are selected for the 1st, 2nd and 3rd dataset respectively, before further classifications.

3.2 Classifications

We compared the classification results from the NPCA-SVM algorithm under the sparse coding (α=10, sparseness=0.20) with the PCA-SVM and SVM algorithm under

Table 2. Average classification performance of three algorithms

	Average Sensitivity	Average Specificity	Average Classifying rates
Ovarian			
npca-svm-linear	98.35±1.03	99.98±0.24	98.94±0.65
npca-svm-rbf	100.0±0.0	99.42±0.99	99.79±0.35
svm-linear	100.0±0.0	98.63±2.21	99.50±0.83
svm-rbf	100.0±0.0	0.0±0.0	64.13±2.88
pca-svm-linear	99.98±0.17	99.93±0.51	99.96±0.26
pca-svm-rbf	100.0±0.0	0.0±0.0	64.13±2.88
Ovarian-qaqc			
npca-svm-linear	98.01±1.94	99.27±0.90	98.70±0.89
npca-svm-rbf	98.11±2.25	99.57±0.82	98.91±0.98
svm-linear	96.16±3.52	96.97±2.19	96.57±1.99
svm-rbf	97.00±17.18	3.00±17.18	54.92±44.8
pca-svm-linear	97.14±2.16	97.94±1.57	97.12±1.17
pca-svm-rbf	3.20±17.22	96.80±17.22	54.95±44.7
Liver			
npca-svm-linear	97.68±1.71	94.40±2.22	96.02±1.35
npca-svm-rbf	98.35±1.67	96.20±2.01	97.25±1.30
svm-linear	92.57±3.84	91.04±3.76	91.78±2.27
svm-rbf	38.00±48.78	62.00±48.78	47.92±2.00
pca-svm-linear	90.96±3.69	89.57±3.56	90.21±1.99
pca-svm-rbf	38.00±48.78	62.00±48.78	47.92±2.00

linear and Gaussian kernels, for each proteomics dataset under the *same* 100 sets of training and test data (trials). The 100 trials of training/test data for each dataset are generated under the 50% holdout cross validations. The average classification rates, sensitivities and specificities and their corresponding standard deviations from each classification algorithm are given in the Table 2.

From the classification results, we can make the following observations. 1. It is clear that the PCA-SVM, SVM classification algorithms suffer from overfitting under a Gaussian ('*rbf*') kernel. This is due to the complementary results of the sensitivities and specificities for the three proteomics datasets. For instance, under a '*rbf*' kernel, the PCA-SVM and SVM classification for the *Ovarian* cancer dataset can only classify the positive (cancer) targets. Both of them have an average classification rate of 64.13%, which is approximately the ratio of the positive targets among the total samples: 162/253=64.03%. 2. There is no overfitting problem under a '*rbf*' kernel, for the NPCA-SVM algorithm with sparse coding. On the other hand, the NPCA-SVM has the best classification performance among all the algorithms for the three protein expression datasets. 3. Under a linear kernel, the PCA-SVM achieves slightly better or comparable results than the SVM for the two ovarian datasets. Similarly, the SVM classification also has slightly better average classification rates, sensitivities and specificities than the PCA-SVM for the *Liver* dataset. Thus, we can say that their classification performances for the experimental datasets are comparable. 4. The classification results of the NPCA-SVM have leading advantages for the three datasets, compared with those of the PCA-SVM and SVM classifications. Actually, the average specificities for the two ovarian cancer datasets reach *99%+* under the NPCA-SVM

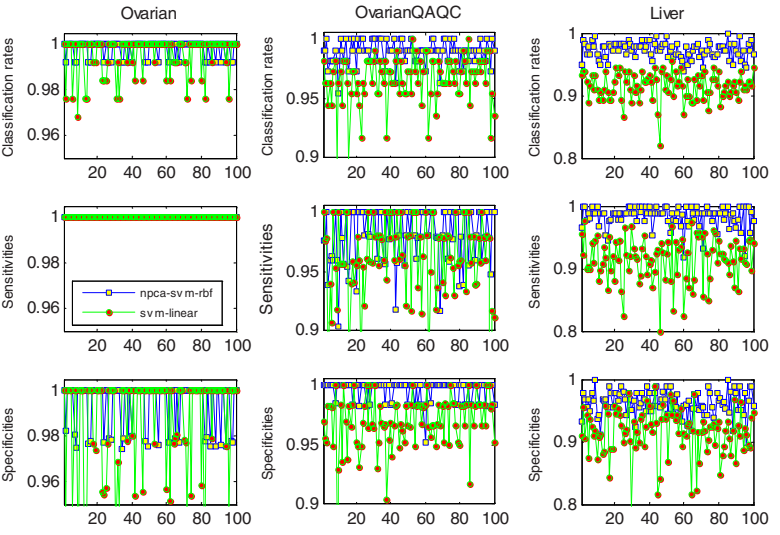


Fig. 1. Comparison on the SVM classification under a linear kernel and the NPCA-SVM classification under a '*rbf*' kernel. For the first dataset, NPCA-SVM ('rbf') has slightly better classification performance than the SVM ('linear'). For the 2nd and 3rd datasets, the NPCA-SVM ('rbf') classification has obvious leading advantages over the SVM ('linear') classification.

classification. The 99%+ specificity level is the population screening requirement ratio in general clinical diagnostics. Figure 1 shows the performances of the SVM algorithm under a linear kernel and the NPCA-SVM algorithm under a '*rbf*' kernel for three datasets.

3.3 Compare Classification Results with Those of Other Algorithms

We also compare the classification performance of the NPCA-SVM algorithm with those of three other classification algorithms: k-NN, PCA-LDA and NMF-SVM. For each dataset, we still use the previous 100 trials of training/test datasets generated under the 50% holdout cross validations in the classifications.

The k-NN and PCA-LDA are widely used algorithms in proteomics data classifications. The k-NN is a simple Bayesian inference method. It determines the class type of a sample based on the class belonging to its nearest neighbors, which are measured by correlation, Euclidean or other distances. The PCA-LDA conducts the PCA processing for the training samples and projects the test samples in the subspace spanned by the principal components of the training data. Then, linear discriminant analysis (LDA) is used to classify the projections of the test data, which is equivalent to solving a generalized eigenvalue problem [10].

The NMF-SVM algorithm is similar to the NPCA-SVM classification algorithm. It conducts the SVM classification for the meta-samples of a proteomics dataset, which are the columns of the feature matrix H in the NMF. We briefly describe the

NMF-SVM algorithm as follows. The NMF-SVM classification decomposes the nonnegative protein expression data $X \in \Re^{n \times m}$ into the product of two nonnegative matrices: $X \sim WH$, under a rank r with the least reconstruction error in the Euclidean sense. The matrix $W \in \Re^{n \times r}$ is termed a basis matrix. Its column space sets up a new coordinate system for X. The matrix $H \in \Re^{r \times m}$ is called a feature matrix. It stores the new coordinate values for each variable of X in the new space. Then, the SVM algorithm is employed to classify the corresponding meta-sample of each sample in the protein expression matrix X**.** Each meta-sample is just the corresponding column in the feature matrix H.

In the k-NN, the distance measures are chosen as the correlation and Euclidean distances. The number of nearest neighbors for each test sample is selected from *2* to *7*; In the NMF-SVM, the matrix decomposition rank in the NMF is selected from *2* to *18* for each dataset under the linear and Gaussian kernel. The final average classification rate for each dataset under the k-NN and NMF-SVM is selected as the best average classification rate of the 100 trials of training and test data among all cases. Table 3 shows the expected classification rates, sensitivities and specificities of the three algorithms and corresponding standard deviations for each of the three datasets, under the 100 trials of training/test datasets generated from 50% holdout cross validations.

Actually, we have found the k-NN algorithm achieves better classification performances under the correlation distance than the Euclidean distance. The NMF-SVM algorithm achieves better classification performances under the correlation distance than the Euclidean distance. For the three protein expression datasets, the NMF-SVM and k-NN classification results are comparable. However, it is obvious that the PCA-LDA algorithm achieves the best performances among the three algorithms.

Table 3. Average classification performances of the NMF-SVM, k-NN, PCA-LDA algorithms

	Average Sensitivity	Average Specificity	Average Classifying rates
Ovarian			
nmf-svm-linear	99.91±0.31	92.92±2.50	97.41±0.94
nmf-svm-rbf	96.27±3.35	90.83±4.48	94.29±2.72
knn-correlation	99.28±1.34	91.67±3.67	96.53±1.57
knn-euclidean	99.58±0.76	90.77±3.19	96.41±1.29
pca-lda	99.93±0.38	99.21±2.00	99.67±0.87
Ovarian-qaqc			
nmf-svm-linear	92.02±5.01	86.24±5.67	88.69±3.47
nmf-svm-rbf	76.18±9.12	78.57±6.38	77.30±3.67
knn-correlation	89.99±4.68	91.82±4.43	90.87±2.92
knn-euclidean	82.03±6.86	87.71±5.86	85.03±3.71
pca-lda	98.81±1.68	96.99±0.03	97.69±0.65
Liver			
nmf-svm-linear	84.58±5.14	71.30±5.12	77.76±2.48
nmf-svm-rbf	80.69±6.01	69.21±5.57	74.79±2.25
knn-correlation	72.27±4.60	80.80±4.57	76.48±2.20
knn-euclidean	77.04±5.81	75.38±5.33	76.11±2.51
pca-lda	91.39±5.81	88.87±3.95	90.08±2.13

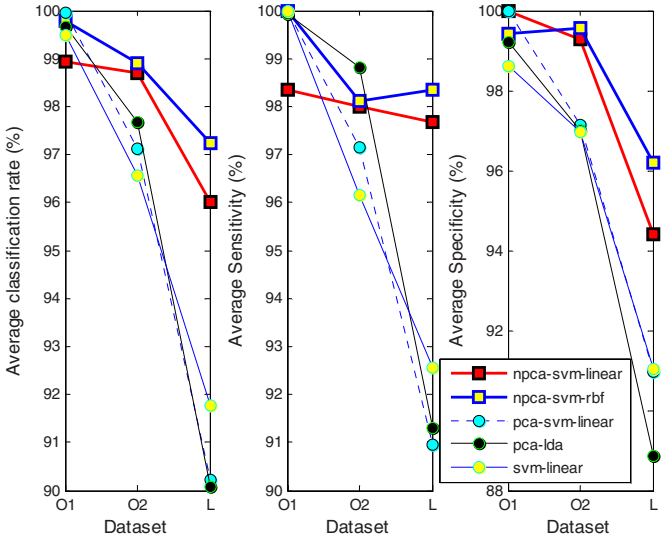


Fig. 2. Comparison on the classification performances of four algorithms for three proteomics datasets: 'O1' (*Ovarian*), 'O2' (*Ovarian-qaqc*), 'L' (*Liver*). The NPCA-SVM has the best performances among the four algorithms with respect to average classification rates, sensitivities and specificities for all three datasets, though the PCA-LDA and SVM algorithms both achieve comparable classification performances for the first ovarian dataset.

Figure 2 compares the classification performances of the NPCA-SVM algorithm with sparse coding to those of the PCA-LDA, PCA-SVM and SVM with respect to average classification rates, sensitivities and specificities. For all three proteomics datasets, it is obvious that the NPCA-SVM algorithm with sparse coding under the '*rbf*' and '*linear*' kernel has generally achieved the best or second-best classification results among all these algorithms respectively. The NPCA-SVM algorithm with sparse coding also gives the same leading results under the leave-one-out cross validation (*LOOCV)* according to our experimental results.

4 Conclusion and Discussions

In this study, we develop a novel feature selection algorithm: nonnegative principal component analysis (NPCA) and propose the NPCA-SVM algorithm under sparse coding for the cancer molecular pattern discovery of protein expression data. We also demonstrate the superiority of this novel algorithm over the NMF/PCA-SVM, SVM, k-NN and PCA-LDA classification algorithms for three benchmark proteomics datasets. Our algorithm also overcomes the overfitting problem of the SVM and PCA-SVM classifications under a Gaussian kernel.

With nonnegative principal component analysis, we can develop a family of NPCA-based statistical learning algorithms by applying NPCA as a feature selection algorithm before a classification or clustering algorithm, e.g., NPCA-based Fisher

discriminant analysis (NPCA-FDA), NPCA-based K-means or hierarchical clustering. In future work, we plan to investigate the NPCA-based classifications, such as NPCA-FDA, NPCA-SVM in the protein folding, gene, microRNA profiles data classification and biomarker discovery.

References

1. Han, X.: Cancer molecular pattern discovery by subspace consensus kernel classification, Computational Systems Bioinformatics, In: Proceedings of the Conference CSB 2007, vol. 6, pp. 55–65 (2007)
2. Hauskrecht, H., et al.: Feature Selection for Classification of SELDI-TOF-MS Proteomic Profiles. Applied Bioinformatics 4(4), 227–246 (2005)
3. Zou, H., Hastie, T., Tibshirani, R.: Sparse principal component analysis. Journal of Computational and Graphical Statistics 15(2), 262–286 (2006)
4. D'Aspremont, A., Ghaout, L., Jordan, M., Lanckriet, G.: A direct formulation for sparse PCA using Semidefinite Programming. SIAM Review 49(3), 434–448 (2007)
5. Zass, R. and Shashua, A.: Nonnegative sparse PCA, Neural Information and Processing Systems (NIPS) (2006)
6. Lee, D.D., Sebastian Seung, H.: Learning the parts of objects by non-negative matrix factorization. Nature 401, 788–791 (1999)
7. Hoyer, P.O.: Hoyer: Non-negativematrix factorization with sparseness constraints. Journal of Machine Learning Research 5, 1457–1469 (2004)
8. National Center Institute Center for Cancer Research Clinical Proteomics Program, `http://home.ccr.cancer.gov/ncifdaproteomics/ppatterns.asp`
9. Ressom, H., Varghese, R., Saha, D., Orvisky, R., et al.: Analysis of mass spectral serum profiles for biomarker selection. Bioinformatics 21(21), 4039–4045 (2005)
10. Lilien, R., Farid, H.: Probabilistic Disease Classification of Expression-dependent Proteomic Data from Mass Spectrometry of Human Serum. Journal of Computational Biology 10(6), 925–946 (2003)

Appendix: Nonnegative Principal Component Analysis Parameter Derivations

In this section, we give the detailed parameter derivation for equations (5) and (6). Computing the stationary points for the objective function $f(u_{sl})$ in equation (4), we have a cubic root finding problem. The final U matrix consists of a set of nonnegative roots of equation (9).

$$p(u_{sl}) = df(u_{sl}) / du_{sl} = -4\alpha u_{sl}^3 + 2c_2 u_{sl} + c_1 = 0 \tag{9}$$

We derive coefficients c_2, c_1 in equation (9) as follows. For convenience, we rewrite the terms in equation (2) as $\left\| I - U^T U \right\|_F^2 = L_1 + L_2$, where L_1 and L_2 represent the contributions from the diagonal elements and non-diagonal elements of the

matrix $I-U^TU$ to its Frobenius norm respectively, where $L_1=\sum_{l=1}^{k}(1-u_l^Tu_l)^2$ and $L_2=\sum_{l=1}^{k}\sum_{j=1,j\neq l}^{k}(u_l^Tu_j)^2$. Similarly, we also set $L_3=\left\|U^TX\right\|_F^2=\sum_{l=1}^{k}\sum_{j=1}^{n}(u_l^Tx_j)^2$ and compute the parameters c_2, c_1 by checking the coefficients of u_{sl} and u_{sl}^2 in equation (2). From the equation, we have following results:

$$L_1=k-2\sum_{l=1}^{k}\sum_{s=1}^{d}u_{sl}^2+\sum_{l=1}^{k}\sum_{s=1}^{d}u_{sl}^4+2\sum_{l=1}^{k}\sum_{s=1}^{d}\sum_{t=1,t\neq s}^{d}u_{sl}^2u_{tl}^2 \tag{10}$$

$$L_2=\sum_{s=1}^{d}\sum_{l=1}^{k}\sum_{j=1,l\neq j}^{k}u_{sl}^2u_{sj}^2+2\sum_{s=1}^{d}\sum_{l=1}^{k}\sum_{j=1,l\neq j}^{k}\sum_{t=1,t\neq s}^{d}u_{sl}u_{sj}u_{tl}u_{tj} \tag{11}$$

$$L_3=\sum_{l=1}^{k}\sum_{i=1}^{n}\sum_{s=1}^{d}u_{sl}^2x_{si}^2+2\sum_{l=1}^{k}\sum_{i=1}^{n}\sum_{s=1}^{d}\sum_{t=1,t\neq s}^{d}u_{sl}x_{si}u_{tl}x_{ti} \tag{12}$$

By substituting for the coefficients of u_{sl} and u_{sl}^2, we have

$$c_2=\frac{1}{2}\sum_{i=1}^{n}x_{si}^2-\alpha\sum_{j=1,j\neq l}^{k}u_{sj}^2-2\alpha\sum_{t=1,t\neq s}^{d}u_{tl}^2+2\alpha \tag{13}$$

$$c_1=\sum_{i=1}^{n}\sum_{t=1,t\neq s}^{d}x_{si}u_{tl}x_{ti}-2\alpha\sum_{j=1,j\neq l}^{k}\sum_{t=1,t\neq s}^{d}u_{sj}u_{tl}u_{tj} \tag{14}$$

Gene Ontology Assisted Exploratory Microarray Clustering and Its Application to Cancer

Geoff Macintyre[1,2], James Bailey[1,2], Daniel Gustafsson[4], Alex Boussioutas[3], Izhak Haviv[5,6], and Adam Kowalczyk[2]

[1] Department of Computer Science and Software Engineering, University of Melbourne, Victoria, Australia
[2] National ICT Australia, Victorian Research Lab, Australia
[3] Ian Potter Centre for Cancer Genomics and Predictive Medicine, Peter MacCallum Cancer Centre, St. Andrew's Place, East Melbourne, Victoria, Australia
[4] Department of Computer Science and Computer Engineering, La Trobe University, Victoria, Australia
[5] The Alfred Medical Research and Education Precinct, Baker Medical Research Institute, Epigenetics Group, Melbourne, Australia
[6] Department of Biochemistry and Molecular Biology, University of Melbourne, Victoria, Australia

Abstract. Gene expression profiling provides insight into the functions of genes at a molecular level. Clustering of gene expression profiles can facilitate the identification of the underlying driving biological program causing genes' co-expression. Standard clustering methods, grouping genes based on similar expression values, fail to capture weak expression correlations potentially causing genes in the same biological process to be grouped separately. We have developed a novel clustering algorithm which incorporates functional gene information from the Gene Ontology into the clustering process, resulting in more biologically meaningfull clusters. We have validated our method using a multi-cancer microarray dataset. In addition, we show the potential of such methods for the exploration of cancer etiology.

Keywords: Microarray, Gene Ontology, Clustering, Cancer.

1 Introduction

Gene expression profiling using microarrays has become a key tool in the analysis of biological systems at a molecular level. While still producing relatively noisy data, much improvement has been made in noise correcting normalisation procedures and feature selection, providing rich datasets for further biological analysis. Microarray analysis pipelines generally come in two flavours: differential expression analysis and exploratory clustering. The purpose of differential expression analysis is to find the set of genes which are differentially expressed between two or more experimental conditions or samples. Once the list of genes has been determined, it can be used to classify further microarrays into similar sample categories. Alternatively the differentially expressed genes can be

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 400–411, 2008.

analysed to try to unravel the underlying biology responsible for observed expression patterns. This is similar in approach to exploratory clustering. The aim in exploratory clustering is to try to uncover groups of genes with similar expression patterns. This is useful under the assumption that genes with shared expression patterns have similar function or are involved in similar biological processes. Each of the clusters of genes identified provide a starting point for further biological analysis based on gene expression.

While exploratory clustering has been shown to be successful in many cases, it can suffer from some common problems. Clusters can be dominated by strong or noisy expression patterns, forcing genes of similar function or those belonging to the same process with less correlated expression, to join another cluster. Therefore the resulting clusters may not represent a biological process in its entirety or majority, making it hard to determine which molecular processes a particular cluster of genes represents.

To improve the clustering process, additional information can be introduced to ensure genes with similar function or shared pathways can be clustered together. Sequence similarity, protein structure similarity, shared pathways and functions, are all ways in which genes can be shown to be related. There exist tools that use this information in trying to unravel the biology behind the observed expression behavior.

The Gene Ontology (GO) [1] is a curated, structured vocabulary that describes genes and gene products. This provides a source for finding shared molecular functions, biological processes or cellular components between two genes. In the GO, two genes may be annotated to the same biological term, or they may be related through a shared term higher in the GO hierarchy (see Fig. 1). From this information a similarity metric [2,3] can be defined which measures the relatedness of each gene via semantic functionality. This similarity measure can then be used as a biological prior probability measure on the clustering of genes via expression profiles. Previous attempts have been made to utilize the GO in clustering of gene expression profiles. Cheng et al [4] developed a clique-finding algorithm for the GO and used the cliques to perform co-clustering analysis with gene expression profiles. A biclustering approach which yields clusters designed to map onto the GO structure was developed by Liu et al[5]. Huang et al [6] and Pan [7] used GO annotations shared between genes to modify standard distance and model based clustering algorithms, and Boratyn et al [8] proposed a general method modifying the distance measure based on prior shared functional information between genes.

There are however two fundamental drawbacks with these approaches. Firstly, the GO is constructed as a directed acyclic graph, with terms lower in the tree being specialisations, or parts of, terms higher in the tree. Genes are then annotated to one or more terms in the tree, at the lowest (most specific) level possible. Drawing a path from one gene to another through this tree to determine similarity of the genes does not necessarily imply shared biology. The abstraction of terms across each level of the ontology can be such that two genes with a single shared parent term, may be extremely diverse in terms of their specific function.

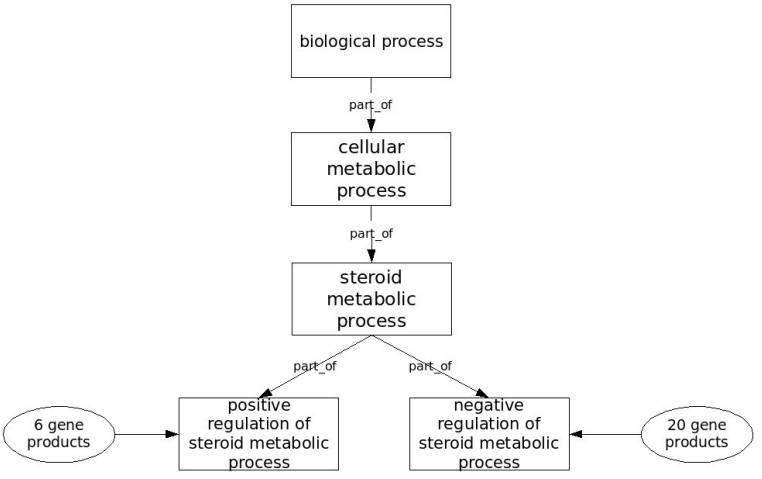


Fig. 1. This is a cut down example GO hierarchy for illustrative purposes. In the full Gene Ontology there would be additional terms between each of the nodes in the graph and the gene products would be annotated to more specialised terms lower in the tree.

For example, the two terms *negative regulation of steroid metabolic process* and *positive regulation of steroid metabolic process* share the parent *steroid metabolic process*. Genes annotated to each of these terms have the opposite effect on steroid metabolism. Therefore it would not be correct to state they had similar function based on their shared parent, especially in the context of their co-expression. Secondly, having genes annotated to the same term does not necessarily imply they have similar function or share a biological pathway, in the context of their expression patterns. A single gene can act differently in various biological contexts and thus have context specific roles.

The GO is also used by tools such as GeneMerge [9], FatiGO [10] and others [11,12,13,14] to determine over-represented GO terms given a group of genes, thus giving a semantic representation of the biology spanning a group of genes. In the context of microarrays, these tools provide the ability to explore clusters of genes and form biological hypotheses about the observed co-expression. One of the benefits of this approach is that one is not scrutinizing the behavior of a single gene, but rather groups of genes in the same biological context. This provides an abstracted level of analysis, which encapsulates a single gene's behavior, within the complex biological system represented by the cluster.

A method is needed which looks for commonalities between genes based on the GO that does not traverse the GO hierarchy and is relevant to the gene set of interest (the gene's biological context). We have developed GOMAC: **G**ene **O**ntology assisted **M**icro**a**rray **C**lustering, a modified k-means clustering algorithm which incorporates GO information only when it is relevant to the gene's context, thus avoiding problems with irrelevant gene similarities. We have validated our method on a microarray dataset [15] spanning 12 cancer types

demonstrating that our method results in an increase in number and biological relevance of meaningful clusters. We also discuss the biological implications of our results with respect to future research in cancer etiology.

2 Methods

The key biological assumption of the algorithm presented in this paper is that genes that share a particular annotation in the GO, will share a detectable similarity in their microarray expression pattern. There are two key differences between our approach and the previous attempts at clustering using the GO outlined above.

- Only GO terms that are statistically over-represented within a cluster are used to calculate the similarity between genes. This ensures that only GO terms within the gene's context are used.
- We iteratively calculate similarities between genes using the GO, rather than having GO similarities as a set prior.

In order to construct a model capable of the key points outlined above, each potential cluster of genes to be determined, requires both an expression profile to model the genes' expression measurements and an annotation profile to model the genes' GO similarities. As we are using a k-means based clustering algorithm, the number of clusters C is a parameter set by the user.

2.1 Algorithm Overview

1. Initialise using k-means clustering, grouping genes based on expression values using the method in Eisen et al [16] with C clusters.
2. Determine the *expression profile* for each cluster.
3. Determine the *annotation profile* for each cluster.
4. Re-cluster genes based on *both* expression and GO annotations.
5. Re-estimate the expression and annotation profiles.
6. Repeat steps 4 and 5 until convergence.

2.2 Expression Profile

Let κ be the number of samples; let G_c be the set of genes in a cluster c. Each gene g can be viewed as a vector $\mathbf{x}_g = (x_{gi})_{1 \leq i \leq \kappa} \in R^\kappa$ of its expression values across all samples. The centroid of cluster c is defined as the vector $\mathbf{X}_c = (x_1^c ... x_\kappa^c) \in R^\kappa$ with entries defined as:

$$x_i^c = \frac{\sum_{g \in G_c} x_{gi}}{|G_c|}. \tag{1}$$

where x_{gi} is the expression measurement for a particular gene g and sample i.

2.3 Gene Ontology Annotation Profile

To generate a Gene Ontology annotation profile for a cluster, all GO terms annotated to the genes in a cluster which are statistically over-represented need to be found. This means that rather than reporting all terms that are annotated to the genes in a cluster, report only those that have sufficiently low probability of being present if we sampled a random selection of genes. For this purpose we are using a program called GeneMerge1.2 [9]. This uses the hypergeometric distribution with Bonferroni correction to generate a p-value for each term which is annotated to genes in a cluster. We use a threshold of $b \leq 0.2$ of the Bonferroni corrected score as it provides a biologically meaningful number of terms that describe a cluster. A lower threshold yields clusters based mainly on expression distances with little or no GO terms and a higher threshold results in many GO terms which are less descriptive. All terms reported above the threshold are ignored. Let τ^c be the number of terms *below the threshold* b for a given cluster c. From this, a weight d is assigned proportional to the number of genes in the cluster that are annotated to that term, normalised over all of a cluster's GO terms. The weight d_t shows the degree in which a term t is associated with a particular cluster. Then we can denote a cluster c's annotation profile to be the vector $\mathbf{T}^c = (d_t)_{1 \leq t \leq \tau^c}$ with entries defined as

$$d_t = \frac{n_t}{\sum_{j=1}^{\tau^c} n_j}. \tag{2}$$

where n_t is number of genes in the cluster that are annotated to GO term t (below the threshold b).

2.4 Algorithm

Input

- Gene list G
- For each gene g, expression measurements $\mathrm{E}_{g1}...\mathrm{E}_{g\kappa}$ for κ samples
- For all GO terms $\mathrm{A}_1...\mathrm{A}_f$, given a particular gene g and the t^{th} term, A_{gt} takes the value true if the gene g is directly annotated to the term t, (obtained by querying the September 2007 release of the GO).

Initialisation

- Form initial groupings of genes using k-means clustering on the expression values.
- Calculate the cluster centroid (*expression profile*) $\mathbf{X}_c$ for each cluster.
- Calculate the *annotation profile* $\mathbf{T}^c$ for each cluster.

Optimisation

1. Gene assignment: In the gene assignment step, we re-assign a gene to a cluster based on the current values for the expression and annotation profiles for that

cluster. We use a gene's match to a cluster annotation profile to scale the Euclidean expression distance of the gene from that cluster.

For each gene g let the known expression values be $E_{g\beta}$ where $\beta \in N_g \subset \{1....\kappa\}$ are all indices of samples with known values for gene g. This is due to imperfections in the microarray experimental procedure which may generate data with missing or unknown expression values for a gene. Given this, we define the Euclidean distance of each gene g from cluster c's centroid as:

$$DE_g^c = \frac{1}{|N_g|} \cdot \sqrt{\sum_{\beta \in N_g} (x_\beta^c - E_{g\beta})^2}. \tag{3}$$

Then, given a gene g and its GO annotations, we also determine a scaling factor S_g^c (where $0 \leq S_g^c \leq 1$). This is based on how many of the τ^c terms in the cluster's annotation profile match the terms annotated to a gene g:

$$S_g^c = 1 - \sum_{t=1}^{\tau^c} (d_t^c \cdot A_{gt}). \tag{4}$$

Next, the expression distance DE of gene g from cluster c is scaled by the degree in which it's annotated terms correlates with that of cluster c:

$$DES_g^c = DE_g^c \times S_g^c. \tag{5}$$

Finally, we simply use the minimum of this modified distance to assign a gene to a particular cluster:

$$c_g = \arg\min_c (DES_g^c). \tag{6}$$

2. Re-estimation of cluster profiles: With the new assignment of genes, we re-calculate the centroids of each cluster and determine the new GO terms which are over-represented and their associated weights.
3. Repeat steps 1 and 2 until convergence (genes stop changing clusters)

Output

– A series of gene clusters with associated GO annotations, which can be used as a starting point for further biological analysis.

3 Clustering Performance Assessment

External clustering assessment typically uses a 'gold standard' clustering determined by external means, to compare clusterings to. However, in the case of exploratory clustering, there is no 'gold standard'. Instead, a standard measure to determine whether a new algorithm provides biologically better clusters than a previous algorithm, is to look for statistically over-represented GO terms in each of the clusters and show that the new algorithm has clusters of superior biological relevancy. However because we used the GO in the clustering process, this measure is not suitable. Two alternative means of external validation were devised.

3.1 Histological Enrichment Criterion

A measure was generated to try to test the biological significance of the clusters output by GOMAC with respect to their usefullness in biological hypothesis generation, in particular, uncovering differences in cancer histologies. This measure is based on the ability of the algorithm to uncover clusters which help answer or generate a specific biological question. In the case of the multi-class cancer dataset used, this is the ability to uncover clusters which show well defined differential expression across various cancer types. The idea here is that if the algorithm works successfully, the genes in a cluster should encapsulate a particular biology.

As cancer types can be similar or vary significantly depending on their location in the body, one would expect certain cancers to have similar biological expression behaviour and others to differ. To measure this, additional hierarchical clustering of the samples of each cluster was performed to partition the cancers into two groups. If successfull, one group contains all cancers that have predominantly upregulated genes in the cluster, and the other group has all samples which have predominantly downregulated expression. In the dataset used, the samples were sub-divided by label, into classes (cancer types). Given a good clustering, the upregulated partition of a cluster should be saturated by all of a particular sub-class of sample (cancer type). That is, we would expect cancers sharing some (perhaps unknown) biology to be grouped together. To quantify this, considering a single partition, the hypergeometric distribution was used to determine the probability of observing a particular enrichment of sample classes (cancer types) by chance. From this, a Bonferroni corrected p-value was generated which was used to determine the quality of a particular cluster in reference to its biological usefullness.

3.2 Functional Annotation Enrichment Criterion

In addition, an alternative information source to the GO was used to determine biological significance of a cluster. Each of the clusters were analysed through the use of Ingenuity Pathway Analysis (Ingenuity®Systems, www.ingenuity.com). The genes were overlayed with function and disease information provided by IPA. The overlay procedure takes a gene list as input and outputs the functions and diseases annotated to the genes that are over-represented, similar in process to GeneMerge mentioned above. Each of the terms has an associated p-value and all relationships have been extracted from various literature sources curated by experts. The IPA results were used as a comparison to assess the accuracy of the GO annotation profiles reported for each cluster by GOMAC.

4 Cancer cDNA Microarray Test Data

For testing, a robust microarray dataset with various sample classes was chosen to demonstrate the potential of GOMAC to uncover biological similarities across

classes. We applied GOMAC to a published collection of cDNA microarray data [15] representing 12 cancer types and their subtypes. The full dataset was filtered retaining only genes with greater than 400 signal intensity in the test channel (Cy5) and greater than 4 fold change (using per gene median normalised data) in at least 5 samples. These cut-offs were suggested by the original authors in order to minimise experimental noise and maximise the differential gene expression across samples. This left 2185 genes and 165 samples: Breast(23), Colorectal(12), Gastric(7), Lung(Adenocarcinoma 10, Large Cell Carcinoma 8, Squamous Cell Carcinoma 9), Melanoma(11), Mesothelioma(5), Ovarian(21, Mucinous 11), Pancreatic(8), Prostate(5), Renal(12), Squamous Cell Carcinoma (SCC) (11), Testicular(3), Uterine(9). The test set was clustered using regular k-means and GOMAC with the number of clusters C=10,20,30,40,50,100,200 and 300. The value C=40 provided the best granularity of GO terms for discussion and results presented in this paper are exclusively for C=40. An added bonus of this dataset is that novel biological hypotheses could be generated in terms of shared behaviour between cancers through using GOMAC.

5 Results

The dataset was clustered using the k-means clustering algorithm of Eisen et al [16] and with GOMAC. Additional hierarchical clustering was performed on the samples within each cluster using the Bioconductor Package Heatmap.2 [17]. The resulting dendrogram was cut so that the samples were partitioned into two groups. Table 1 contains the significance scores representing how well the partitioning of samples into two groups splits the sample sub-classes, without dividing the sub-classes themselves (histological enrichment criterion).

Table 1 shows that GOMAC provided 3 additional biologically significant clusters over regular k-means clustering and improved the biological significance of approximately 55% of the partionable clusters. In addition, GOMAC resulted in a greater number of GO descriptors (terms) for the clusters.

Ingenuity Pathway Analysis (IPA) was carried out on each cluster of genes to provide an external source of semantic validation (functional annotation criterion). Significantly reported functions and diseases for each cluster from IPA were grouped with the GO annotation profile of the cluster and the samples enriched by partitioning to see if there was any correlation. Table 2 gives a summary of the diseases and functions reported by IPA associated with a selection of the significant clusters.

6 Discussion and Future Work

Cancers from multiple sites in the body have been expression profiled in single datasets in the past. This was mainly done in order to design tools to identify the site of origin of Cancer of Unknown Primary [18,19,15]. In addition, meta analysis of multiple cancers was used to identify common themes in cancer gene expression [20]. The Gene Ontology is usually used only after the clustering of

Table 1. The significance score reported in each instance is calculated on one of the partitions of samples. Only significance scores generated from the same partition sample size for the same cluster can be compared and are indicated with a *. N\A values were cases where the dendrogram could not be cut to partition the samples into two sizeable groups. In this case the cluster is likely to be representative of noisy data or ubiquitously expressed genes and is therefore uninteresting in this context. Clusters omitted from this table also fall into this category. The number of terms significantly over-represented in the cluster are listed as an indication of the descriptive power of the clusters.

Cluster	***k*-means partition significance**	**/GO terms**	**GOMAC partition significance**	**/GO terms**	**Cluster**	***k*-means partition significance**	**/GO terms**	**GOMAC partition significance**	**/GO terms**
1*	3.6E-27	2	4.7E-26	2	**20***	3.6E-07	8	4.7E-06	8
2*	2.8E-08	0	5.7E-05	1	**21***	1.4E-15	0	**1.3E-18**	1
4	1.4E-08	1	N\A	0	**22**	1.0E-14	1	N\A	0
5	3.9E-12	6	4.5E-16	6	**24**	2.6E-20	2	1.8E-17	2
10*	2.4E-01	10	**1.9E-02**	10	**27**	1.9E-10	1	N\A	0
11	N\A	3	**8.9E-05**	5	**33***	1.4E-10	5	6.0E-10	5
12	N\A	0	**1.3E-13**	0	**36***	3.6E-07	0	**4.6E-09**	1
13*	4.9E-11	0	**1.9E-11**	0	**37**	N\A	7	**3.6E-10**	8
14	N\A	0	**3.8E-18**	0	**38**	N\A	0	**2.9E-09**	2
15	8.8E-20	0	1.6E-14	0	**39***	4.6E-09	2	**3.6E-10**	2
16	5.2E-16	0	N\A	0	**40**	N\A	0	**3.3E-07**	1
19	N\A	0	**1.6E-02**	1					

genes and samples has been done. Here we reasoned that since multiple genes are co-ordinately expressed by means of biological programs, such as cell types and organs, the use of the GO in the process of clustering would focus the analysis on the driving program rather than individual genes.

Cumulative evidence during the last 50 years argues that cancer progression arises through accumulation of somatic changes in the cancer cell that confer selective advantage to the mutant cell in terms of extension and unlicensed extended lifespan. As these selection pressures are different in different organs of the body, one expects that some of the somatic changes would be organ specific. This point would be missed in classical expression profiles as different cancer types are not profiled on the same platform in the same instance. Therefore by using the cancer profiles of multiple samples in combination with GO clustering, we are positioned for the first time to address this possibility.

While Table 1 demonstrates that both regular *k*-means and GOMAC can reveal gene clusters which can be partitioned to demonstate biology specific to certain cancer types, the improved significance and increased number of significant clusters provided by GOMAC suggests that clustering using the GO is advantageous. The greater number of GO terms uncovered by GOMAC also gives increased power in biological interpretation of the clusters. Therefore our method improves the ability to uncover biological processes that are specific to

Table 2. Three examples of the significant clusters identified by GOMAC are shown. The cancers identified are those that were partitioned into one group for that cluster. The genes belonging to the cluster are defined by the GO terms. The terms identified by the independent IPA analysis with p-value less than 0.001 are also reported.

Cluster	GO terms	Cancers	IPA Function/Disease
1	digestion steroid metabolism	Gastric Colorectal Muc. Ovarian Pancreatic	metabolism of steroid transcription of GATA site digestive organ tumor pancreatic tumor colon cancer
5	spleen development urogenital system development embryonic organ development prostate gland development sex differentiation fatty acid metabolism	Breast Prostate	guidance of motor axons size of prostate gland breast cancer cancer of mammalia mammary tumor genital tumor
38	keratinization epidermis development	Lung-SCC Mesothelioma SCC	ichthyosis development of epidermis adhesion of cells differentiation of keratinocytes binding of stromal cells

certain cancer types as each of the resulting clusters have been both reduced in noise and made more biologically informative.

It is immediately obvious from the IPA overlay in Table 2 that genes associated with cancer processes have been identified. Moreover, many of the significant IPA terms actually match the cancers saturating the cluster. There are also examples of the over-represented GO terms correlating with the IPA over-represented terms. This success in linking cancer types, with gene expression profiles and a statistically significant semantic description of the underlying biology, provides an excellent starting point for the exploration of the similarities and dissimilarities of various types of cancers. In fact, the GO terms identified for cluster 1 in Table 2, *digestion* and *steroid metabolism*, have clinical observations linked with the cancers significantly partitioned. The cancers in this cluster are *predominantly* derived from gastrointestinal and ovarian epithelium. The mucinous production of both of these cancers may be linked through steroid metabolism, as outlined by this cluster. Interestingly, *all* of the tumours in this dataset may be derived from the gastrointestinal system. There are mucinous tumours of the ovary that have been found to be metastatic deposits from Gastric cancer primaries which have been termed Krukenberg tumours [21]. Furthermore, Krukenberg tumours have been linked to virilisation and hence another link with steroid metabolism. While this cluster was identified by standard k-means, it was also identified by GOMAC. This suggests that as well as GOMAC being able to identify a greater number of informative clusters than k-means (for example cluster 38 has a major shared similarity of squamous differentiation across the cancers in the group, with Mesothelioma being an exception), it can also retain

initial informative clusters. Therefore, coupled with a similar analysis of normal tissues, these results can potentially uncover a combination of tissue specific and cancer effects which have not been identified before.

The gene ontology tool however, suffers from a substantial flaw in its potential to assist the deciphering of genome language; the genes are curated according to whether they belong to a pathway, without discriminating between antagonistic to agonistic genes. Consequently, in our analysis, samples could appear to belong in the same group, while in fact they divide into two groups, based on whether the pathway is induced or repressed. This problem may be resolved by utilizing more advanced gene curation algorithms, such as those collected in Gene Set Enrichment Analysis [22], which represent genes that have been observed in microarray experiments as those responding to defined molecular changes, such as activating mutations or forced expression of defined genes. Interestingly, even before using such algorithms, integrating the GO into the clustering process improves the segregation of samples, and, in addition, it sheds new light on the biological processes that drive specific cancer types, and organ specific biological programs.

Summary. We have shown through our analysis, that incorporation of additional biological information into the microarray clustering process in a biologically justified manner, can enhance the interpretability of microarray data. Specifically, we have shown the potential of such a method to unravel the complex nature of the biological processes involved in cancer. Ideally, our method would be repeated multiple times, while alternating the source of the ontology, the cancer types, and genes. Followed by ranking of the segregating lists according to significance, then formation of an integrated summary list, that records all possible drivers of the biological systematic variations among cancers in different organs. A key benefit of such an exercise would be hypothesis generation, in the field of cancer etiology with an organ specific focus.

Acknowledgements. Many thanks to Dr Richard Tothill for providing the microarray data and to Justin Bedo for help preparing the manuscript. This work is partially supported by NICTA. NICTA is funded by the Australian Government through the Department of Broadband, Communications and the Digital Economy and the Australian Research Council.

References

1. Ashburner, M., et al.: Gene ontology: tool for the unification of biology. Nat. Genet. 25, 25–29 (2000)
2. Schlicker, A., Domingues, F., Rahnenfuhrer, J., Lengauer, T.: A new measure for functional similarity of gene products based on gene ontology. BMC Bioinf. 7 (2006)
3. Zhang, P., Zhang, J., Sheng, H., Russo, J., Osborne, B., Buetow, K.: Gene functional similarity search tool (gfsst). BMC Bioinf. 7, 135 (2006)
4. Cheng, J., Cline, M., Martin, J., Finkelstein, D., Awad, T., Kulp, D., Siani-Rose, M.A.: A knowledge-based clustering algorithm driven by gene ontology. J. Biopharm. Stat. 14, 687–700 (2004)

5. Liu, J., Wang, W., Yang, J.: Gene ontology friendly biclustering of expression profiles. In: Proceedings of CSB 2004, pp. 436–447. IEEE, Los Alamitos (2004)
6. Huang, D., Pan, W.: Incorporating biological knowledge into distance-based clustering analysis of microarray gene expression data. Bioinf. 22, 1259–1268 (2006)
7. Pan, W.: Incorporating gene functions as priors in model-based clustering of microarray gene expression data. Bioinf. 22, 795–801 (2006)
8. Boratyn, G.M., Datta, S., Datta, S.: Incorporation of biological knowledge into distance for clustering genes. Bioinformation 1 (2007)
9. Castillo-Davis, C.I., Hartl, D.L.: Genemergepost-genomic analysis, data mining, and hypothesis testing. Bioinf. 19, 891–892 (2003)
10. Al-Shahrour, F., et al.: Fatigo: a web tool for finding significant associations of gene ontology terms with groups of genes. Bioinf. 20, 578–580 (2004)
11. Martin, D., Brun, C., Remy, E., Mouren, P., Thieffry, D., Jacq, B.: Gotoolbox: functional analysis of gene datasets based on gene ontology. Genome biology 5 (2004)
12. Lee, S.G., Hur, J.U., Kim, Y.S.: A graph-theoretic modeling on go space for biological interpretation of gene clusters. Bioinf. 20, 381–388 (2004)
13. Alexa, A., Rahnenfuhrer, J., Lengauer, T.: Improved scoring of functional groups from gene expression data by decorrelating go graph structure. Bioinf. 22, 1600–1607 (2006)
14. Zhong, S., Tian, L., Li, C., Storch, K.F., Wong, W.: Comparative analysis of gene sets in the gene ontology space under the multiple hypothesis testing framework. In: Proceedings of CSB 2004, pp. 425–435. IEEE, Los Alamitos (2004)
15. Tothill, R.W., et al.: An expression-based site of origin diagnostic method designed for clinical application to cancer of unknown origin. Cancer Res. 65, 4031–4040 (2005)
16. 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)
17. Gentleman, R., et al.: Bioconductor: open software development for computational biology and bioinformatics. Genome Biology 5 (2004)
18. Su, A.I., et al.: Molecular classification of human carcinomas by use of gene expression signatures. Cancer Res. 61, 7388–7393 (2001)
19. Ramaswamy, S., et al.: Multiclass cancer diagnosis using tumor gene expression signatures. PNAS 98, 15149–15154 (2001)
20. Segal, E., Friedman, N., Koller, D., Regev, A.: A module map showing conditional activity of expression modules in cancer. Nat. Genet. 36, 1090–1098 (2004)
21. Joshi, V.V.: Primary krukenberg tumor of ovary. review of literature and case report. Cancer 22, 1199–1207 (1968)
22. Subramanian, A., et al.: Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. PNAS 102, 15545–15550 (2005)

Discovery of Biomarkers for Hexachlorobenzene Toxicity Using Population Based Methods on Gene Expression Data

Cem Meydan[1], Alper Küçükural[2], Deniz Yörükoğlu[3], and O. Uğur Sezerman[4]

Biological Sciences and Bioengineering, Sabanci University
Sabancı Üniversitesi, Orhanlı-Tuzla, 34956 İstanbul, Türkiye
Tel.: + (90)2164839000; Fax: + (90)2164839550
{cemmeydan,kucukural,denizy}@su.sabanciuniv.edu,
ugur@sabanciuniv.edu

Abstract. Discovering toxicity biomarkers is important in drug discovery to safely evaluate possible toxic effects of a substance in early phases. We tried evolutionary classification methods for selecting the important classifier genes in hexachlorobenzene toxicity using microarray data. Using modified genetic algorithms for selection of minimum number of features for classification of gene expression data, we discovered a number of gene sets of size 4 that were able to discriminate between the control and the hexachlorobenzene (HCB) exposed group of Brown-Norway rats with >99% accuracy in 5-fold cross-validation tests, whereas classification using all of the genes with SVM and other methods yielded results that vary between 48.48% to 81.81%. Making use of this small number of genes as biomarkers may allow us to detect toxicity of substances with mechanisms of toxicity similar to HCB in a fast and cost efficient manner when there are no emerging symptoms.

Keywords: Feature selection, toxicogenomics, genetic algorithms, biomarker discovery.

1 Introduction

Finding reliable toxicity biomarkers is important in toxicogenomics to safely evaluate possible toxic effects of a substance in early phases of drug discovery. Discovering the important mechanisms of toxicity for known toxic substances and developing biomarkers that detect these can lead to classification of new substances with respect to their toxicity in a cost-efficient manner.

Using microarray technology to evaluate the changes in gene expression data between control and experiment data sets, the significant set of genes that indicate the existence of the toxicity may be obtained. Discovering these genes that are correlated with the substance class may point the mechanisms of toxicity and the effected pathways. These sets of genes may also be used for development of diagnostic kits that can be used to detect possible existence of toxicity of a substance on the test subjects, or on the early diagnosis of toxin exposure.

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 412–423, 2008.

Minimizing the number of genes that are used as a biomarker, without affecting the accuracy of the prediction, is essential for practical purposes. Each redundant control will have a negative effect time-, complexity- and cost-wise, therefore finding the minimal set of genes with highest classification accuracy is in practical interest. Feature subset selection refers to this problem of selecting important set of attributes from a large set of redundant attributes that are uncorrelated with the class used in classification purposes.

The method proposed by Kucukural et al. successfully selects the minimum number of features for classification of gene expression data using evolutionary methods with low attribute counts and very high accuracy in cancer data sets compared to other methods [1]. The viability of this method on toxicity data sets was unexplored, and if any changes were necessary for adapting it to toxicogenomics. To compare its performance, we tried this evolutionary classification method and other classifiers for selecting the minimum number of important set of genes in hexachlorobenzene toxicity using microarray data.

Hexachlorobenzene (HCB) is an organochlorine fungicide with persistent environmental pollution effects, and has various toxic mechanisms in man. During the period 1955-1959, about 4000 people in southeast Anatolia in Turkey developed porphyria due to the ingestion of HCB that were used on wheat seedlings [2]. HCB has been also classified as a Group 2B carcinogen (possibly carcinogenic to humans) by the International Agency for Research on Cancer (IARC). Animal carcinogenicity data for hexachlorobenzene show increased incidences of liver, kidney (renal tubular tumours) and thyroid cancers [3].

Although HCB usage was banned in most areas, it is still generated as waste by-products of industrial processes. The pollution of the sea coasts and groundwater persists due to its stability, and HCB is still detectable in human milk and blood in some areas of the world.

In this study HCB data set is used because its effects, mechanisms of toxicity, the pathways affected and the toxicology data such as oral and inhalation dosage in mice, rats and humans are well documented, therefore the obtained set of classifier genes may be compared with the literature.

2 Related Work

Studies for finding genomic markers for detection of changes in an organism are aimed at different reasons. One is mainly for practical diagnostic purposes, and other is for discovering the underlying mechanism in that change. Although both can be used for other purposes as well, the goal in finding diagnostic markers is to minimize the number of needed data without affecting accuracy.

If the toxin causes a response in gene expression level, microarray technology is very powerful for biomarker discovery [4-5]. The entire human genome can be contained on a single microchip, enabling us to generate complete profile of the response to toxicity [6-8]. Representational difference analysis (RDA) of tissue- or cell-specific arrays are used to find candidate biomarker proteins from protein-coding genes that have a specific change in expression, when control and experimental values are compared [9-10]. However, using only genomic data is insufficient, since it only measure changes in mRNA

expression, but abundant quantities of mRNA does not necessarily equal abundant quantities of protein [5], thus semi-quantitative assays are required for checking the proteins. An example to this is the study from Ichimura et al. who identified the gene with most significant change in the postischemic rat kidney, KIM-1, and confirmed its viability for use as a biomarker by subsequent immunoblot, immunostaining, and RNA in situ hybridization [9]. Examples to other similar studies in which analysis of microarray data is used to find markers and changed gene expression levels for damage due to radiation toxicity [11], hemolytic anemia induced by drugs [12], nephrotoxicity induced by cisplatin [13], identification of glutathione depletion-responsive genes in rat liver [14], and many more. This approach is powerful, and pharmaceutical and biotechnology companies even created panels of biomarkers that detect drugs causing hepatotoxicity upon in vitro exposure to rat hepatocytes or in vivo dosing in rats [15].

For markers of toxicity, studies are mostly in mechanistic field. The work by Ezendam et al. [16], whose data set we studied, tried to find the significant changes in gene expression due to hexachlorobenzene exposure, and found a total of 104 genes in different tissues that are affected by the HCB. Similar works concentrating on changes in expression levels in specific tissues or on the whole organism due to exposure to several chemicals are also numerous.

Other works in which the biomarker discovery by feature selection is studied are also present. Many studies concentrate on heuristic search and randomized population based techniques such as genetic algorithms. Early studies used known computational procedures such as greedy optimization, branch and bound, tabu search, simulated annealing, gibbs sampling, evolutionary programming, genetic algorithms, ant colony optimization and particle swarm optimization [17-24]. These perform differently in different conditions due to the heuristic methods; no best solution can be found due to the practically non-computable number of possible solutions, making exhaustive search impossible.

Recently, feature selection by coupling genetic algorithms with statistical classifiers has been studied. Alon et al. used clustering algorithms to find genes with correlated expression levels that can be used for diagnosis. They found a set of genes that can classify colon cancer by 90% accuracy [25]. On the same data set, Fröhlich et al. used genetic algorithms coupled with support vector machines to find minimum number of genes that can classify the data, which resulted in a set of 30 genes with 85% accuracy [26]. The work by Kucukural et al. that we used in this study concentrates on dynamic parent generation by fitness score of features using genetic algorithms, and was very efficient in finding a low number of highly accurate solutions, resulting in 98% accuracy using 12 genes in the same colon cancer data set and 100% accuracy with 12 genes in ovarian cancer data set [1].

3 Methodology

To discover the minimum number of features that can classify the data, we have to find a way represent these sets of genes. In the genetic algorithm, each individual in the population represents a candidate solution to the feature subset selection problem. The genetic code of a parent is boolean vector of size m, where m is the number of attributes. A value of 1 means the parent has that attribute, and 0, not. Since there are 2^m possible parents, exhaustive search is impossible with more than a handful of attributes, thus evolutionary algorithms or other heuristic methods are necessary.

The method is based on the selfish gene idea by Richard Dawkins, in which the individuals are only the carriers of genes, and the function of a parent is to leave the strong genes to next generation [27]. Thus, the main goal is the survival of the gene. If an individual has a good fitness score with fewer genes, the gene (i.e. feature) count can be decreased. Basically, the algorithm uses this concept to select features. Details of the genetic algorithm are given below.

The main base of the genetic algorithm uses standard mutation and cross-over algorithms, using support vector machine (SVM) learning to assign a fitness score. SVM is a very accurate supervised learning method, widely used in computational problems in biology. Score of an individual is calculated by the accuracy of classification by SVM in 5-fold cross-validation tests.

The genetic algorithm employs elitism in which low scoring children are replaced with best scoring parents, if the score of the parents are higher. Also a small number of "bad" parents are also selected, which keeps the algorithm from being stuck in local minima, thus acting as simulated annealing along with the roulette wheel method.

Used SVM parameters are given below. LibSVM library [28-29] was used in both the genetic algorithm and in the following tests.

In the first generation, parents are randomly generated with each having a set number of features and each feature being covered a set amount, for reducing the chance of a good attribute being dropped due to redundant neighbours in that parent. Then, for a number of generations (given as non-reducing generations below) the genetic algorithm runs without trying to reduce the number of features. In this phase each feature is assigned the fitness score of its parent. After the first run for a set number of generations, average fitness score for each attribute is obtained by dividing the total fitness score by the number of times that feature was chosen in an individual.

After this first pass is completed, the reducing phase of the genetic algorithm with roulette wheel based selection strategy is used in which the number of features is reduced for filtering the redundant attributes. In roulette wheel the probability of selecting a feature is its fitness score, therefore high scoring attributes are selected more often. Each child carries a specific gene set generated by the roulette wheel selection, cross-over and mutation steps, and when a gene is selected more than once for a specific set, the duplicates will be removed, consequently decreasing the total feature count. This way the number of features of a child decreases if the same attribute is selected more than once. This parent generation scheme, which focuses on "gene" instead of parent, allows the dynamic selection of optimal number of features. The effectiveness of this approach can be clearly seen in Figures 1 and 2.

The genetic algorithm uses the following parameters;

Number of Features: 8799
Feature Coverage (the number each feature is covered in the first generation): 2
Number of features in each parent in 1st generation: 30
Number of non-reducing generations: 50
Number of reducing generations (see above): 500
Population Size: 587 ((# of features)x(feature coverage) / (# of features in each parent))
Crossover Rate: 0.9, Mutation Rate: 0.1
Elite Parent Rate: 0.2, Bad Elitist Rate: 0.01

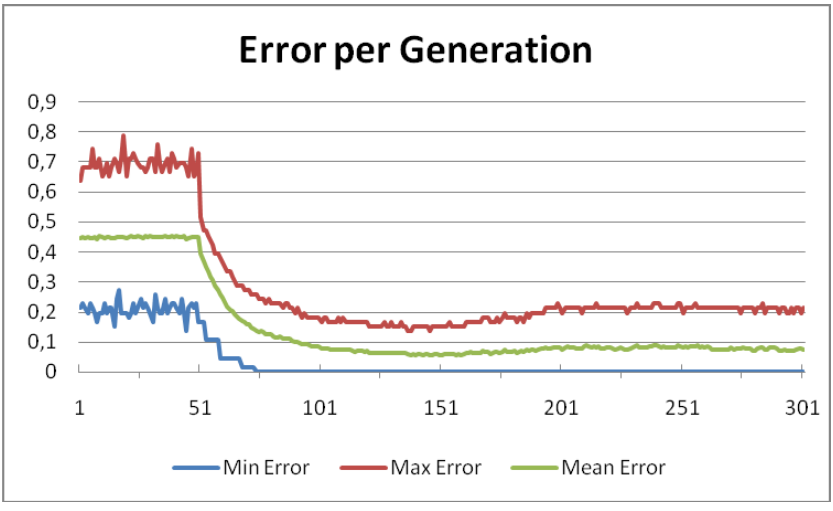


Fig. 1. Min/Max/Mean errors in run of the algorithm with respect to generation in set 1. The given errors are the proportion of the false positives and false negatives in the whole test set. Notice that after 50 generations, algorithm changes its selection strategy from non-reducing to reducing (see text), and the min. error rate quickly converges to 0.

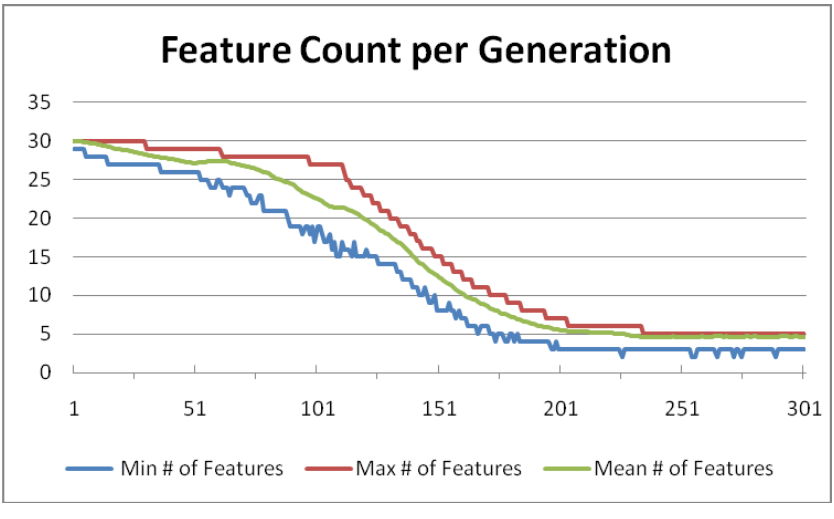


Fig. 2. Number of features in the population with respect to generation. While the error rate becomes 0 at about 75 (see Figure 1 above), the feature count gradually decreases until it reaches a minimum of 4 in about 200th generation. Although there are some selected ones with 3 features, they are not able to classify with 100% accuracy and are dropped against the sets with 4 features.

SVM Parameters: C_SVC with kernel Radial Basis Function
C: 100, Gamma: 1.0 / 8799
Using normalization and shrinking. Not using probability estimates.

The algorithm was repeated for 10 iterations for each 0-150, 0-450, and 150-450mg/kg comparison (see Sections 4.1 and 4.2). Each run takes approximately 15 minutes on a standard PC with a 2GHz CPU. After the genes are selected iteratively, different SVM parameters are used to find the optimal SVM score. By optimization of the other parameters more accurate solutions can be found [28], however the results obtained were ~100%, thus no optimizations were necessary.

4 Results

4.1 Data Set

The microarray data used is from the study Ezendam et al. in Dutch National Institute for Public Health [16]. Ezendam et al. fed Brown Norway rats with diets supplemented with 0, 150 and 450mg/kg HCB for 4 weeks, after which spleen, mesenteric lymph nodes (MLN), thymus, blood, liver, and kidney were collected and analyzed using the Affymetrix rat RGU-34A GeneChip microarray. Using 1 microchip per tissue per animal, they obtained a total of 96 hybridizations; 35 from untreated control group, 30 from 150mg/kg and 31 from 450mg/kg exposed group, each having 8799 genes of RGU-34A chip.

The data set is available on EBI ArrayExpress [30] (accession: E-TOXM-15).

4.2 Experiments

HCB has relatively low acute toxicity, but it is cumulative in lipid tissues, and is persistent. According to Extension toxicology network data, oral toxic dosage of HCB is 10,000mg/kg [31] in rats. Rats exposed to 450mg/kg per day for 28 days accumulate about 12600mg/kg of HCB, thus showing the most toxic effects [31-33]. According to the previous works that studied the posology of HCB, the time of onset, severity and size of skin lesions, the increase in body, liver and spleen; in other words the toxic effects of HCB with respect to exposure, ideal dose of exposure in which pathological problems are seen in rats is 450mg/kg, so comparison between 0 and 450mg/kg is studied in most detail to find the markers of sub-chronic exposure. One rat in 450mg/kg exposed group even died after 25 days of exposure [16], thus we expect the changes in gene expression levels to differ more from the 0-150mg/kg comparison. However, for early prediction, the low-dose markers and studies for 0-150mg/kg and also the classification of dosage by 150-450mg/kg are also given, though not as detailed.

The found marker genes are then analysed and classified with other methods in the data mining environments WEKA (Waikato Environment for Knowledge Analysis from University of Waikato) [34] and Gist SVM package [35]. The genes are also searched in Affymetrix NetAffx Analysis Center [36] and in Kyoto Encyclopedia of Genes and Genomes (KEGG) [37-39] for correlation between chip result and gene functions and pathways.

Table 1. Accuracy of the classification with different methods using both unfiltered 8799 features and the selected 4 features (set 3), in cases of 5-fold CV and testing on training data. Parameters are default values if not stated, except in SVM, which uses the parameters shown in section 3. N/A values were not classified due to high memory requirements. Post-selection accuracy values showing increase are indicated in bold/green, showing decrease in italic/red.

Method	Accuracy			
	Using all 8799 Genes in 0-450mg/kg		Using the Selected 4 Genes	
	5-fold cross-validation	Test on training data	5-fold cross-validation	Test on training data
SVM	53,03%	100%	**100%**	100%
Decision Table	71,21%	96,97%	**77,27%**	*86,36%*
Naive Bayesian	65,15%	100%	**77,27%**	*78,78%*
Naive Bayes Multinomial	54,55%	100%	**93,94%**	*93,94%*
Multi-layer Perceptron	N/A	N/A	100%	100%
RBF Network	43,94%	100%	**92,42%**	100%
Simple Logistics	62,12%	100%	**95,45%**	100%
Random Forest	68,18%	100%	**90,90%**	100%
NBTree	75,76%	100%	**78,79%**	*95,45%*
C4.5 Decision Tree	66,67%	96,97%	**93,94%**	96,97%
k-nearest neighbour	81,81%	100%	**96,97%**	100%
K-Star	N/A	N/A	**95,45%**	100%
Classification via clustering (N=2)	48,48%	50%	*45,45%*	**53,03%**

4.3 Results

We discovered a number of gene sets of size 4 that were able to discriminate between the control and the 450mg/kg HCB exposed group of Brown-Norway rats with >99% accuracy by SVM classification in 5-fold cross-validation tests, whereas classification using all of the genes with the same methods such as SVM, Naïve Bayesian, C4.5 decision tree, RandomForest yielded results that vary between 48,48% to 81.81% (Table 1). Similarly, in 0-150mg/kg 7 genes, and in 150-450mg/kg 6 genes that gave 100% accuracy were discovered. Since the changes in gene expressions should be more subtle due to exposure, it is normal for them to have more features to classify correctly.

For 0-450mg/kg comparison, 8 of the iterations gave 4, 1 iteration gave 3 and 1 gave 5 genes, for an average of 4 genes per set. The distributions were similar for others, with few less than and few more than 6 and 7.

As seen in Table 1, since these genes are discriminative, any classifier can be used, not only SVM. Using multi-layer perceptron classification on selected feature set also gave 100%, C4.5 gave 93.94%, 1-nearest neighbour clustering gave 96.97% accuracy. Thus, without using SVM, even simple classifiers such as decision trees may be used for accurate classification by hand.

An important finding is that, while filtering features (and thus decreasing the information content) mostly reduced the classification accuracy using all of the training data for testing, at the same time it increased the accuracy of 5-fold cross-validation dramatically. We can conclude that the 100% accuracy in testing by training data is due to overfitting, and when the redundant features (i.e. noise) are filtered, the classifiers actually work much better in real world conditions. Reducing the feature count not only decreases the test cost and time/memory requirements, but also increases the accuracy.

In 10 iterations, about 40 to 70 attributes are selected in total for each run. The selection of attributes shows half-normal distribution; a small number of genes appear in most of the sets while most of them only appear in one. It is possible that by running the algorithm for more iterations and selecting the most common elements may give more robust solutions for use in real world.

Table 2. Details of the genes in feature set of 3rd iteration and cross selection of these genes in other sets

Accession Number	Gene Symbol	Description	# of times selected		
			0-150	0-450	150-450
D00913_g_at	Icam1	intercellular adhesion molecule 1	1	8	0
AF093139_s_at	Nxf1	nuclear RNA export factor 1 (mRNA_processing_Reactome)	0	3	0
rc_AA892154_g_at	Mxd4_predicted	Max dimerization protein 4 (predicted)	0	3	0
rc_AA892325_at	Cept1	choline/ethanolamine phosphotransferase 1	0	2	0

However, while intra-class gene counts are half-normally distributed, inter-class gene similarity is very low. As it can be seen in Table 2, in the genes selected for 0-450 separation are selected, just 1 of them is selected once in 0-150 and 150-450 classification. The results from other sets are similar. Although these occurrences are more frequent than randomly selecting the same genes from a pool of 8799 genes, they are still somewhat lower considering the selected genes are effected by HCB exposure.

Kucukural et al. compared the results on colon and ovarian cancer with other studies in terms of number of genes and accuracy. However, there are no studies in HCB toxicity focusing on minimum number of predictive genes. Study by Ezendam et al. focused on microarray data for important genes in various tissues that have role or affected by HCB toxicity, and although feature selection was done for selecting significant ($p < 0.001$) expression level changes, the number was not minimized. Nevertheless, 45 changed genes in spleen, 16 genes in MLN, 7 genes in thymus, 27 in blood and 19 in liver were detected. Of those, M63122, D00913, K00996, AA891209 and E00778 are also present in the genes we selected.

To compare the number of features given by the algorithms, we used other feature selection methods such as Sequential Floating Forward Selection (SFFS) and SVM-based Recursive Feature Elimination (RFE-SVM, or SVM-RFE), which are widely used for biomarker selection in the literature due to their good results. We also added some standard feature selection methods. Of those, feature selection using the Gain Ratio evaluation, Information Gain, SVM weight evaluation and Correlation-based feature subset selection (CFS-Subset) are all done in the WEKA data mining software [34]. RFE-SVM is evaluated in the Gist software [35], and SFFS was our implementation.

Table 3. Comparison of different feature selection methods on the data set. The method proposed by Kucukural et al. clearly surpasses other methods; it has the best accuracy among the tested methods with only 4 features. Even though others have more features (except SFFS), they still fail to reach 100% accuracy. In RFE-SVM, Gain ratio, Information gain and SVM weight methods, the top n elements are filtered by score.

Methods	Feature Count	ROC Area	Accuracy	Specificity	Sensitivity
Our Method	4	1.0000	1.0000	1.0000	1.0000
RFE-SVM, n=4	4	0.8000	0.7727	0.7097	0.8286
RFE-SVM, n=8	8	0.9152	0.8182	0.8387	0.8000
RFE-SVM, n=17	17	0.9926	0.9697	0.9677	0.9714
Gain Ratio, n=5	5	0.8590	0.8636	0.7742	0.9429
Gain Ratio, n=10	10	0.9070	0.9091	0.8710	0.9429
Information Gain, n=10	10	0.8930	0.8939	0.8710	0.9143
SVM Weight, n=10	10	0.9700	0.9697	0.9677	0.9714
CFS-Subset	66	0.9370	0.9394	0.9032	0.9714
SFFS	2	0.8190	0.8182	0.8387	0.8000

The 5-fold cross validation results are given in table 3. These comparisons show that the method proposed by Kucukural et al. surpasses others in both feature count and accuracy. The only other method that had lower feature count was SFFS with 2 features, however its accuracy was only 81%, compared to 100% with 4 features on our method.

5 Conclusions and Discussion

The dynamic parent generation with emphasis on feature suitability for classification performs much better than the standard genetic algorithm in which the chance of good features coming together is dependent on crossover probabilities. In the standard approach heuristics are used only to decrease the search space, while this method tries to select the optimal features. Along with parents (i.e. sets of features), features themselves get a score and these features are then put into survival. With a higher fitness score, a feature gets selected more often while others are not selected at all and eliminated, and better features have a chance to get selected more than once in an individual, thus decreasing the number of features in the parent.

Unlike statistical tests which examine the genes that have the most significant change in expression levels but lacks classification power, this algorithm finds genes with not critical change but with high degree of separation. For example Lipocalin 2 (extension AA946503), has shown about 50-folds change with $p < 0.001$ [16], but is not usable for classification with great accuracy (due to the overlaps), and was not selected by our algorithm, although it is definitely in the pathways affected by HCB.

Although no prior work on biomarker detection for HCB induced toxicity was done, we compared the method with other well known feature selection and classifier algorithms, in which the selected gene count and accuracy was better in our case. To compare our accuracy, we classified the data without any feature selection. Classifying the data using all of the genes, without any selection, results in lower accuracy. Thus, low

number of genes used as biomarkers favour both accuracy and financial/computational costs of the tests.

With this study, we concluded that biomarkers of toxicity can be discovered with the method improved upon the work proposed by Kucukural et al, and possibly with even better results than cancer in this case of HCB. This method resulted in only 4 features out of 8799 that can classify HCB exposure in Brown Norway rats with 100% accuracy, which is higher than the results obtained in colon cancer (12 features, 98.38% accuracy) and ovarian cancer (12 features, 100% accuracy) [1]. Comparison of the selected genes with the literature [16, 36-39] showed that most of these genes are known for their functions in the pathological effects of toxicity induced by HCB. Using the low dose classifiers and other markers obtained by further study, toxin exposure can be detected when there are no emerging symptoms, which can be used in both medicine and experimental drug discovery. Studying only the changes in blood cells can lead to unintrusive markers that can be used detect the toxicity or disease from only blood samples.

Another point of use for these markers is that, by gathering data from various toxic studies and finding reliable biomarkers of toxicity pathways for different mechanisms of toxicity (e.g. in immunotoxicity; immunosuppression, immunostimulation, hypersensitivity reactions, autoimmune reactions, etc.) can allow us to generate toxicity assays that are more efficient and accurate than the ones used today, which will allow detection of toxicity in very early stages of drug discovery, thus saving much time, effort and money.

References

1. Küçükural, A., Yeniterzi, R., Yeniterzi, S., Sezerman, O.U.: Evolutionary selection of minimum number of features for classification of gene expression data using genetic algorithms. In: 9th Annual ACM Conference on Genetic Evolutionary Computation, pp. 401–406. ACM Press, New York (2007)
2. Gocmen, A., Peters, H.A., Cripps, D.J., Bryan, G.T., Morris, C.R.: Hexachlorobenzene episode in Turkey. Biomed Environ. Sci. 2(1), 36–43 (1989)
3. International Agency for Research on Cancer, IARC Monographs on the Evaluation of Carcinogenic Risk to Humans, World Health Organisation, vol.79, 493–567 (2001)
4. Collings, F.B., Vaidya, V.S.: Novel technologies for the discovery and quantitation of biomarkers of toxicity. Toxicology (2008) doi:10.1016/j.tox.2007.11.020
5. Tugwood, J.D., Hollins, L.E., Cockerill, M.J.: Genomics and the search for novel biomarkers in toxicology. Biomarkers 8(2), 79–92 (2003)
6. Arcellana-Panlilio, M., Robbins, S.M.: Cutting-edge technology: I. Global gene expression profiling using DNA microarrays. Am.J. Physiol. Gastrointest. Liver Physiol. 282, 397–402 (2002)
7. Geschwind, D.H.: DNA microarrays: translation of the genome from laboratory to clinic. Lancet Neurol. 2, 275–282 (2003)
8. Ishkanian, S., Malloff, C.A., Watson, S.K., DeLeeuw, R.J., Chi, B., Coe, B.P., Snijders, A., Albertson, D.G., Pinkel, D., Marra, M.A., Ling, V., MacAulay, C., Lam, W.L.: A tiling resolution DNA microarray with complete coverage of the human genome. Nat.Genet. 36, 299–303 (2004)
9. Ichimura, T., Bonventre, J.V., Bailly, V., Wei, H., Hession, C.A., Cate, R.L., Sanicola, M.: Kidney injury molecule-1 (KIM-1), a putative epithelial cell adhesion molecule containing a novel immunoglobulin domain, is up-regulated in renal cells after injury. Biol. Chem. 273, 4135–4142 (1998)

10. Hubank, M., Schatz, D.G.: Identifying differences in mRNA expression by representational difference analysis of cDNA. Nucleic Acids Res. 22, 5640–5648 (1994)
11. Kruse, J., Stewart, F.A.: Gene expression arrays as a tool to unravel mechanisms of normal tissue radiation injury and prediction of response. World. J. Gastroenterol. 13(19), 2669–2674 (2007)
12. Rokushima, M., Omi, K., Imura, K., Araki, A., Furukawa, N., Itoh, F., Miyazaki, M., Yamamoto, J., Rokushima, M., Okada, M., Torii, M., Kato, I., Ishizaki, J.: Toxicogenomics of drug-induced hemolytic anemia by analyzing gene expression profiles in the spleen. Toxicol Sci. 100(1), 290–302 (2007)
13. Huang, Q., Dunn, 2.R.T., Jayadev, S., DiSorbo, O., Pack, F.D., Farr, S.B., Stoll, R.E., Blanchard, K.T.: Assessment of cisplatin-induced nephrotoxicity by microarray technology. Toxicol Sci. 63(2), 196–207 (2001)
14. Kiyosawa, N., Uehara, T., Gao, W., Omura, K., Hirode, M., Shimizu, T., Mizukawa, Y., Ono, A., Miyagishima, T., Nagao, T., Urushidani, T.: Identification of glutathione depletion-responsive genes using phorone-treated rat liver. J. Toxicol Sci. 32(5), 469–486 (2007)
15. Mendrick, D.L.: Genomic and Genetic Biomarkers of Toxicity. Toxicology (2007)doi:10.1016/j.tox.2007.11.013
16. Ezendam, J., Staedtler, F., Pennings, J., Vandebriel, R.J., Pieters, R., Boffetta, P., Harleman, J.H., Vos, J.G.: Toxicogenomics of subchronic hexachlorobenzene exposure in Brown Norway rats. Environ Health Perspect 112(7), 782–791 (2004)
17. Schölkopf, B., Smola, A.J.: Learning with Kernels. MIT Press, Cambridge (2002)
18. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern classification, 2nd edn. John Wiley and Sons, New York (2001)
19. Webb, A.R.: Statistical Pattern Recognition. John Wiley and Sons, New York (2002)
20. Vapnik, V.: Statistical Learning Theory. John Wiley and Sons, New York (1998)
21. Raymer, M., Punch, W., Goodman, E., Kuhn, L., Jain, A.: Dimensionality Reduction Using Genetic Algorithms. IEEE Transactions on Evolutionary computing (2000)
22. Ferri, F.J., Kadirkamanathan, V., Kittler, J.: Feature Subset Search using Genetic Algorithms. In: IEE/IEEE Workshop on Natural Algorithms in Signal Processing, Essex (1993)
23. Richeldi, M., Lanzi, P.: A Tool for Performing effective feature selection by investigating the deep structure of the data. In: Proc. of the International Conference on Tools with Artifcial Intelligence, pp. 102–105 (1996)
24. Witten, H., Frank, E.: Data Mining: Practical machine learning tools and techniques, 2nd edn. Morgan Kaufmann, San Francisco (2005)
25. Alon, U., Barkai, N., Notterman, D., Gish, K., Ybarra, S., Mack, D., Levine, A.: Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon cancer tissues probed by oligonucleotide arrays. Cell Biology 96, 6745–6750 (1999)
26. Fröhlich, H.: Feature Selection for Support Vector Machines by Means of Genetic Algorithms. Diploma Thesis in Computer Science, University Marburg (2002)
27. Dawkins, R.: The Selfish Gene – new edition. Oxford University Press, Oxford (1989)
28. Chang, C., Lin, C.: LIBSVM: a library for support vector machines, Software (2001), `http://www.csie.ntu.edu.tw/~cjlin/libsvm`
29. EL-Manzalawy, Y., Honavar, V.: WLSVM: Integrating LibSVM into Weka Environment, Software (2005), http://www.cs.iastate.edu/~yasser/wlsvm
30. Brazma, A., Parkinson, H., Sarkans, U., Shojatalab, M., Vilo, J., Abeygunawardena, N., Holloway, E., Kapushesky, M., Kemmeren, P., Lara, G.G., Oezcimen, A., Rocca-Serra, P., Sansone, S.A.: ArrayExpress–a public repository for microarray gene expression data at the EBI. Nucleic Acids Res. 1 31(1), 68–71 (2003)

31. Extoxnet. The Extension Toxicology Network, P.I.P. Pesticide Information Profiles: Hexachlorobenzene hexachlorobenzene-ext.html (1996a), http://pmep.cce.cornell.edu/profiles/extoxnet/haloxyfop-methylparathion/
32. Michielsen, C., Zeamari, S., Leusink-Muis, A., Vos, J., Bloksma, N.: The environmental pollutant hexachlorobenzene causes eosinophilic and granulomatous inflammation and in vitro airways hyperreactivity in the Brown Norway rat. Arch. Toxicol. 76, 236–247
33. Imai, N., Ichihara, T., Nabae, K., Hagiwara, A., Tamano, S., Shirai T.: Dose Dependent Promoting Effects of Hexachlorobenzene on Hepatocarcinogenesis in a Rat Medium-Term Liver Bioassay. In: Proc. of the 32nd Annual Meeting of Carcinogenicity
34. Garner, S.R.: The waikato environment for knowledge analysis. In: Proc. of the New Zealand Computer Science Research Students Conference, pp. 57–64 (1995)
35. Pavlidis, P., Wapinski, I., Noble, W.S.: Support vector machine classification on the web. Bioinformatics 1 20(4), 586–587 (2004)
36. Liu, G., Loraine, A.E., Shigeta, R., Cline, M., Cheng, J., Chervitz, S.A., Kulp, D., Siani-Rose, M.A.: NetAffx: affymetrix probeset annotations. In: 2002 ACM Symposium on Applied Computing, pp. 147–150. ACM Press, New York (2002)
37. Kanehisa, M., Araki, M., Goto, S., Hattori, M., Hirakawa, M., Itoh, M., Katayama, T., Kawashima, S., Okuda, S., Tokimatsu, T., Yamanishi, Y.: KEGG for linking genomes to life and the environment. Nucleic Acids Res. 36, 480–484 (2008)
38. 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, 354–357 (2006)
39. Kanehisa, M., Goto, S.: KEGG: Kyoto Encyclopedia of Genes and Genomes. Nucleic Acids Res. 28, 27–30 (2000)

Exploiting Fine-Grained Parallelism in the Phylogenetic Likelihood Function with MPI, Pthreads, and OpenMP: A Performance Study

Alexandros Stamatakis[1] and Michael Ott[2]

[1] The Exelixis Lab, Teaching and Research Unit Bioinformatics, Department of Computer Science, Ludwig-Maximilians-University Munich
stamatakis@bio.ifi.lmu.de, http://icwww.epfl.ch/~stamatak/
[2] Department of Computer Science, Technische Universität München
ottmi@in.tum.de, http://www.lrr.in.tum.de/~ottmi/

Abstract. Emerging multi- and many-core computer architectures pose new challenges with respect to efficient exploitation of parallelism. In addition, it is currently not clear which might be the most appropriate parallel programming paradigm to exploit such architectures, both from the efficiency as well as software engineering point of view. Beyond that, the application of high performance computing techniques and the use of supercomputers will be essential to deal with the explosive accumulation of sequence data. We address these issues via a thorough performance study by example of RAxML, which is a widely used Bioinformatics application for large-scale phylogenetic inference under the Maximum Likelihood criterion. We provide an overview over the respective parallelization strategies with MPI, Pthreads, and OpenMP and assess performance for these approaches on a large variety of parallel architectures. Results indicate that there is no universally best-suited paradigm with respect to efficiency and portability of the ML function. Therefore, we suggest that the ML function should be parallelized with MPI and Pthreads based on software engineering criteria as well as to enforce data locality.

1 Introduction

Emerging parallel multi- and many-core computer architectures pose new challenges not only for the field of Bioinformatics, since a large number of widely used applications will have to be ported to these systems. In addition, due to the continuous explosive accumulation of sequence data, which is driven by novel sequencing techniques such as, e.g., pyrosequencing [1], the application of high performance computing techniques will become crucial to the success of Bioinformatics. Applications will need to scale on common desktop systems with 2–8 cores for typical everyday analyses as well as on large supercomputer systems with hundreds our thousands of CPUs for analyses of challenging large–scale datasets. While many problems in Bioinformatics such as BLAST searches [2], statistical tests for host-parasite co-evolution [3], or computation of Bootstrap

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 424–435, 2008.

replicates [4] for phylogenetic trees are embarrassingly parallel [5], they might nonetheless, soon require the introduction of an additional layer of parallelism, i.e., hybrid [3,6] or multi–grain [7] parallelism to handle constantly growing dataset–sizes. Moreover, for large embarrassingly parallel problems, hybrid parallelizations can potentially allow for more efficient exploitation of current computer architectures by achieving super-linear speedups due to increased cache efficiency (see Section 4 and [8]). To this end, we focus on fine–grained loop–level parallelism, which is typically harder to explore than embarrassing parallelism. We study performance of MPI-, OpenMP-, and Pthreads-based loop–level parallelism by example of RAxML [9] which is a widely used program (2,400 downloads from distinct IPs; over 5,000 jobs submitted to the RAxML web-servers) for Maximum Likelihood-based (ML [10]) inference of phylogenetic trees. The program has been used to conduct some of the largest phylogenetic studies to date [11,12].

Apart from considerable previous experience with parallelizing RAxML and mapping the phylogenetic ML function to a vast variety of hardware architectures that range from Graphics Processing Units [13], over shared memory systems [8] and the IBM Cell [7], to the SGI Altix [14] and IBM BlueGene/L [15] supercomputers, RAxML exhibits properties that make it a well-suited candidate for the proposed study: *Firstly,* the communication to computation ratio can easily be controlled by using input alignments of different lengths; *secondly* the computation of the ML function requires irregular access of floating point vectors that are located in a tree; *thirdly* the parallelization strategies described here are generally applicable to *all* ML-based programs for phylogenetic inference, including Bayesian methods.

The current study represents the first comparison of MPI, Pthreads, and OpenMP for the phylogenetic ML function, which is among the most important statistical functions in Bioinformatics. It is important to note that, despite a more demanding development process, MPI naturally enforces data locality, which might significantly improve performance on NUMA architectures and ensures portability to systems such as the IBM BlueGene.

1.1 Related Work

A previous study on the comparison of OpenMP, MPI, and Pthreads [16] focused on performance for sparse integer codes with irregular remote memory accesses. Other recent papers [17,18] conduct a comparison of OpenMP versus MPI on a specific architecture, the IBM SP3 NH2, for a set of NAS benchmark applications (FT, CG, MG). The authors show that an OpenMP-based parallelization strategy, that takes into account data locality issues, i.e., requires a higher MPI-like programming effort, yields best performance. However, such an approach reduces portability of codes. In a parallelization of a code for analysis of Positron Emission Tomography images [19] the authors conclude that a hybrid MPI-OpenMP approach yields optimal performance.

Shan *et al.* [20] address scalability issues of a dynamic unstructured mesh adaptation algorithm using three alternative parallel programming paradigms

(MPI, SHMEM, CC-SAS) on shared and distributed memory architectures and report medium scalability for an MPI-based parallelization which however provides a high level of portability.

Our study covers a larger diversity of current architectures than the aforementioned papers, in particular with respect to multi-core systems, and assesses performance of three common programming paradigms for the ML function. To the best of our knowledge, this is the first comparative study, that provides a comparison of the three programming paradigms for loop-level parallelism on multi-core architectures, cluster, and SMP architectures.

2 General Fine–Grained Parallelization Scheme

The computation of the likelihood function consumes over 90-95% of total execution time in all current ML implementations (RAxML [9], IQPNNI [21], PHYML [22], GARLI [23], MrBayes [24]). Due to its intrinsic fine-grained parallelism, the ML function thus represents the natural candidate for parallelization at a low level of granularity. Though the ML method also exhibits a source of embarrassing parallelism at a significantly more coarse–grained level [15], in our study, we exclusively focus on fine–grained parallelism, which will become increasingly important for multi–gene analyses (see [11,25] for examples) or even larger whole–genome phylogenies that can have memory requirements exceeding 16–32GB.

To compute the likelihood of a fixed *unrooted* tree topology with given branch lengths, initially one needs to compute the entries for all internal likelihood vectors that essentially reflect the probabilities of observing an `A,C,G,` or `T` at an inner node for each site of the input alignment, bottom-up towards a virtual root that can be placed into any branch of the tree.

Note that, all computations of the partial likelihood vectors towards the virtual root can be conducted independently. As outlined in Figure 1 synchronization is only required before reduction operations that are conducted by the functions that compute the overall log likelihood score of the tree or optimize branch lengths (branch length optimization not shown in Figure 1). The computation of partial likelihood array entries consumes about 75% of total execution time. Once the partial likelihood arrays have been computed, the log likelihood value can then be calculated by essentially summing up over the likelihood vector values to the left and right of the virtual root. This means, that a reduction operation is required at this point.

In order to obtain the *Maximum* Likelihood value all individual branch lengths must be optimized with respect to the overall likelihood score. For a more detailed description please refer to [5] and [10]. Note that, most current search algorithms such as GARLI, RAxML, or PHYML, do not re-optimize *all* branch lengths and do not re-compute all partial likelihood arrays after a change in tree topology but rather carry out local optimizations as outlined in Figure 1 in the neighborhood of the tree region that is affected by the change. The main bulk of all of the above computations consists of `for`-loops over the length m of the multiple sequence input alignment, or more precisely over the number m' of

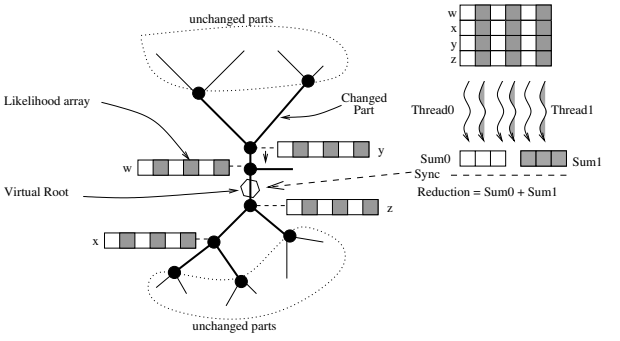


Fig. 1. Outline of the parallelization scheme used for Pthreads and MPI

distinct patterns in this alignment. The individual iterations of the `for`-loops of length m' in the functions that are used to calculate the phylogenetic likelihood function are independent and can therefore be computed in parallel, i.e., the maximum degree of parallelism is m'. This property is due to one of the fundamental assumptions of the ML model which states that individual alignment columns evolve independently [10].

3 Parallelization with OpenMP, Pthreads, and MPI

3.1 OpenMP

The parallelization of RAxML with OpenMP, is straight–forward, since only a few `pragma`'s have to be inserted into the code. Note that, in the current RAxML release (available as open-source code at `http://icwww.epfl.ch/~stamatak/`, version 7.0.4) we only parallelized the standard GTR model of nucleotide substitution [26] under the Γ model of rate heterogeneity [27]. The parallelization scheme is analogous to the concept presented in [8], however there is one fundamental difference: OpenMP might induce serious numerical problems, because the order of additions in reduction operations is non-deterministic. Given, e.g., four partial likelihood scores $l_0, ..., l_3$ from 4 threads $t_0, ..., t_3$ the order of additions to compute the overall likelihood is unspecified and can change during the inference. This behavior might cause two —otherwise exactly identical— mathematical operations to yield different likelihood scores. This has caused serious problems in RAxML with 4 threads on a large multi-gene alignment, i.e., it is not only a theoretical problem. To this end we modified the straight–forward OpenMP parallelization of the `for`-loops to enforce a guaranteed addition order for reduction operations.

While OpenMP clearly requires the lowest amount of programming overhead, it is less straight-forward to identify and resolve issues that require a higher degree of control over mechanisms such as thread affinity, memory locality, or reduction operation order.

3.2 Pthreads and MPI

The basic parallelization concept for Pthreads and MPI is analogous to the strategy for the BlueGene/L as outlined in [15]. While this parallelization mainly focused on proof–of–concept aspects and only implements the GTR+Γ model (see above) for a single version of the RAxML search algorithm, the parallelization presented here represents a complete re-implementation that covers the full functionality and plethora of models provided by RAxML (please consult the RAxML Manual for details [28]). The main goal of this re-engineering effort was to develop a single code that will scale well on multi-core architectures, shared memory machines, as well as massively parallel machines. Since the concepts devised for the Pthreads- and MPI-based parallelizations are conceptually similar we provide a joint description.

In a distributed memory scenario each of the p worker processes allocates a fraction m'/p of memory space (where m' is the number of unique columns in the alignment) required for the likelihood array data–structures which account for $\approx$ 90% of the overall memory footprint. Threads will just use an analogous portion of a global data structure. Thus the memory space and computational load for likelihood computations is equally distributed among the processes/threads and hence the CPUs. Moreover, the vector fractions m'/p are consistently enumerated in all processes, either locally (MPI) or globally (Pthreads).

The master thread/process orchestrates the distribution or assignment of data structures at start-up and steers the search algorithm as well as the computation of the likelihood scores. Thus, the master simply has to broadcast commands such as, e.g., compute likelihood array entries, given certain branch lengths, for vectors w, x, y, and z (see example in Figure 1) for the respective fraction m'/p and compute the likelihood score. In the Pthreads-based version the master thread also conducts an equally large part m'/p of the likelihood computations.

Global reduction operations, which in both cases (likelihood computation & branch length optimization) are simply an addition over m' double values, are performed via the respective MPI collective reduction operation while jobs are distributed with `MPI_Broadcast`. The Pthreads version is implemented accordingly, i.e., threads are generated only once at program start and then synchronized and coordinated via a master-thread. Job distribution and reduction operations in the Pthreads-based version are less straight-forward than with MPI, since Pthreads lack an efficient barrier method. Therefore, we implemented a dedicated function that uses a busy-wait strategy.

In contrast to the branch length optimization and likelihood computation operations, the computation of partial likelihood arrays frequently consists of a series of recursive calls, depending on how many vectors must be updated due to (local) changes in the tree topology or model parameters (see Figure 1). In order to reduce the communication frequency such series of recursive calls are transformed into a single iterative sequence of operations by the master. The master then sends the whole iterative sequence of inner likelihood vector updates that are stored by their vector numbers in an appropriate tree traversal data structure via a single broadcast to each worker or makes it available in

shared memory. Note that, in contrast to the "classic" fork-join paradigm used in OpenMP, this approach explicitly makes use of a dependency analysis of the algorithm and reduces the number of synchronization points.

An important change with respect to the previous version is the striped assignment of alignment columns and hence likelihood array structures to the individual threads of execution (see Figure 1). The rationale is that this allows for better and easier load distribution, especially for partitioned analyses of multi–gene datasets. Using a striped allocation every processor will have an approximately balanced portion of columns for each partition. Moreover, this also applies to mixed analyses of DNA and protein data, since the computation of the likelihood score for a single site under AA models is significantly more compute-intensive than for nucleotide data.

An important observation since the release of the Pthreads-based version in January 2008 is that the parallel code is used much more frequently than the previous OpenMP–based version since it compiles "out-of-the-box" on Unix/Linux and Macintosh platforms with `gcc` and allows for explicit specification of the number of threads via the command line. Such considerations are important for tools whose users are mainly non–experts. Development and maintenance experience over the last years has shown that potential users quickly abandon a tool if it requires installation of additional software and compilers. The programming effort to re-engineer RAxML and implement the Pthreads- as well as MPI-based parallelizations amounted to approximately 6 weeks.

4 Experimental Setup and Results

4.1 Test Datasets, Experimental Setup and Platforms

In order to test scalability of the three parallel versions of RAxML we extracted DNA datasets containing 50 taxa with 50,000 columns (d50_50000, 23,285 patterns) and 500 taxa with 5,000 columns (d500_5000, 3,829 patterns) from a 2,177 taxon 68 gene mammalian dataset [29]. In addition, we extracted DNA subsets with 50 taxa and 500,000 base-pairs (d50_500000, 216,025 patterns) as well as 250 taxa and 500,000 base-pairs (d250_500000, 403,581 patterns) from a large haplotype map alignment [14].

We used the Intel compiler suite version 10.1 for all three program versions on all platforms. Additionally, the platform-specific compiler optimizations flags used where identical for each of the three versions with only one exception: On the Altix interprocedural optimizations (IPO) caused a performance degradation by a factor of 3 if applied to the sequential version. Therefore we disabled these optimizations in that case. For all other compilations we enabled IPO as it slightly improved performance.

To measure the speedup we started RAxML tree searches under the GTR+Γ model on a fixed Maximum Parsimony starting tree (see [28] for details) on all platforms. Note that, we only report the best speedup values for every number of cores used on multi-core platforms with respect to the optimal thread to CPU assignment/mapping. Due to architectural issues, an execution on two cores

that are located on a single socket, can be much slower than an execution with two cores, located on two distinct sockets (see [30] for a more detailed study of thread-to-core mapping effects on performance). For instance we observed execution time differences of around 40% on the Intel Clovertown system for different assignments of two threads to the 8 cores of the system and over 50% for distinct mappings of four threads.

As test platforms we used a 2-way quad-core AMD Barcelona system (8 cores), a 2-way quad-core Intel Clovertown system (8 cores), an 8-way dual-core Sun x4600 system (16 cores) that is based on AMD Opteron processors. We measured execution times for sequential execution as well as parallel execution on 2, 4, 8, and 16 (applies only to x4600) cores. In addition, we used a cluster of 4-way SMP (4 single cores) 2.4 GHz AMD Opteron processors, that are interconnected via Infiniband, to test scalability of the Pthreads-, OpenMP-, and MPI-based versions up to 4 CPUs, and up to 128 CPUs (127 worker processes) for the MPI-based version. Finally, we used an SGI Altix 4700 system with a total of 9,728 Intel Itanium2 Montecito cores, an aggregated peak performance of 62.3 Teraflops, and 39 Terabyte of main memory (the HLRB2 supercomputer at the Leibniz Rechenzentrum, `http://www.lrz-muenchen.de/services/compute/hlrb`). On the SGI Altix we assessed scalability of the Pthreads- and OpenMP-based versions up to 32 CPUs and up to 256 CPUs for the MPI-based version.

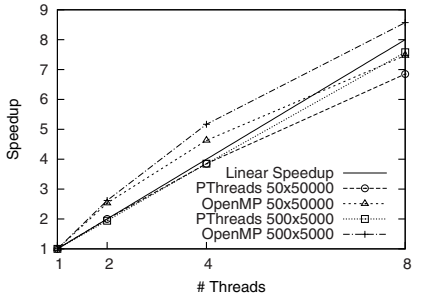
As outlined above we directly compare MPI, Pthreads, and OpenMP on the SGI Altix and AMD Opteron cluster. In addition, we provide comparisons between OpenMP and Pthreads on the Barcelona, Clovertown, and x4600 systems.

4.2 Results

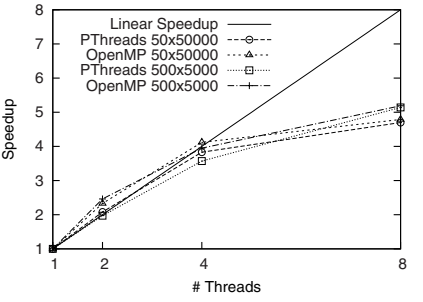
In Figures 2(a) through 2(c) we indicate speedup values for the Pthreads and OpenMP versions on datasets d50_50000 and d500_5000 on the three multi-core systems: Barcelona, Clovertown, and x4600. We show results for these two datasets because they differ significantly in their computation to communication ratio, i.e., this ratio is approximately 100 times less favorable for dataset d500_5000. Note that, the memory footprint of dataset d500_5000 is about twice as high as for d50_50000. On the Barcelona both versions scale almost linearly up to 8 cores, while there is a significant decrease in parallel efficiency on the Clovertown. This is due to the UMA architecture and the L2 cache which is shared between each two cores: the memory bandwidth can be saturated by only 4 threads and the cache available to each thread is halfed if all cores are utilized. Both Pthreads and OpenMP scale well up to 16 cores on the x4600. However, there is a significant decrease in parallel efficiency for the OpenMP-based version on the more communication-intensive dataset d500_5000 above 8 cores. Thus, the Pthreads-based communication mechanisms we implemented, yields significantly better speedups for the full 16 cores on this system.

In Figure 2(d) we provide the *relative* speedups (relative with respect to a run with 31 worker processes[1]) for the MPI version on the large and memory-intensive

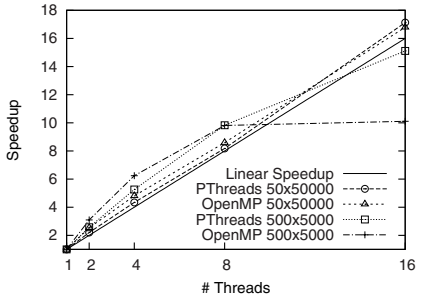
[1] A sequential execution was not possible due to run-times and memory requirements.



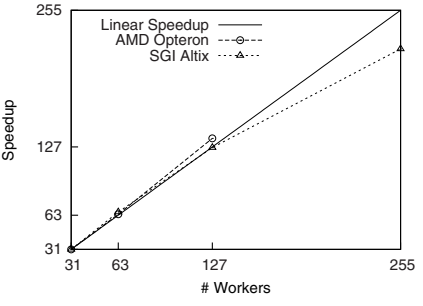
(a) Speedup of Pthreads and OpenMP versions for datasets d50_50000 and d500_5000 on the AMD Barcelona



(b) Speedup of Pthreads and OpenMP versions for datasets d50_50000 and d500_5000 on the Intel Clovertown



(c) Speedup of Pthreads and OpenMP versions for datasets d50_50000 and d500_5000 on the x4600

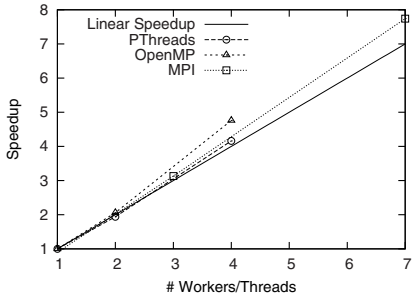


(d) Scalability of the MPI-based version for dataset 250_500000 on the Opteron cluster and the SGI Altix 4700

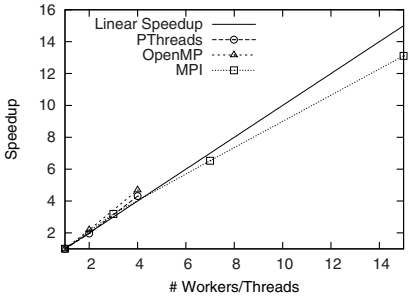
Fig. 2. Scalability of Pthreads, OpenMP, and MPI versions on various architectures

d250_500000 dataset up to 128 CPUs of the AMD Opteron cluster and up to 256 cores on the SGI Altix 4700. This plot demonstrates that the entirely re-designed MPI version for production runs achieves similar parallel efficiency as the previous proof-of-concept implementation [14,15].

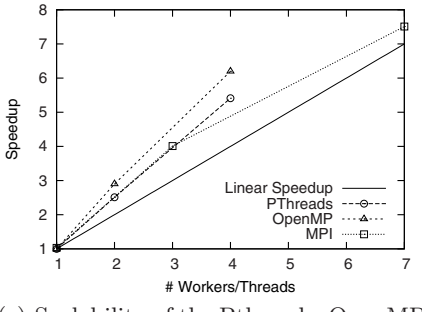
In Figures 3(a) to 3(d) we provide a direct comparison of the Pthreads, OpenMP, and MPI versions for the AMD Opteron cluster and the SGI Altix 4700. Speedup values for dataset d50_50000 on the Opteron cluster (Figure 3(a)) are super-linear due to increased cache efficiency for all three programming paradigms. On the larger d50_500000 dataset (Figure 3(b)) scalability of OpenMP and Pthreads are similar, while the MPI-based version yields slightly sub-linear speedups on the Opteron cluster for 7 and 15 worker processes. This is due to the fact that execution times in these cases become relatively short, such that the initial sequential portion of the code (striped data distribution) has an impact on performance. The more communication–intensive dataset d500_5000 also scales well for all three paradigms on the AMD Opteron system (Figure 3(c)). In general, the OpenMP version scales best on this system. However, this is not the case on all architectures, especially for more communication-intensive datasets sizes (see Figure 2(c)).



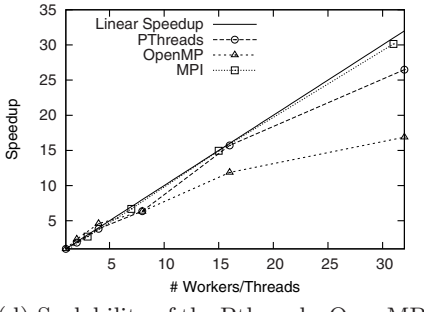
(a) Scalability of the Pthreads, OpenMP, and MPI versions for dataset d50_50000 on the Opteron cluster



(b) Scalability of the Pthreads, OpenMP, and MPI versions for dataset d50_500000 on the Opteron cluster



(c) Scalability of the Pthreads, OpenMP, and MPI versions for dataset d500_5000 on the Opteron cluster



(d) Scalability of the Pthreads, OpenMP, and MPI versions for dataset d50_50000 on the SGI Altix 4700

Fig. 3. Performance Comparison of Pthreads, OpenMP, and MPI versions

Both MPI and Pthreads also yield super-linear speedups in most cases. Finally, in Figure 3(d) we provide performance data for all three parallel versions on the SGI Altix for dataset 50_50000 up to 31 worker processes/threads. The MPI and Pthreads versions scale significantly better than OpenMP for more than 7 threads/workers which is also consistent with the observations on the x4600 (see Figure 2(c)). For more than 15 threads/workers, MPI outperforms Pthreads as the MPI version only accesses local memory while the Pthreads version has to access most of its data structures remotely at the master – with lower bandwidth and higher latency.

5 Conclusion and Future Work

We have conducted a detailed performance study of parallel programming paradigms for exploitation of fine–grained loop–level parallelism, by example of the widely used phylogenetic ML function as implemented in RAxML on a broad variety of current multi-core, cluster, and supercomputer architectures. Results

indicate that none of the three paradigms outperforms the others across all architectures. We thus conclude that the selection of programming paradigms should be based on software engineering and portability criteria.

One important aspect is that Bioinformatics applications are typically used by non-experts such that the easier to compile Pthreads option should be preferred over OpenMP and MPI for shared memory architectures. The usage of MPI on shared memory machines could lead to serious performance degradations in the case that MPI implementations are used that have not been optimized for communication via shared memory.

In terms of portability, we argue in favor of the usage of both Pthreads and MPI, since programs can easily be compiled for massively parallel machines such as the BlueGene as well as for shared memory architectures. Note that, we do not consider hybrid parallelism here because, as already mentioned, ML-based inferences exhibit embarrassing parallelism at a more coarse-grained level. In addition, Pthreads allow for explicit allocation of local memory, i.e., to distribute the data structures and thus facilitate the joint development and maintenance of the MPI and Pthreads versions. In this case, synchronization and communication can be handled via one single generic interface that can then be mapped to appropriate MPI or Pthreads constructs and greatly reduce the complexity of the code. Moreover, such an —in principle— distributed memory Pthreads-based parallelization can improve performance on NUMA architectures. A striped distribution of alignment sites, which is required to achieve load–balance on concatenated DNA and AA (Protein) data would induce a significant programming overhead in OpenMP as well. Our experiments show that the performance of the Pthreads-based and OpenMP-based implementations is platform–specific, such that one should opt for the more generic approach. Finally, the Pthreads-based version can be further improved by removal of some synchronization points and exploitation of data locality.

Thus, future work will cover the performance analysis and profiling of data locality impact for a Pthreads–based version that allocates and uses local instead of global likelihood vector data structures.

Acknowledgements

We are grateful to Olaf Bininda-Emonds, Dan Janies, and Andrew Johnson for providing us their alignments for performance tests. We would like to thank Dimitris Nikolopoulos for useful discussions on previous comparative performance studies for different programming paradigms. The Exelixis Lab (AS) is funded under the auspices of the Emmy-Noether program by the German Science Foundation (DFG).

References

1. Hamady, M., Walker, J., Harris, J., Gold, N., Knight, R.: Error-correcting barcoded primers for pyrosequencing hundreds of samples in multiplex. Nature Methods 5, 235–237 (2008)

2. Darling, A., Carey, L., Feng, W.: The Design, Implementation, and Evaluation of mpiBLAST. In: Proceedings of ClusterWorld 2003 (2003)
3. Stamatakis, A., Auch, A., Meier-Kolthoff, J., Göker, M.: Axpcoords & parallel axparafit: Statistical co-phylogenetic analyses on thousands of taxa. BMC Bioinformatics (2007)
4. Felsenstein, J.: Confidence Limits on Phylogenies: An Approach Using the Bootstrap. Evolution 39(4), 783–791 (1985)
5. Bader, D., Roshan, U., Stamatakis, A.: Computational Grand Challenges in Assembling the Tree of Life: Problems & Solutions. In: Advances in Computers. Elsevier, Amsterdam (2006)
6. Minh, B.Q., Vinh, L.S., Schmidt, H.A., von Haeseler, A.: Large maximum likelihood trees. In: Proc. of the NIC Symposium 2006, pp. 357–365 (2006)
7. Blagojevic, F., Nikolopoulos, D.S., Stamatakis, A., Antonopoulos, C.D.: Dynamic Multigrain Parallelization on the Cell Broadband Engine. In: Proceedings of the 12th ACM SIGPLAN symposium on Principles and practice of parallel programming, pp. 90–100 (2007)
8. Stamatakis, A., Ott, M., Ludwig, T.: RAxML-OMP: An Efficient Program for Phylogenetic Inference on SMPs. In: Malyshkin, V.E. (ed.) PaCT 2005. LNCS, vol. 3606, pp. 288–302. Springer, Heidelberg (2005)
9. Stamatakis, A.: RAxML-VI-HPC: maximum likelihood-based phylogenetic analyses with thousands of taxa and mixed models. Bioinformatics 22(21), 2688–2690 (2006)
10. Felsenstein, J.: Evolutionary trees from DNA sequences: a maximum likelihood approach. Journal of Molecular Evolution 17, 368–376 (1981)
11. Dunn, C.W., Hejnol, A., Matus, D.Q., Pang, K., Browne, W.E., Smith, S.A., Seaver, E., Rouse, G.W., Obst, M., Edgecombe, G.D., Sorensen, M.V., Haddock, S.H.D., Schmidt-Rhaesa, A., Okusu, A., Kristensen, R.M., Wheeler, W.C., Martindale, M.Q., Giribet, G.: Broad phylogenomic sampling improves resolution of the animal tree of life. Nature (2008) (advance on-line publication)
12. Robertson, C.E., Harris, J.K., Spear, J.R., Pace, N.R.: Phylogenetic diversity and ecology of environmental Archaea. Current Opinion in Microbiology 8, 638–642 (2005)
13. Charalambous, M., Trancoso, P., Stamatakis, A.: Initial Experiences Porting a Bioinformatics Application to a Graphics Processor. In: Bozanis, P., Houstis, E.N. (eds.) PCI 2005. LNCS, vol. 3746, pp. 415–425. Springer, Heidelberg (2005)
14. Ott, M., Zola, J., Aluru, S., Johnson, A.D., Janies, D., Stamatakis, A.: Large-scale Phylogenetic Analysis on Current HPC Architectures. Scientific Programming (Submitted, 2008)
15. Ott, M., Zola, J., Aluru, S., Stamatakis, A.: Large-scale Maximum Likelihood-based Phylogenetic Analysis on the IBM BlueGene/L. In: Proceedings of IEEE/ACM Supercomputing Conference 2007 (2007)
16. Berlin, K., Huan, J., Jacob, M., Kochhar, G., Prins, J., Pugh, B., Sadayappan, P., Spacco, J., Tseng, C.: Evaluating the Impact of Programming Language Features on the Performance of Parallel Applications on Cluster Architectures. In: Rauchwerger, L. (ed.) LCPC 2003. LNCS, vol. 2958. Springer, Heidelberg (2004)
17. Cappello, F., Etiemble, D.: MPI versus MPI+ OpenMP on the IBM SP for the NAS Benchmarks. In: Proc. Supercomputing 2000, Dallas, TX (2000)
18. Krawezik, G., Alleon, G., Cappello, F.: SPMD OpenMP versus MPI on a IBM SMP for 3 Kernels of the NAS Benchmarks. In: Zima, H.P., Joe, K., Sato, M., Seo, Y., Shimasaki, M. (eds.) ISHPC 2002. LNCS, vol. 2327. Springer, Heidelberg (2002)

19. Jones, M., Yao, R.: Parallel programming for OSEM reconstruction with MPI, OpenMP, and hybrid MPI-OpenMP. Nuclear Science Symposium Conference Record, 2004 IEEE 5 (2004)
20. Shan, H., Singh, J., Oliker, L., Biswas, R.: A Comparison of Three Programming Models for Adaptive Applications on the Origin2000. Journal of Parallel and Distributed Computing 62(2), 241–266 (2002)
21. Minh, B.Q., Vinh, L.S., von Haeseler, A., Schmidt, H.A.: pIQPNNI: parallel reconstruction of large maximum likelihood phylogenies. Bioinformatics 21(19), 3794–3796 (2005)
22. Guindon, S., Gascuel, O.: A Simple, Fast, and Accurate Algorithm to Estimate Large Phylogenies by Maximum Likelihood. Systematic Biology 52(5), 696–704 (2003)
23. Zwickl, D.: Genetic Algorithm Approaches for the Phylogenetic Analysis of Large Biological Sequence Datasets under the Maximum Likelihood Criterion. PhD thesis, University of Texas at Austin (April 2006)
24. Ronquist, F., Huelsenbeck, J.: MrBayes 3: Bayesian phylogenetic inference under mixed models. Bioinformatics 19(12), 1572–1574 (2003)
25. McMahon, M.M., Sanderson, M.J.: Phylogenetic Supermatrix Analysis of GenBank Sequences from 2228 Papilionoid Legumes. Systematic Biology 55(5), 818–836 (2006)
26. Tavar, S.: Some Probabilistic and Statistical Problems in the Analysis of DNA Sequences. Some Mathematical Questions in Biology: DNA Sequence Analysis 17 (1986)
27. Yang, Z.: Maximum likelihood phylogenetic estimation from DNA sequences with variable rates over sites. Journal of Molecular Evolution 39, 306–314 (1994)
28. Stamatakis, A.: The RAxML 7.0.4 Manual, The Exelixis Lab. LMU Munich (April 2008)
29. Bininda-Emonds, O., Cardillo, M., Jones, K., MacPhee, R., Beck, R., Grenyer, R., Price, S., Vos, R., Gittleman, J., Purvis, A.: The delayed rise of present-day mammals. Nature 446, 507–512 (2007)
30. Ott, M., Klug, T., Weidendorfer, J., Trinitis, C.: Autopin - Automated Optimization of Thread-to-Core Pinning on Multicore Systems. In: Proceedings of 1st Workshop on Programmability Issues for Multi-Core Computers (MULTIPROG) (January 2008)

Massively Parallelized DNA Motif Search on the Reconfigurable Hardware Platform COPACOBANA

Jan Schröder, Lars Wienbrandt, Gerd Pfeiffer, and Manfred Schimmler

Department of Computer Science, Christian-Albrechts-University of Kiel, Germany
{jasc,lwi,gp,masch}@informatik.uni-kiel.de

Abstract. An enhanced version of an existing motif search algorithm BMA is presented. Motif searching is a computationally expensive task which is frequently performed in DNA sequence analysis. The algorithm has been tailored to fit on the COPACOBANA architecture, which is a massively parallel machine consisting of 120 FPGA chips. The performance gained exceeds that of a standard PC by a factor of over 1,650 and speeds up the time intensive search for motifs in DNA sequences. In terms of energy consumption COPACOBANA needs 1/400 of the energy of a PC implementation.

Keywords: Motif finding, DNA sequence analysis, FPGA, High Performance Reconfigurable Computing (HPRC).

1 Introduction

The discovery of regulatory sequences in DNA - called motif-finding - is one of the most challenging problems in the field of bioinformatics. In fact there are problem instances of motif-finding which are unsolvable by current techniques. There are two reasons that make this problem so difficult: firstly, the parameters of a given problem instance (like sequence length, motif length, grade of mutation) can make it impossible to identify motifs due to background noise. Secondly, it is computationally expensive. So a precise algorithm can fail to discover a motif in a given sequence because its execution time exceeds rational means. We address both problems with a new approach to motif searching making use of a novel massively parallel architecture to speed up the execution time.

Motif searching has been an issue in many publications of the last ten years. As the most popular approaches to this topic we reference MEME [16] [17] [18] and the similar Gibbs sampler [14] [15] which iteratively develops matrices representing motifs of the input sequence using the expectation maximization technique; the projection algorithm [13] [20] which creates a representation of the highly conserved region over all motif instances; and CONSENSUS [21] - a greedy approach which constructs likely motif candidates by aligning only small parts of the genome at a time.

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 436–447, 2008.

The algorithm IGOM (Iterative Generation of position frequency matrices) has been published in [22]. This method iteratively develops a set of strings which are likely to be instances of an underlying motif by featuring two new ideas. It makes use of the structure of position frequency matrices of already known motifs which imply a distribution on only one or two nucleotides in each position rather than all four of them [19]. The *sp*-model has been introduced in [22] to describe this restriction. This observation is utilized to develop a very precise description of a kernel of the motif in the first few iterations of the algorithm. This has the advantage that the likelihood of false positives which fit to this description although not belonging to the motif is minimized. Regulatory sequences that match the observations of the *sp*-model (for example the *SigmaB* regulator in *Bacillus subtilis* [24]) are discovered easier and more accurately by this algorithm compared to the other methods of motif searching.

The second key feature of IGOM is the surveillance of the expected false positives which could arise from loosening the description of the motif. The algorithm will only make those changes to the matrix where the quotient of valid new candidates divided by the expected number of random strings which fit this change - and appear in any sequence of the given length without relevance - is maximal. The authors describe this quotient with the term *signal to noise ratio* (SNR) because of its correlation to signal theory where one tries to maximize the signal opposing to the background noise of the medium.

Further improvements of the algorithm has been published in [23]. The main idea of this publication is a Boolean representation of motif kernels. Instead of position frequency/weight matrices we use Boolean matrices to describe a motif. A value of "1" in a Boolean matrix (BM) considers the nucleotide to be a valid representation for a motif instance in the corresponding position [23]. It leads to a huge improvement of the complexity and makes this method highly applicable for special purpose architectures. Most of the methods in Bioinformatics gain performance when applied to special hardware because of the small alphabet sizes when dealing with DNA or protein sequences and simple operations on the input data. We chose to implement the IGOM/BMA algorithm in hardware because of its ideal qualifications:

1. The input data can be represented in a very efficient way with only two bits per nucleotide
2. The algorithm can be parallelized in an ideal way because of the independent search operations on the data.
3. The Boolean matrices used to represent motifs can be stored very efficiently in hardware allowing many processes working on a single FPGA chip simultaneously.

The amount of biological sequence information is increasing more rapidly [1] than the exponential performance growth of general purpose microprocessor-based computers. Due to this observation highly optimized special-purpose computers have been developed. Today, the technology of Field Programmable Gate Arrays (FPGAs) exhibit impressive performance compared to microprocessor-based machines, among other things in the field of bioinformatics. Successful

special purpose hardware are for example SPLASH 2 [2], JBits [3], BEE2 [4], XD1000 [5], RASC RC100 [6], and DeCypher [7]. The recent massively parallel FPGA-based architecture COPACOBANA [8] from *SCIENGINES* [9] is chosen as target for the proposed motif search algorithm. Taking advantage of the hardware architecture and the highly parallel nature of the algorithm we can accelerate this method with huge efficiency. Implementing the iterative development of motif kernels on the COPACOBANA we outperform a single desktop PC by a factor of over $1,650$. Taking into account the higher cost of the COPACOBANA, a cost performance ratio would be fairer for comparison. This leads to a performance per cost ratio up to five times higher compared to desktop PCs and of course accordingly faster execution time. Additionally the power consumption of PCs for the same task is much higher than that of COPACOBANA. We reach an energy efficiency more than 420 times better than standard PCs.

This paper is organized as follows. In chapter 2 the COPACOBANA hardware is specified, in chapter 3 the implemented algorithm is described. Chapter 4 will discuss the details concerning the implementation of the algorithm in hardware. Performance analysis, conclusion and outlook will follow in chapters 5 and 6.

2 COPACOBANA

The massively parallel computer COPACOBANA consists of 120 low cost FPGAs which are connected to a controller module by a bus system. It can be integrated in any standard Local Area Network (LAN) environment and is fully remotely controlled. Originally COPACOBANA has been developed as C*ost-*O*ptimized* PA*rallel* CO*de* B*reaker* in 2006. The goal was to break the 56-bit Data Encryption Standard (DES) in 10 days for production and material cost of less than $\$10,000$. [10] Actually it breaks DES in 7 days [8] in the mean. Due to the universality of FPGA-chips [11] this machine is suited for all kinds of fine grained parallel applications with low communication and memory requirements, and with special attention to the cost/performance ratio.

The FPGAs are of the type *Xilinx Spartan-3 1000* [12] (*XC3S1000*, speed grade -4, FTG256 packaging). Each comes with 1 million system gates, $17,280$ equivalent logic cells, $1,920$ Configurable Logic Blocks (CLBs) equivalent to $7,680$ slices, 120 kbit distributed RAM, 432 kbit Block RAM (BRAM), 24 dedicated $18x18$ multipliers, and 4 digital clock managers (DCMs). Figure 1 depicts the data path of COPACOBANA. Pluggable cards in DIMM format are holding 6 FPGAs each. Twenty of these cards are plugged into slots of a common backplane together with a controller card. The latter is the interface to a host computer via Ethernet LAN. It is transferring data and controlling the single master bus system which is currently operating at up to 1 Mbit/s. The host computer is executing a front-end software which uses an Application Programming Interface (API) for accessing COPACOBANA. Additionally some parts of the target algorithm are implemented here which for instance are sequential, perform a post- or preprocessing, or access a hard drive. This software represents the highest control instance, because it initiates any action of the controller, hence

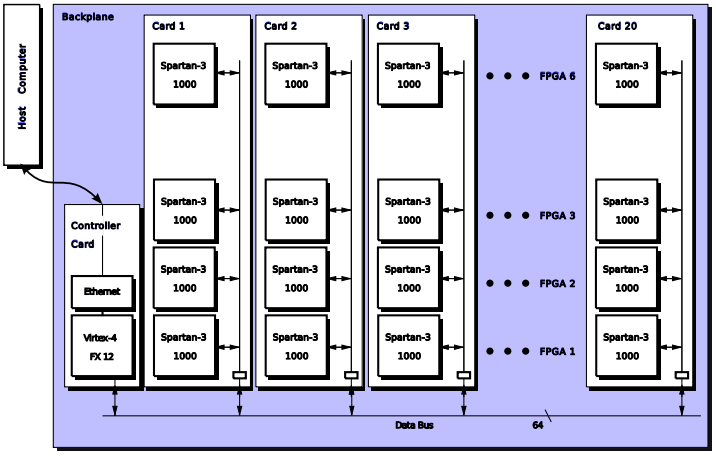


Fig. 1. COPACOBANA Data Path

it controls the entire machine. In other words, communication can not be initialized by one of the 120 slave FPGAs because COPACOBANA does not support interrupts. Therefore a static communication scheduling has to be considered for the host software.

The controller provides the following addressing modes. A single FPGA can be selected for writing and reading data. Any set of FPGAs on one card up to all 6 can be written to from the controller, and finally via broadcast the controller can write data to all 120 FPGAs. Each of the FPGAS can be configured to suit its purpose exactly: small processing units can be implemented on the chip that are designed only for one specific task.

3 Algorithm

In this section a description of the BMA algorithm is given. Since we aim for a massively parallel implementation (see section 4) it will be slightly modified with respect to the scoring function given in [22] and [23]. Given the input data - a whole genome or a particular set of sequences - and a fixed motif length $l2$ the algorithm will develop motif kernels in increasing order by the likelihood of their occurrence in a randomly distributed sequence. So assuming a normal distribution of the input data we are interested in the least likely occurrence of motif candidates in terms of over-representation. We will analyze the signal to noise ratio (SNR) to find those candidates:

1. The algorithm starts with a single string of the motif length and specifies the Boolean matrix.
2. Each iteration it will modify one column of the matrix so that two nucleotides will represent the given position of the motif - following the conclusions of the *sp*-model. The algorithm chooses the position in the matrix by analyzing

the SNR so it minimizes the probability of false positives and finds the best representation of the motif.
3. Beneath all the matrices generated each round (one for every start string) the best in terms of SNR are chosen and analyzed further if they are likely to represent a real motif in the organism represented by the input data. We will not discuss this third step in this paper since only the development of motif kernels is the time consuming part of the method which we apply to hardware.

For the sake of an efficient implementation we will restrict the algorithm in the following way. In each iteration there will be one change of the matrix - whether it is good or not in terms of SNR - and every change will add a "1" to a column of the matrix where there was exactly one "1" before. This has the great benefit that every matrix in the same round of the algorithm has exactly the same value for the expected noise. So SNR can be compared easily only by analyzing the number of candidates from the input data matching the describing matrices. We can take great advantage of this restriction in the implementation because it allows much simpler and smaller units processing the matrices.

3.1 Example

To illustrate how the algorithm works, we show a short example. Let the BM look like the first matrix illustrated in figure 2 after the first iteration of the algorithm. So the strings "GAAGT" and "GCAGT" match the matrix. In the second iteration all strings that match all but one position of the BM will contribute to a scoring matrix. For example, the string "GCAAT" would score for an "A" in the fourth column, whereas the string "AAAAT" would not contribute at all because it has too many mismatches in this iteration. After scoring all substrings of the genome in this manner a scoring matrix like the second matrix in figure 2 could arise. With the identified maximum of the scoring matrix (marked in the figure) the two strings "GAAAT" and "GCAAT" will be taken into the motif kernel forming a new BM, which is depicted as the third matrix in figure 2.

A	0	1	1	0	0	A	0	x	x	**4**	1	A	0	1	1	**1**	0
C	0	1	0	0	0	C	1	x	1	0	2	C	0	1	0	0	0
G	1	0	0	1	0	G	x	1	3	x	2	G	1	0	0	1	0
T	0	0	0	0	1	T	0	1	0	0	x	T	0	0	0	0	1

Fig. 2. Example: Boolean matrix at the beginning of an iteration, matching the strings "GAAGT" and "GCAGT", and a possible scoring matrix with the resulting new boolean matrix after the iteration.

4 Hardware Implementation

4.1 Parallel Processing Scheme

Since we are starting without any knowledge about possible motif candidates, the algorithm requires to analyze any possible position frequency matrix (PFM). For

a motif length of 12 nucleotides there are $4^{12} = 16,777,216$ such PFMs. There is no data dependency between any two of them. So, we can use a trivial parallelization scheme where a maximum number of PFMs is computed in parallel. COPACOBANA contains 120 FPGA chips. Each of them can be configured to provide 32 independent search entities. This accumulates to $3,840$ search entities to work concurrently.

The DNA is viewed as a sequence over the alphabet $\{A, C, G, T\}$. Every character can be represented with two bits. The restricted size of the local memory of the *Spartan-3* chips does not allow to store the complete DNA sequence in every search entity. Instead, it is provided by globally broadcasting it to all search entities character by character. Each entity continuously accumulates the relevant information to update its particular PFM using the globally broadcasted data stream.

Since 4^{12} is greater than $3,840$ it is necessary to compute the complete problem in $4^{12}/3,840 = 4,370$ subsequent identical computation runs. Each run requires a fixed number of iterations for updating the PFMs. It has turned out that more than six iterations do not provide useful results anymore. Therefore, the complete DNA sequence has to be broadcast to all processors a total number of $4,370 \cdot 6 = 26,220$ times. In our implementation, the DNA sequence is locally stored in the controller of COPACOBANA in order to reduce the traffic on the TCP/IP connection.

The PFM analysis is done in four steps:

1. The host application sends a command to initialize the search entities on the FPGAs. This command also provides the initialization matrix in form of an index. The index is in the range from 0 to $16,777,215$, each identifying a unique PFM.

The following steps will be repeated six times:

2. Each search entity scores all subsequences of the broadcasted DNA sequence against its own PFM.
3. The local results are read from the search entities. The best scores are stored in sorted lists on the host. There is one list for each iteration after initialization, so there will finally be six lists in this case. The number of best results saved is user defined.
4. Finally the host sends an update command to alter the position frequency matrices in the search entities.

After all six iterations have finished the next initialization is done with new indices, i.e. with new PFMs. The algorithm starts again with step 1. When the application has finished the lists with the user defined amount of best results for each of the six iterations are ready for further analysis.

4.2 FPGA Design

The main processing unit is the *search entity* which provides the core functionality of the algorithm. 32 search entities fit on a single FPGA chip and thus can work in parallel. In the following one of those is described in detail.

Every search entity has a unique identifier for individualization. The identifier is a natural number starting with zero. One search entity consists of an implementation of the boolean position frequency matrix and its matching functionality, a score counter and a counter for differing sequences, further called "difference counter".

Initially, the search entity receives an index which corresponds to a gene sequence of length 12, the expected motif size. By adding the identifier of the search entity to the incoming index every entity generates its individual initialization sequence. Hence, an FPGA has to be provided only one time with an initialization index to initialize 32 search entities at once. The initialization sequence is easily converted to the matrix structure by using lookup tables. "*A*" is "1000", "*C*" is "0100" etc.

An incoming gene sequence is matched with the position frequency matrix. If the sequence matches the score counter increments a locally stored score value. For every matrix position a counter is provided which is increased whenever the string under consideration has a mismatch in this position and it is the only mismatch with the PFM. This implicates that with a motif size of 12 we need 48 counters for each search entity. Given the fact that only one counter per search entity has to be accessed at maximum in one clock cycle, the counters can easily be stored in the local block RAM which is available in every *Spartan-3* FPGA.

The only data locally stored by the difference counter is the maximum counter value and a corresponding gene sequence which causes this counter to increment. This makes a matrix update fast and easy. The update command converts the sequence to the matrix structure like for the initialization followed by a simple or-operation on the old matrix. Every search entity provides its position frequency matrix and score as result.

The control of the motif search operation is realized by the *search control entity*. It manages the incoming control instructions and user data from the host application. The user data is read from the bus in 64 bit blocks which equals 32 characters. It is then provided to a FIFO buffer as a data stream. The buffer always provides a window of 12 characters as data input to the search entities. The search control entity also provides the best result of the search entities to the host application. Therefore the results are compared by their score. The comparison is made by comparators which are aligned to each search entity in a chain. Every comparator compares the result of its predecessor and one search entity. If the best result is fetched by the host application its score is cleared on the corresponding search entity. Hence, the second best result will be automatically provided to the host.

Figure 3 shows a simplified overview of the FPGA design.

4.3 Data and Control Flow on the FPGA

The search entities are organized on the FPGA in two chains due to the two rows of block RAM on the *Spartan-3*. All user and control data is buffered by one entity and provided to the successor in its chain in the next clock cycle. This keeps data paths short and permits higher frequencies. Except for the command

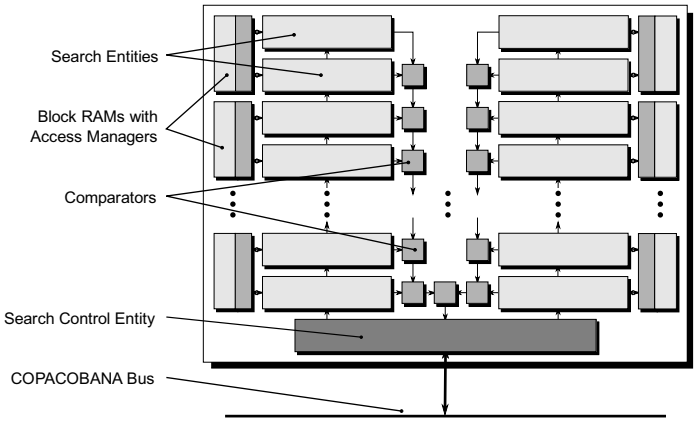


Fig. 3. FPGA design overview

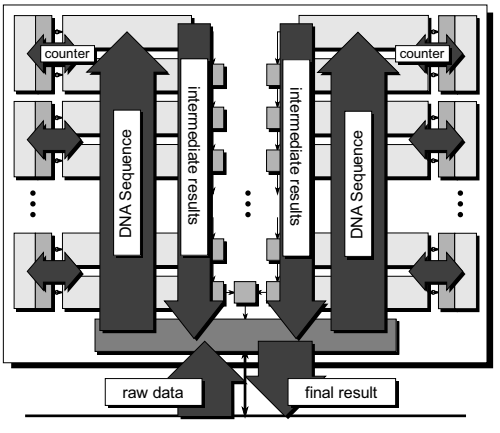


Fig. 4. FPGA dataflow overview

to read a result, all control instructions plus the user data from the host are provided by the search control entity directly to the first two search entities in the chains.

The comparators for the results are organized in two chains along the search entities as well. Every comparator compares the result of its predecessor and one search entity in one clock cycle. At the beginning of the chain the first comparators compare the results of the last two search entities. So the data flow of the results is contrary to the data flow of the sequence data. This again keeps data paths short because the maximum result of both chains is provided back to the search control entity after a final comparison. The signal to clear the best results score after being fetched by the host is routed through the comparator chain as well.

The overview of the design flow is shown in figure 4.

5 Performance Analysis and Conclusion

5.1 *C++* Implementation

For comparison the DNA motif search algorithm has been implemented in *C++*. It has been compiled with the *GNU Compiler Collection* (GCC) v4.0.2 and the "`-O3`"-flag for highest optimization. Additionally the "`-march=...`"-flag and other optimization flags like "`-msse3`" were set for the corresponding target architecture. The testing systems were a standard PC with an *Intel Pentium IV* at 2.8 GHz and a PC with an *AMD Thurion64 X2* dual core at 1.9 GHz running a *Linux* operating system, and a *Macintosh Pro* with two *Intel Xeon 5150* dual cores at 2.6 GHz running *Mac OS X*. The implementation uses static memory for the gene sequence, the score and the counter values for the missed matches. So no new memory is allocated dynamically at runtime except for new results in the lists. Since we store only 100 results for each iteration the allocations are very scarce and do not significantly delay the process. Actually the host application using COPACOBANA does the same. For the dual core architectures the task was equally divided into two processes.

Because the algorithm can not be parallelized for one processing core the application takes one initial position frequency matrix at a time. But we made one significant improvement which could not be applied to the parallelized solution. This application does not always perform six iterations per initialization index. It does a further iteration only if the position frequency matrix was updated in the preceding one. This causes a significant speedup for the iterative solution.

5.2 Performance

The applications were configured to analyze the DNA sequences of *Cowpox Virus* (280*k* bases), *Rickettsia canadensis str. McKiel* (1.2*M* bases) and *Bacillus subtilis* (5.9*M* bases) as an example. The desired motif size is 12 and the number of iterations is set to 6. Table 1 shows the duration of the computation and the speedup of COPACOBANA vs. the specified architectures. Figure 5 shows

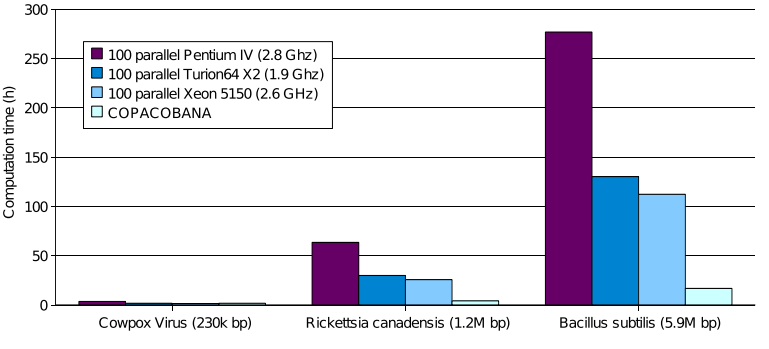


Fig. 5. Computation times for *Cowpox Virus*, *Rickettsia canadensis* and *Bacillus subtilis*

Table 1. computation times and speedups of the DNA motif search algorithm

		COPACOBANA	*Xeon 5150* 2.6 GHz dual core	*Thurion64 X2* 1.9 GHz dual core	*Pentium IV* 2.8 GHz single core
Cowpox	time	$1h40m$	$144h$ (6 d.)	$167h$ (7 d.)	$355h$ (14.8 d.)
Virus	speedup	1	> 86	> 100	> 210
Rickettsia	time	$4h10m$	$2,575h$ (107.3 d.)[1]	$2,987h$ (124.5 d.)[1]	$6,350h$ (264.6 d.)[1]
canadensis	speedup	1	> 615	> 715	$> 1,520$
Bacillus	time	$16h45m$	$11,236h$ (1.3 y.)[1]	$13,031h$ (1.5 y.)[1]	$27,700h$ (3.2 y.)[1]
subtilis	speedup	1	> 670	> 775	$> 1,650$

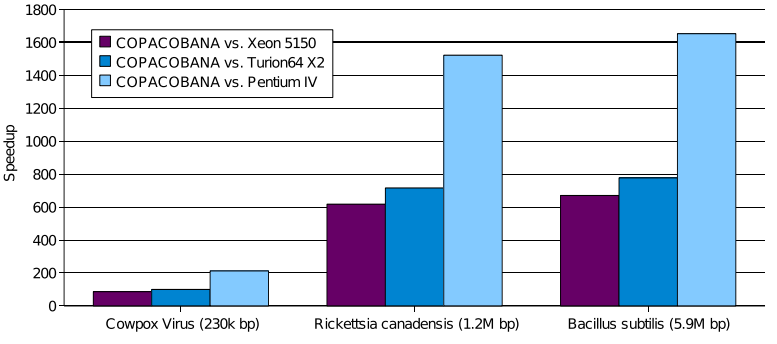


Fig. 6. COPACOBANA speedups vs. several processors

a graphical presentation of these computation times. To make the results more reasonable the computation times for the PCs are adapted to match 100 ideally parallelized PCs of each type in this figure. Figure 6 shows the speedups of COPACOBANA vs. one PC of the specified architecture.

5.3 Conclusion

Although significant improvements have been made to the iterative algorithm the parallel solution generates the same results in much shorter time. Previously nearly unreachable results due to the length of the computation time could now be afforded in less than one day. Unfortunately the drawback is the need of the special purpose hardware, but with the cost of €60,000 for a COPACOBANA and €200 for a standard PC the cost/performance ratio is only $(€60,000/€200)/1,650 = 0.18$. This means COPACOBANA is more than 5 times more cost effective than a standard PC. Another advantage of COPACOBANA is energy efficiency. Due to the short computation time and only 600W power consumption it consumes about 10.5 kWh to calculate the motif candidates for *Bacillus subtilis*. In contrast standard PCs consume about 4,155

[1] This value is computed by measuring a small part and extrapolating the duration.

kWh for the same task at 150W per PC. Assuming energy costs of €0.20/kWh this would be €831.00 against €2.10. This is about 400 times the energy costs of COPACOBANA. Easy calculation shows that the hardware price of COPACOBANA would be payed for energy costs of a PC cluster solving only 100 problems of the size of calculating motif candidates of *Bacillus subtilis*. Additionally the costs to build a cluster out of several PCs to reach the performance of one COPACOBANA are not considered. These are costs for connection cables, switches and the place to deposit these components. Furthermore the knowledge to get such a PC cluster working with this application has to be paid for as well.

6 Outlook

The performance analysis for COPACOBANA is made with a slow controller having a very little bandwidth of approximately 1 Mbit/s. There is already a new controller module under development which reaches a bandwidth of about 100 Mbit/s. First tests with this application reached 5 to 7 times the speed of the slow controller. Hence the controller is still the bottleneck of this application and even greater speedups could be reached easily. This leads to another improvement of the cost performance ratio and energy efficiency.

With the speedup gained by the COPACOBANA implementation we can intensify motif searching on real datasets. We will put it to use by analyzing motifs in virus datasets in close collaboration with the medical institute of the Free University of Berlin.

References

1. Meyer, F.: Genome Sequencing vs. Moore's Law. In Cyber Challenges for the Next Decade. CTWatch Quarterly 2, 1–2 (2006)
2. Hoang, D.T.: Searching Genetic Databases on SPLASH 2. In: Proc. Workshop on FPGAs for Custom Computing Machines (1993)
3. Guccione, S.A., Keller, E.: Gene Matching Using JBits. In: Proc. 12th Field Programmable Logic and Applications. Springer, Berlin (2002)
4. Herbordt, M.C., Model, J., Sukhwani, B., Gu, Y., Van Court, T.: Single pass streaming BLAST on FPGAs. In: Parallel Computing. Special issue on High-Performance Computing Using Accelerators, vol. 33, pp. 741–756 (2007)
5. XtremeData, Inc., http://www.xtremedatainc.com/
6. Silicon Graphics, Inc., http://www.sgi.com/products/rasc/
7. Time Logic Corp., http://www.timelogic.com/
8. COPACOBANA Research Project, http://www.COPACOBANA.org/
9. SCIENGINES Corp., http://www.sciengines.com/
10. Kumar, S., Paar, C., Pelzl, J., Pfeiffer, G., Rupp, A., Schimmler, M.: How to Break DES for € 8,980. In: 2nd Workshop on Special-purpose Hardware for Attacking Cryptographic Systems - SHARCS, April 3-4, Cologne, Germany (2006)
11. DeHon, A.: The Density Advantage of Configurable Computing. IEEE Computer 33(4), 41–49 (2000)
12. Xilinx Inc., http://www.xilinx.com/

13. Buhler, J., Tompa, M.: Finding motifs using random projections. J. Comput. Biol. 9, 225–242 (2002)
14. Lawrence, C.E., Reilly, A.A.: An expectation maximization (EM) algorithm for the identification and characterization of common sites in unaligned biopolymer sequences. Proteins 7, 41–51 (1990)
15. Lawrence, C.E., Altschul, S.F., Boguski, M.S., Liu, J.S., Neuwald, A.F., Wootton, J.C.: Detecting subtle sequence signals: a Gibbs sampling strategy for multiple alignment. Science 262, 208–214 (1993)
16. Bailey, T.L., Elkan, C.: Fitting a mixture model by expectation maximization to discover motifs in biopolymers, UCSD Technical Report, CS94-351. University of California at San Diego (March 1994)
17. Bailey, T.L., Elkan, C.: Unsupervised Learning of Multiple Motifs in Biopolymers using EM. Machine Learning 21(1-2), 51–80 (1995)
18. Redhead, E., Bailey, T.L.: Discriminative motif discovery in DNA and protein sequences using the DEME Algorithm. BMC Bioinformatics 8, 385 (2007)
19. RegTransBase, http://regtransbase.lbl.gov/
20. Varghese, G., Raphael, B., Lung-Tien Liu, A.: Uniform Projection Method for Motif Discovery in DNA Sequences. IEEE Transactions On Computational Biology And Bioinformatics 1(2) (April-June 2004)
21. Hertz, G., Stormo, G.: Identifying DNA and protein patterns with statistically significant alignments of multiple sequences. Bioinformatics 15(7-8), 563–577 (1999)
22. Schröder, J., Schimmler, M., Tischer, K., Schröder, H.: IGOM - Iterative Generation of Position Frequency Matrices (submitted, 2008), http://www.informatik.uni-kiel.de/fileadmin/arbeitsgruppen/technical_cs/Files-Jan/IGOM_paper.pdf
23. Schröder, J., Schimmler, M., Tischer, K., Schröder, H.: BMA - Boolean Matrices as Model for Motif Kernels. In: 2008 International Conference on Bioinformatics, Computational Biology, Genomics, and Chemoinformatics (BCBGC 2008) (July 2008), http://www.informatik.uni-kiel.de/fileadmin/arbeitsgruppen/technical_cs/Files-Jan/paper_bcbgc125.pdf
24. Petersohn, A., Brigulla, M., Haas, A., Hoheisel, J., Völker, U., Hecker, M.: Global Analysis of the General Stress Response of Bacillus subtilis. Journal Of Bacteriology, 5617–5631(October 2001)

GPU-MEME: Using Graphics Hardware to Accelerate Motif Finding in DNA Sequences

Chen Chen, Bertil Schmidt, Liu Weiguo, and Wolfgang Müller-Wittig

School of Computer Engineering, Nanyang Technological University, Singapore
{cchen,asbschmidt,liuweiguo,askwmwittig}@ntu.edu.sg

Abstract. Discovery of motifs that are repeated in groups of biological sequences is a major task in bioinformatics. Iterative methods such as expectation maximization (EM) are used as a common approach to find such patterns. However, corresponding algorithms are highly compute-intensive due to the small size and degenerate nature of biological motifs. Runtime requirements are likely to become even more severe due to the rapid growth of available gene transcription data. In this paper we present a novel approach to accelerate motif discovery based on commodity graphics hardware (GPUs). To derive an efficient mapping onto this type of architecture, we have formulated the compute-intensive parts of the popular MEME tool as streaming algorithms. Our experimental results show that a single GPU allows speedups of one order of magnitude with respect to the sequential MEME implementation. Furthermore, parallelization on a GPU-cluster even improves the speedup to two orders of magnitude.

1 Introduction

A major challenge in computational genomics nowadays is to find patterns (or *motifs*) in a set of sequences. In particular, discovering motifs that are crucial for the regulation of gene transcription in DNA (such as Transcription Factor Binding Sites) are of growing importance to biological research. With the production of vast quantities of data, genomic researchers want to perform this analysis on a larger scale, which in turn leads to massive compute requirements. In this paper we show how modern streaming architectures can be used to accelerate this highly compute-intensive task by one to two orders of magnitude.

Algorithmic approaches to motif discovery can be classified into two main categories: *iterative* and *combinatorial*. Iterative methods are based on local stochastic search techniques such as expectation maximization (EM) [1, 2] or Gibbs sampling [5], while combinatorial algorithms use deterministic methods like dictionary building [8] or word enumeration [11]. Iterative methods are often preferred since they are using PSSMs (*Position Specific Scoring Matrices*) instead of a simple Hamming distance to describe the matching between a motif instance and a sequence. Among the iterative approaches, MEME (*Multiple EM for Motif Elicitation*) [2, 3] is a popular and well established method. However, its complexity is $O(N^2 \cdot L^2)$, where N is the number of input sequence and L is the length of each sequence. Therefore, this approach is time consuming for applications involving large data sets such as whole-genome motif discovery. Corresponding runtime requirements are likely to become

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 448–459, 2008.

even more severe due to the rapid growth in the size of available genomic sequence and transcription data. An approach to get results in a shorter time is to use high performance computing. Previous approaches to accelerate the motif finding process are based on expensive compute clusters [3] and specialized hardware [9].

This paper presents a proof-of-concept parallelization of motif discovery with MEME on commodity graphics hardware (GPUs) to achieve high performance at low cost. Our software currently supports the OOPS (one occurrence per sequence) and ZOOPS (zero or one occurrence per sequence) search models for DNA sequences. Our future work includes integrating the more complex TCM (two-component mixture) model and making the software available for public use. We are also planning to port the presented GLSL code to the newly released CUDA programming interface for GPU programming, which was not was not available at the time of writing the GPU-MEME code. Our achieved speedups on an NVIDIA GeForce 8800 GTX compared to the sequential MEME implementation are between 9 (for small data sets) and 12 (for large data sets). The runtime on a single GPU also compares favourably to the MPI-based ParaMEME running on a cluster with 12 CPUs. Furthermore, we have combined the fine-grained GPU parallelization with a coarse-grained parallel approach. This hybrid approach improves the speedup on a cluster of six GPUs to over 60.

The rest of this paper is organized as follows. In Section 2, we provide necessary background on motif discovery and general-purpose computing on GPUs. Section 3 presents our parallel streaming algorithm for motif finding. Performance is evaluated in Section 4. Finally, Section 5 concludes the paper.

2 Background

2.1 Motif Discovery

Iterative methods like EM search for motifs by building statistical motif models. A motif model is typically represented by a matrix (θ). For a motif of width W and an alphabet $\Sigma = \{x_0,\ldots,x_{A-1}\}$ of size A the matrix θ is of size $A \times (W+1)$. The value at position (i,j), for $0 \le i \le A-1$, $0 \le j \le W$, of the matrix is defined as follows:

$$\theta_{i,j} = \begin{cases} \text{Probability of } x_i \text{ appearing at position } j \text{ of the motif} & \text{if } 1 \le j \le W \\ \text{Probability of } x_i \text{ appearing at positions outside the motif} & \text{if } j = 0 \end{cases}$$

The overall goal of the EM approach is to find a matrix with maximal posterior probability given a set of input sequences.

The outline of the MEME (Multiple EM for Motif Elicitation) [2] algorithm is shown in Figure 1. The search for a motif at each possible motif width W consists of two phases. Since EM is easily trapped in local minima, the first phase iterates over a large number of possible starting points to identify a good initial model $\theta^{(0)}$. In the second phase, the algorithm then performs the full EM algorithm until convergence using $\theta^{(0)}$. Profiling of the MEME algorithm (see Table 1) reveals that over 96% of the overall running time is usually spent on the first phase (called "*starting point search*"). We therefore describe the starting point search algorithm in more detail in the following.

```
procedure MEME(X:set of sequences)
  for pass = 1 to num_motifs do
    for W = W_min to W_max do
      for all starting points (i,j) in X do
        estimate score of the initial motif model which includes the
        W-length substring starting at position j in sequence i;
      end
      choose initial model θ(0) from starting position with maximal
      estimated score;
      run EM to convergence starting with model θ(0);
    end
    print converged model with highest likelihood;
    "erase" appearance of discovered shared motif in X;
  end
end
```

Fig. 1. Outline of the MEME algorithm

Given is the input dataset $X = \{S_1, S_2, \ldots, S_n\}$ consisting of n sequences over the alphabet $\Sigma = \{x_0, \ldots, x_{A-1}\}$ and the motif width W. Let sequence S_i be of length $L(i)$ for $1 \le i \le n$. Then, the total number of substrings of length W in X is $\left(\sum_{i=1}^{n} L(i)\right) - N \cdot (W-1) \cdot$ Let $S_{i,j}$ denote the substring of length W starting at position j in sequence S_i for all $1 \le i \le n$, $1 \le j \le L(i)$, The starting point search algorithm considers all these substrings as possible starting points using the three steps shown in Figure 2.

```
for each length-W substring S_{i,j} in X do
  [Step 1] Compare S_{i,j} to all other length-W substrings S_{k,l} to
           calculate the score P(S_{k,l}, S_{i,j});
  [Step 2] For each sequence k determine the substring S_{k,maxk} where
           maxk = argmax_{1<=l<=L(k)-W+1}{P(S_{k,l}, S_{i,j})};
  [Step 3] Sort and align the identified N substrings in Step 2 to
           determine the estimated score of starting point (i,j);
end
```

Fig. 2. Starting point search algorithm

Table 1. Percentage of MEME (version 3.5.4) execution time spent on starting point search for data sets of various sizes

Dataset	Number of sequences	Average sequence length	Runtime using default parameters	Percentage spent on "starting point search"
Mini-drosoph	4	124,824	15,642 sec	99.4%
Hs_100	100	5000	16,017 sec	96.3%
Hs_200	200	5000	60,142 sec	97.5%
Hs_400	400	5000	233,228 sec	98.7%

In practice, W can be considered to be much smaller than the sequence lengths. Therefore, we assume that W is a constant and determine the time complexities of the three steps in Figure 2 as shown in Table 2. The score between two substrings of length W in Step 1 is calculated using Equation (1). In Equation (1) $S_i[j]$ denotes the letter occurring at position j of sequence i and *map* is a letter frequency matrix of size $A \times A$.

$$P(S_{k,l}, S_{i,j}) = \sum_{r=0}^{W-1} map(S_k[l+r], S_i[j+r]) \quad (1)$$

Table 2. Time complexities of the three steps in Figure 2

Step	Computational requirement	Time complexity
1	Requires an all-against-all comparison of length-W substrings in X	$O\left(\left(\sum_{i=1}^{n} L(i)\right)^2\right)$
2	Requires a linear search of all scores computed in Step 1	$O\left(\left(\sum_{i=1}^{n} L(i)\right)^2\right)$
3	Only deals with the n maximum scores identified in Step 2 and therefore has a lower overall complexity	$O\left(n \cdot \sum_{i=1}^{n} L(i)\right)$

2.2 General Purpose Computations on GPUs

In the past few years, the fast increasing power of the GPU (Graphics Processing Unit) has made it a compelling platform for computationally demanding tasks in a wide variety of application domains. Currently, the peak performance of state-of-the-art consumer graphics cards is more than ten times faster than that of comparable CPUs. Furthermore, GPU performance has been increasing from two to two-and-a-half times a year. This growth rate is faster than Moore's law as it applies to CPUs, which corresponds to about one-and-half times a year. The high price/performance ratio, rapid increase in performance, and widespread availability of GPUs has propelled them to the forefront of high performance computing.

Recently, NVIDIA has released the multi-threaded CUDA programming interface for GPU programming. However, CUDA was not available at the time of writing our GPU-MEME code. Therefore, the presented GPU-MEME algorithm is implemented using the graphics-based GLSL language [4]. Computation using GLSL on a GPU follows a fixed order of processing stages, called the graphics pipeline (see Figure 3). The streaming pipeline consists of three stages: vertex processing, rasterization and fragment processing. The vertex processing stage transforms three-dimensional vertex world coordinates into two-dimensional vertex screen coordinates. The rasterizer then converts the geometric vertex representation into an image fragment representation. Finally, the fragment processor forms a color for each pixel by reading texels from the texture memory. In order to meet the ever-increasing performance requirements set by the gaming industry, modern GPUs support programmability of the vertex and fragment processors using two types of parallelism. Firstly, multiple processors work on the vertex and fragment processing stage, i.e. they operate on different vertices and fragments in parallel. Secondly, operations on 4-dimensional vectors (the four channels Red/Green/Blue/Alpha (RGBA)) are natively supported without performance loss.

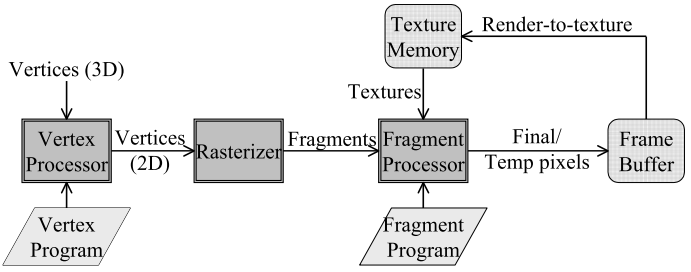


Fig. 3. Graphics pipeline

3 GPU-Accelerated Motif Discovery

3.1 Parallel Streaming Algorithm

The GPU analog of arrays on the CPU are textures. GPUs treat objects as polygon meshes, textures can then be attached to the polygon. Each vertex of the polygon contains texture location information in the form of (x,y) coordinates and the requested texture is interpolated across the polygon surface. This process is called *texture mapping*.

From Step 1 in Figure 2, we can see that for a given length-W substring $S_{i,j}$ the scores $P(S_{k,l},S_{i,j})$ need to be calculated independently from each other for all $1 \le k \le n$ and $1 \le l \le L(k)-W+1$. Our method takes advantage of the fact that all $n\cdot(L(k)-W+1)$ scores can be computed independent of each other. Therefore, we map the sequence dataset (X), the letter frequency matrix (*map*) and the score matrix (i.e. all scores for a fixed (i,j), denoted as: $[P(S_{k,l},S_{i,j})]_{1\le k\le n,1\le l\le L(k)-W+1}$) onto the following three textures:

1) **Sequence dataset texture (Tex_{seq}).** We are using one row of the texture memory to store one sequence. If the sequence length is longer than the row width of the texture, several rows of texture memory will be used. Since the maximum texture size of modern GPUs is 4096×4096 and one texture element can store up to four values (RGBA), a sequence of length L requires $\lceil L/16384 \rceil$ rows of texture memory. In this section, we assume that one sequence fits into one row of texture memory. The partitioning of a sequence onto multiple rows is discussed in Section 3.2.
2) **Letter frequency matrix texture (Tex_{freq}).** This is a relatively small matrix of size $A \times A$. The utilized alphabet for DNA sequences in MEME is Σ = {A, C, G, T, X}, where X represents an unknown nucleotide. Hence, the letter frequency matrix for DNA can be stored in a 5×5 texture.
3) **Score texture (Tex_{score}).** The output of each rendering pass will be written to graphics memory directly, which can then be fed back in as a new stream of texture data for further processing. The dimension of the score matrix texture is equal to the dimension of Tex_{seq}. This allows reusing the coordinates of Tex_{seq} to do lookup operations for Tex_{score}, thus reducing extra coordinate computations. If multiple sequence dataset textures have to be used, the same number of score textures is required to store the rendering results.

Fragment programs are used to implement the arithmetic operations on the above textures specified by Equation (1). Equation (1) requires W table lookups and $W-1$ additions to calculate $P(S_{k,l},S_{i,j})$. The number of operations can be reduced to two lookups and two additions/subtractions by using $P(S_{k,l},S_{i,j})$ to calculate $P(S_{k,l-1},S_{i,j+1})$ as follows:

$$P(S_{k,l},S_{i,j}) = P(S_{k,l-1},S_{i,j}) + map(S_i[j+W],S_k[l+W-1]) - map(S_i[j-1],S_k[l-1]) \quad (2)$$

As shown in Equation (2), during each rendering pass the newly computed score matrix $[P(S_{k,l},S_{i,j})]_{1\le k\le n,1\le l\le L(k)-W+1}$ is stored in the texture memory as a texture. The subsequent rendering pass reads the previous score matrix from the texture memory. Since the calculation of the score matrix $[P(S_{k,l},S_{i,j+1})]_{1\le k\le n,1\le l\le L(k)-W+1}$ depends on the score matrix $[P(S_{k,l},S_{i,j})]_{1\le k\le n,1\le l\le L(k)-W+1,}$ two score matrices have to be stored as separate texture buffers. We are using a cyclic method to swap the buffer function as follows: First, the score matrix $[P(S_{k,l},S_{i,j})]_{1\le k\le n,1\le l\le L(k)-W+1}$ is in the form of a texture input, and $[P(S_{k,l},S_{i,j+1})]_{1\le k\le n,1\le l\le L(k)-W+1}$ is the render target. In the subsequent iteration, $[P(S_{k,l},S_{i,j+1})]_{1\le k\le n,1\le l\le L(k)-W+1}$ is treated as the input texture and $[P(S_{k,l},S_{i,j})]_{1\le k\le n,1\le l\le L(k)-W+1}$ is the render target.

Once $[P(S_{k,l},S_{i,j})]_{1\le k\le n,1\le l\le L(k)-W+1}$ is calculated, the maximum score for each sequence sample has to be found (Step 2 in Figure 2). In order to collect the maximum score for each sequence in texture memory, a preprocessing step eliminates invalid scores in Tex_{score}. These scores will be zeroed in the preprocess step. Thus, they will not influence the final maximum comparison results. This step requires a new texture called Tex_{length}, which stores information about the length of each sequence. After a preprocessing operation, a series of parallel reduction steps are performed on Tex_{score}. Each parallel reduction step consists of two operations. Firstly,

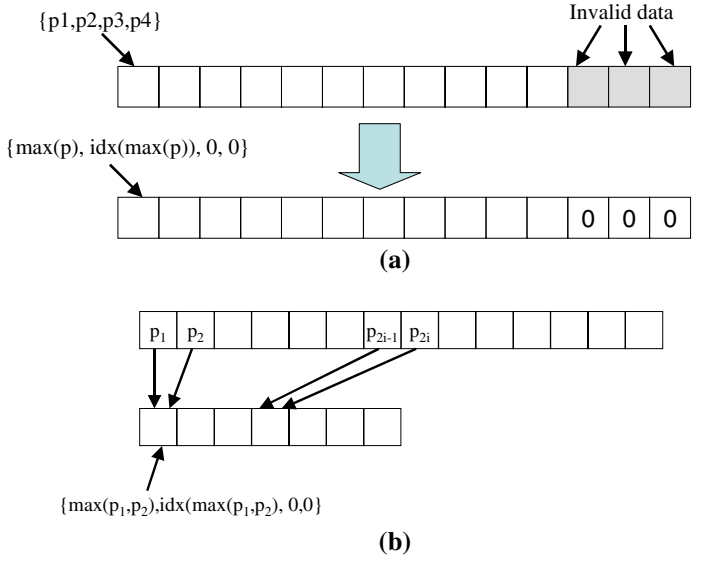


Fig. 4. (a) Preprocessing step; (b) Parallel reduction

all elements with odd indices in the score texture will be compared to their corresponding following elements with even indices. Secondly, an adjustment of texture coordinates is performed. These two operations iterate until the maximum score of each is calculated (see Figure 4(b)). Note that the operations in the first reduction step are slightly different from the following steps. In the first step, each fragment processor compares the four scores in the *R*, *G*, *B* and *A*-channels of a single texture pixel and then outputs the maximum score together with its index into the *R* and *G* channels respectively (see Figure 4(a)). Assuming a maximal sequence length of $L_{\max}$, the number of reduction passes for the maximum computation procedure is therefore $1+\log_2\lceil L_{max}/4 \rceil$.

```
activate, enable and create texture Tex_seq and load sequence data into it;
activate, enable and create texture Tex_freq and load letter frequency
   matrix into it;
activate, enable and create texture Tex_length and load sequence length
   information into it;
enable and create textures Tex_score_j and Tex_score_j+1;
create and initialize a render buffer rBuffer;
for each sequence sample i do
   for each substring j in sequence i do
      set Tex_score_j as render buffer and Tex_score_j+1 as read buffer;
      set texture coordinates Tex_seq[4], Tex_freq[4], Tex_length[4];
      set vertex coordinates vertex[4];
      DrawQuad(Tex_seq, Tex_freq, Tex_length, vertex); /*call kernel program */
      do parallel reduction operation on the score matrix texture to
      get the maximum score for each sequence sample;
      change the functions of Tex_score_j and Tex_score_j+1 in a cyclic way;
      Read back the maximum scores to CPU for further processing;
   end
end
```

Fig. 5. Pseudocode of our streaming algorithm for starting point search

As mentioned in Section 2.1, Step (3) in the starting point search algorithm has a lower time complexity than Steps (1) and (2). Therefore, the produced maximum scores are read back from texture memory to the CPU. The CPU then performs Step (3) sequentially. We will show in Section 4 that the runtime for Step (3) on the CPU is dominated by the runtime for Steps (1) and (2) on the GPU. The pseudocode of our streaming algorithm for starting point search is shown in Figure 5.

3.2 Partitioning and Implementation

So far, we have assumed that each sequence fits into one row of texture memory. In practice, the length of the sequences may be larger and the computation must be partitioned onto several rows. This is incorporated into our streaming algorithm as follows.

1) **Multi-row storage in the sequence dataset texture.** As mentioned in Section 3.1, we are using one row of texture memory to store one sequence. If the sequence length is longer than the row width of the texture, several rows of texture memory will be used. Since the maximum texture size of modern GPUs is

4096×4096 and one texture element can store up to four values (RGBA), a sequence of length L requires $\lceil L/16384 \rceil$ rows of texture memory. Assume L_{max} is the length of the longest sequence in the dataset, in practice we let all long sequences take the same $R_{max} = \lceil L_{max}/16384 \rceil$ rows in the texture memory for simplicity. In this case, a texture can contain $N_{max} = \lfloor 4096/R_{max} \rfloor$ long sequences. Overall, we need $\lceil n/N_{\max} \rceil$ textures to store the complete sequence dataset. Correspondingly, $\lceil n/N_{\max} \rceil$ score textures will be used to store the rendering results.

2) **Multi-row indexing for texture lookups.** If $L_{\max} > 16384$, there exist cases where the letters $S_i[j]$ and $S_i[j+1]$ are stored in different texture rows. In order to handle these cases correctly, we use $(i\%N_{max} + \lceil j/16384 \rceil, j\%16384)$ instead of (i, j) to do texture lookups for $(i, j+1)$.
3) **Multi-row parallel reduction.** According to Section 3.1, $1+\log_2\lceil 16384/4 \rceil$ parallel reduction steps are required to get the maximum in each texture row. Additional $\log_2 R_{max}$ passes are required to get the maximum scores for sequences occupying R_{max} texture rows.

In order to make full use of the computing power in a PC, we have designed and implemented a multi-threaded CPU-GPU collaborative architecture for our streaming algorithm. Figure 6 illustrates the structure of this architecture. It contains three kinds of threads:

1) **Daemon thread:** This thread runs in the background and takes care of the execution of the whole process. It will respond to the data readback operations between the CPU and GPU threads.

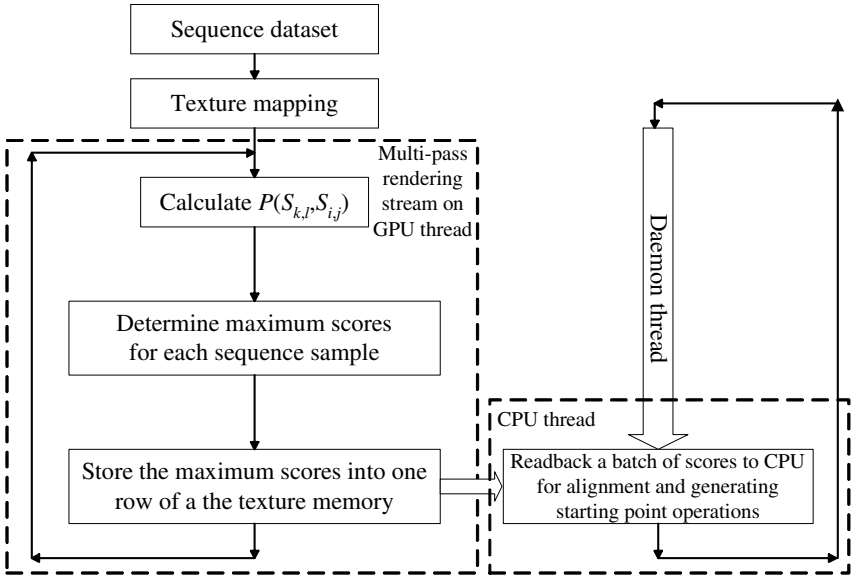


Fig. 6. The structure of our multi-threaded collaborative CPU-GPU architecture

2) **GPU thread:** Because of the implicit data-parallelism processing power of the GPU, it is used to process the compute-intensive calculations. Tasks such as the calculation of $[P(S_{k,l},S_{i,j})]_{1\le k\le n,1\le l\le L(k)-W+1}$ and parallel reduction operations on $[P(S_{k,l},S_{i,j})]_{1\le k\le n,1\le l\le L(k)-W+1}$ are all done by the GPU thread. In order to increase the readback efficiency, the parallel reduction scores during each rendering pass will be first stored in one row of a texture Tex_{max}. After a constant number of rendering passes, a batch of data in Tex_{max} are read back to the CPU for further processing.
3) **CPU thread:** Because of the sequential computing characteristics and the lower time complexity of Step 3 in Figure 2, we let the CPU process this step. When the CPU gets a batch of rendering data from the GPU, it will do the global maximum alignment and starting point generation operations on the data sequentially.

According to our experiments (see Section 4), the GPU thread dominates the runtime. Thus, the runtime of the CPU thread does not influence the overall runtime, since it runs concurrently to the GPU thread.

4 Performance Evaluation

We have implemented the proposed algorithm using C and the GPU programming language *GLSL* (OpenGL Shading Language) [4] and evaluated it on the following graphics card:

- *Nvidia GeForce 8800 GTX*: 1.35 GHz engine clock speed, 900 MHz memory clock speed, 128 stream processors, 768 MB device memory. Tests have been conducted with this card installed in a PC with an Intel Petium4 3.0GHz, 1 GByte RAM running Fedora Core 6 Linux.

A set of performance evaluation tests have been conducted using different numbers of DNA sequences to evaluate the processing time of the GPU implementation versus that of the original MEME implementation. The sequential MEME application is benchmarked on an Intel Pentium4 3GHz processor with 1 Gbyte RAM running Fedora Core 6 Linux. We have used MEME Version 3.5.4, which is available online at http://meme.nbcr.net/meme/intro.html for our evaluation.

The evaluated datasets are the largest dataset supplied by MEME (called minidrosoph) and three datasets of human promoter regions consisting of 100, 200, and 400 sequences of lengths 5,000 base-pairs each (called HS_5000_100, HS_5000_200, HS_5000_400). We have used MEME's default parameters for evaluation. The results for our experiments are shown in Table 3. The CPU alignment part (rows shaded in gray) and the computations on the GPU run concurrently. Since the CPU alignment requires less time, its runtime does not influence the overall runtime. From Table 3 we can see that our GPU implementation achieves speedups of almost fourteen compared to the starting point search stage in MEME and twelve compared to the overall runtime.

Table 3. Comparison of runtimes (in seconds) and speedups of MEME running on a single Pentium4 3GHz to our GPU-accelerated version running on a Pentium4 3GHz with an Nvidia GeForce 8800 GTX for different datasets. The time and percentage spend on different parts of the algorithm are also reported.

Dataset Name, Number of sequences (average length)			HS_5000_100, 100 (5,000)		HS_5000_200, 200 (5,000)	
MEME (P4, 3GHz)	Overall		16017	[100.0%]	60142	[100.0%]
	Starting Point Search		15428	[96.3%]	58656	[97.5%]
	EM		589	[3.7%]	1486	[2.5%]
GPU-MEME (GeForce 8800 GTX)	Overall		1755	[100.0%]	5894	[100.0%]
	Starting Point Search	Score Comp. (GPU)	923	[52.6%]	3565	[60.5%]
		Parallel Red. (GPU)	182	[10.4%]	707	[12.0%]
		Result Readb. (GPU)	61	[3.5%]	136	[2.3%]
		Alignment (CPU)	*1042*	*[59.4%]*	*2045*	*[34.7%]*
	EM (CPU)		589	[33.6%]	1486	[25.2%]
Speedup	Overall		9.1		10.2	
	Starting Point Search		13.2		13.3	

Dataset Name, Number of sequences (average length)			HS_5000_400, 400 (5,000)		Mini-drosoph, 4 (124,824)	
MEME (P4, 3GHz)	Overall		233228	[100.0%]	15642	[100.0%]
	Starting Point Search		230283	[98.7%]	15545	[99.4%]
	EM		2945	[1.3%]	97	[0.6%]
GPU-MEME (GeForce 8800 GTX)	Overall		19895	[100.0%]	1375	[100.0%]
	Starting Point Search	Score Comp. (GPU)	13818	[69.5%]	1061	[77.2%]
		Parallel Red. (GPU)	2764	[13.9%]	209	[15.2%]
		Result Readb. (GPU)	368	[1.8%]	8	[0.6%]
		Alignment (CPU)	*4067*	*[20.4%]*	*244*	*[17.7%]*
	EM		2945	[14.8%]	97	[7.1%]
Speedup	Overall		11.7		11.4	
	Starting Point Search		13.6		12.6	

We have also compared our speedups to the MPI-based ParaMEME implementation ([3], available online at http://meme.nbcr.net/meme/intro.html) on a CPU cluster. The utilized cluster is a 6-node Intel Xeon Dual-Processor cluster with a 1GBit/sec Myrinet switch running Red Hat Linux 3.2.3-24. Table 4 shows a comparison of speedups achieved with ParaMEME compared to our GPU-MEME implementation. As can be seen, our implementation on a single GPU is comparable to the MPI approach on a cluster with 12 CPUs.

Table 4. Speedups of GPU-MEME on a single GPU and ParaMEME on a 12-CPU cluster

Dataset Name	Speedup GPU-MEME	Speedup ParaMEME
Mini-drosoph	11.4	12.6
HS_5000_100	9.1	11.4
HS_5000_200	10.2	11.2
HS_5000_400	11.7	11.1

Table 5. Comparison of the runtime and speedup of MEME running on a P4 3GHz to GPU-MEME running on a cluster with GeForce 8800 GTX cards. Speedup is compared to the sequential MEME code and denoted as "*speedup CPU*". Efficiency with respect to the number of utilized GPUs is denoted as "*efficiency GPU*".

	Mini-drosoph			*HS_5000_100*		
	runtime (sec.)	speedup CPU	efficiency GPU	runtime (sec.)	speedup CPU	efficiency GPU
Seq. MEME	15,642	1.0	N.A.	16,017	1	N.A.
GPU-MEME (**1**×8800GTX)	1,375	11.4	100.0%	1,755	9.1	100.0%
GPU-MEME-MPI (**2**×8800GTX)	760	20.6	90.5%	1,065	15.0	82.5%
GPU-MEME-MPI (**4**×8800GTX)	383	40.8	89.8%	538	29.8	81.5%
GPU-MEME-MPI (**6**×8800GTX)	260	60.2	88.2%	368	43.5	79.5%

	HS_5000_200			*HS_5000_400*		
	runtime (sec.)	speedup CPU	efficiency GPU	runtime (sec.)	speedup CPU	efficiency GPU
Seq. MEME	60,142	1.0	N.A.	233,228	1	N.A.
GPU-MEME (**1**×8800GTX)	5,894	10.2	100.0%	19,895	11.7	100.0%
GPU-MEME-MPI (**2**×8800GTX)	3,394	17.7	87.0%	11,107	20.1	89.5%
GPU-MEME-MPI (**4**×8800GTX)	1,699	35.4	86.8%	5,521	42.2	90.0%
GPU-MEME-MPI (**6**×8800GTX)	1,168	51.5	84.0%	3,817	61.1	86.8%

In order to achieve an even higher speedup, we have extended our GPU-MEME approach to a GPU cluster using MPI. The coarse-grained MPI parallelization assigns to each processor an approximately equal number of starting points to be compared to the input sequence dataset. Table 5 shows a comparison of runtime and speedups of the GPU-MEME cluster version for up to six GPUs compared to the sequential MEME implementation and to our GPU-MEME implementation on a single GPU.

5 Conclusion

In this paper, we have introduced a streaming algorithm for motif finding in biological sequences that can be efficiently implemented on modern graphics hardware. The

design is based on data-parallel computing characteristics in the motif finding process and makes full use of the available computing power in a PC. Our implementation achieves speedups of over an order of magnitude compared to the widely used MEME tool. At least the same number of CPUs connected by a fast switch is required to achieve a similar speedup using the MPI-based ParaMEME code. A comparison of these two parallelization approaches shows that graphics hardware acceleration is superior in terms of price/performance. The presented GPU software is a proof-of-concept parallelization and can be used for the OOPS and ZOOPS search models. Our future work will include integrating the TCM model into our GPU framework and making the software available for public use. We are also planning to port the presented GLSL code to the newly released CUDA programming interface for GPU programming.

Acknowledgement

We thank Geir Kjetil Sandve for the sequence datasets. The work was supported by the A*Star BMRC research grant No. 04/1/22/19/375.

References

1. Bailey, T.L., Elkan, C.: Unsupervised learning of multiple motifs in biopolymers using expectation maximization. Machine Learning 21, 51–80 (1995)
2. Bailey, T.L., Williams, N., Misleh, C., Li, W.W.: MEME: discovering and analyzing DNA and protein motifs. Nucleic Acid Research 34, W369–W373 (2006)
3. Grundy, W.N., Bailey, T.L., Elkan, C.P.: ParaMEME: A parallel implementation and a web interface for a DNA and protein motif discovery tool. Computer Applications in the Biological Sciences (CABIOS) 12, 303–310 (1996)
4. Kessenich, J., Baldwin, D., Rost, R.: The OpenGL Shading Language, Document Revision 8 (2006), http://www.opengl.org/documentation/glsl/
5. Lawrence, C., Altschul, S., Boguski, M., Liu, J., Neuwald, A., Wootton, J.: Detecting subtle sequence signals: a Gibbs sampling strategy for multiple alignment. Science 262, 208–214 (1993)
6. Liu, W., Schmidt, B., Voss, G., Muller-Wittig, W.: Streaming Algorithms for Biological Sequence Alignment on GPUs. IEEE Transactions on Parallel and Distributed Systems 18(10), 1270–1281 (2007)
7. Manavski, S.A., Valle, G.: CUDA compatible GPU cards as efficient hardware accelerators for Smith-Waterman sequence alignment. BMC Bioinformatics 9(Suppl. 2), S10 (2008)
8. Sabatti, C., Rohlin, L., Lange, K., Liao, J.C.: Vocabulon: a dictionary model approach for reconstruction and localization of transcription factor binding sites. Bioinformatics 21(7), 922–931 (2005)
9. Sandve, G.K., Nedland, M., Syrstad, B., Eidsheim, L.A., Abul, O., Drablas, F.: Accelerating motif discovery: Motif matching on parallel hardware. In: Bücher, P., Moret, B.M.E. (eds.) WABI 2006. LNCS (LNBI), vol. 4175, pp. 197–206. Springer, Heidelberg (2006)
10. Schatz, M.C., Trapnell, C., Delcher, A.L., Varshney, A.: High-throughput sequence alignment using Graphics Processing Units. BMC Bioinformatics 8(474) (2007)
11. Sumazin, P., et al.: DWE: Discriminating Word Enumerator. Bioinformatics 21(1), 31038 (2005)

Accelerating BLASTP on the Cell Broadband Engine

Huiliang Zhang, Bertil Schmidt, and Wolfgang Müller-Wittig

School of Computer Engineering, Nanyang Technological University, Singapore
{hlzhang,asbschmidt,askwmwittig}@ntu.edu.sg

Abstract. The enormous growth of biological sequence databases has caused bioinformatics to be rapidly moving towards a data-intensive, computational science. As a result, the computational power needed by bioinformatics applications is growing rapidly as well. The recent emergence of low cost parallel accelerator technologies has made it possible to reduce execution times of many bioinformatics applications. In this paper, we demonstrate how the PlayStation®3, powered by the Cell Broadband Engine, can be used as an efficient computational platform to accelerate the popular BLASTP algorithm.

1 Introduction

Scanning genomic sequence databases is a common and often repeated task in molecular biology. The scan operation consists of finding similarities between a particular query sequence and all sequences of a bank. Dynamic programming (DP) based alignment algorithms whose complexities are quadratic with respect to the length of the sequences can detect similarities between the query sequence and a subject sequence [13]. One frequently used approach to speed up this prohibitively time consuming operation is to introduce heuristics in the search algorithm [2, 7, 9]. Among these heuristics, BLAST (the *B*asic *L*ocal *A*lignment *S*earch *T*ool [1, 2]) is the most popular software. It is used to run millions of queries each day. However, evaluating a single query to a large database with BLAST usually takes several minutes on a modern workstation. These scan time requirements are likely to become even more severe due to the rapid growth in the size of these databases. Hence, finding fast solutions is of highest importance to research.

In this paper we present a new approach to accelerate BLASTP for scanning protein databases on the Cell Broadband Engine (Cell BE). The Cell BE [6, 8, 11] is a recently introduced single-chip heterogeneous multi-core processor, which has been jointly developed by Sony, Toshiba and IBM. Previous work on parallelizing sequence analysis applications on the Cell BE focused on DP-based algorithms such as Smith-Waterman and HMMer [12,14]. Compared to these highly regular applications, parallelization of BLASTP is more challenging since it consists of a pipeline of computations with different memory and processing requirements.

The rest of the paper is organized as follows. Section 2 highlights features of the Cell BE architecture. An overview of the BLASTP algorithm is given in Section 3. Section 4 presents our parallelization approach on the Cell BE. Performance is evaluated in Section 5. Section 6 concludes the paper.

M. Chetty, A. Ngom, and S. Ahmad (Eds.): PRIB 2008, LNBI 5265, pp. 460–470, 2008.

2 Cell Broadband Engine

The Cell BE [6] is a single-chip heterogeneous multi-core processor. It contains two types of processors: a *PowerPC Processor Element* (PPE) and eight *Synergistic Processor Elements* (SPEs) [8,11]. An integrated high-bandwidth bus called the *Element Interconnect Bus* (EIB) connects the processors and their ports to external memory and I/O devices. A block diagram of the Cell BE is shown in Figure 1. The PPE is a 64-bit PowerPC architecture. It is fully compliant with the 64-bit Power Architecture specification and can run 32-bit and 64-bit operating systems and applications. Each SPE is able to run its own individual application program. It consists of a processor designed for streaming workloads, a local memory, and a globally coherent DMA engine. The SPE implements a Cell-specific set of SIMD instructions. With all eight SPEs active, the Cell BE is capable of a peak performance of around 200 GFlops using single precision floating point arithmetic.

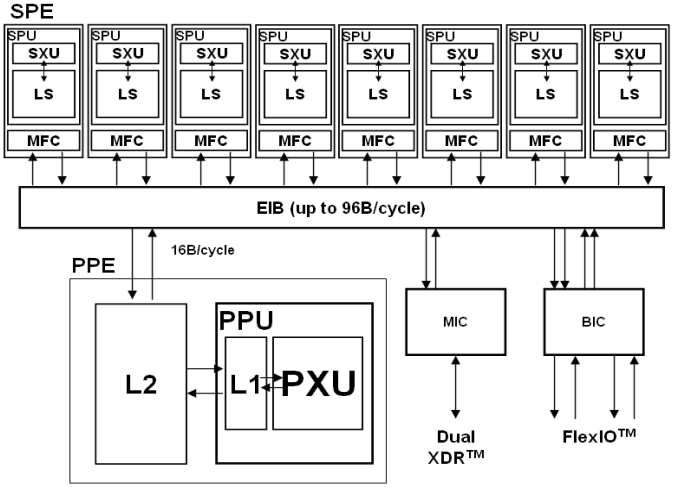


Fig. 1. Block diagram of the Cell BE architecture

Although it is a multiprocessor system on a chip, the Cell BE processor is not a traditional shared-memory multiprocessor. One of the major characteristics is that an SPE can execute programs and directly load and store data only from and to its private *Local Store* (LS). Since SPEs lack shared memory, they must communicate explicitly with the PPE or other SPEs using one of three available communication mechanisms: *DMA transfers*, *mailbox messages*, or *signal-notification messages*. All three communication mechanisms are controlled by the SPE's MFC (Memory Flow Controller).

The design of a parallel algorithm on the Cell BE requires an efficient partitioning of the computation between PPE and SPEs. A general approach is to perform as much as possible computations on the SPEs while the PPE is used for coordinating the control flow. Furthermore, the local memory (LS) of a PPE is very limited (only 256

KB for storing both instructions and data). Therefore, DMA data transfers between main memory and SPEs is often a bottleneck in Cell BE applications and should therefore be minimized.

3 BLASTP Algorithm

The basic idea for fast sequence database search is *filtration*. Filtration assumes that good alignments usually contain short exact matches. Such matches can be quickly computed by using data structures such as lookup tables. Identified matches are then used as seeds for further detailed analysis. The analysis pipeline of the BLASTP algorithm is shown in Figure 2. It consists of four stages. Each stage progressively reduces the search space in the database for significant alignment. We briefly describe each step in the following. More details can be found in [1, 2].

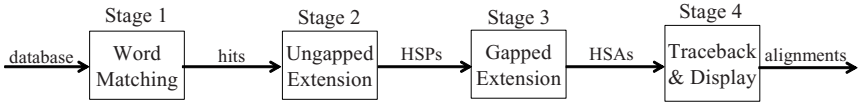


Fig. 2. The BLASTP processing pipeline

Stage 1: This stage identifies *hits*. Each hit is defined as an offset pair (i,j) for which $\sum_{k=0}^{w-1} sbt(Q[i+k],D[j+k]) \geq T$, where *sbt* is a amino acid substitution matrix (e.g. BLOSUM65), w is the user-defined word length, T is a user-defined threshold, Q is the query sequence and D is the database. BLASTP implements this stage by preprocessing Q as follows. For each position i of Q the neighborhood $N(Q[i\ldots i+w-1],T)$ is computed consisting of all w-mers p for which $\sum_{k=0}^{w-1} sbt(Q[i+k],p[k]) \geq T$. The complete neighborhood of a query is typically stored in an efficient data structure such as a lookup table or a finite-state automaton. The default parameter values are w=3 and T=11.

Stage 2: Stage 2 outputs *HSPs* (high-scoring segment pairs) between Q and D. HSPs are identified by performing an ungapped extensions on a diagonal d which contains a non-overlapping hit pair (i_1,j_1), (i_2,j_2) within a window A; i.e. $d = i_1 - j_1 = i_2 - j_2$ and $w \leq i_2 - i_1 \leq A$. If the resulting ungapped alignment scores above a certain threshold it is passed to Stage 3.

Stage 3: This stage outputs *HSAs* (high scoring alignments) between Q and D. HSAs are identified by performing a seeded banded gapped dynamic programming based alignment algorithm using the previously identified HSPs as seeds. Alignments that score above a certain threshold are then passed to the final stage.

Stage 4: The final alignments of the highest scoring sequences are calculated and displayed to the user. This requires the computation of the traceback path using the Smith-Waterman algorithm.

An execution profiling of the BLASTP algorithm for scanning the Genbank non-redundant protein database shows the following breakdown of execution time:

Stage 1: 37%, Stage 2: 31%, Stage 3: 30%, Stage 4: 2%.

Hence, in order to efficiently map BLASTP on the Cell BE all stages except Stage 4 need to be parallelized. Previous work on parallelizing BLASTP has focused on distributed memory architectures such as clusters [10] and reconfigurable hardware [12]. This paper is to our knowledge the first ever reported parallelization of BLASTP on the Cell BE.

4 Parallelizing the BLASTP Algorithm on the Cell BE

In order to achieve an efficient parallelization of map the BLASTP algorithm on the Cell BE we need to address the following challenges.

1. *Limited local storage of the SPE.* A major limitation when designing SPE kernels is that their local memory is only 256 KByte for both instructions and data. Using default parameter for w and T the size of the lookup table used for Stage 1 by NCBI BLASTP is already around 400KByte for 100 randomly selected query sequences. Therefore, we need to use an alternative data structure which requires significantly less memory.
2. *Data transfer and coordination between PPE and SPEs.* The different stages of the BLASTP algorithm constitute a processing pipeline where the throughput of each stage in the pipeline depends on the filtration efficiency of the previous stage. Therefore, an efficient and flexible mechanism to transfer sequences from the database to the SPEs needs to be implemented. The PPE needs to coordinates this data transfer.

Figure 3 shows our mapping of the different stages of the BLASTP algorithm onto the Cell BE. Stage 4 includes a ranking procedure on all database sequences that have passed Stages 1-3: The top 500 or less matching sequences whose scores exceed a certain threshold are displayed in descending order. Thus, this stage is performed by PPE. SPE kernels filter the database as follows. Information about all subject sequences from the database that have passed Stages 1-3 on an SPE are sent to the

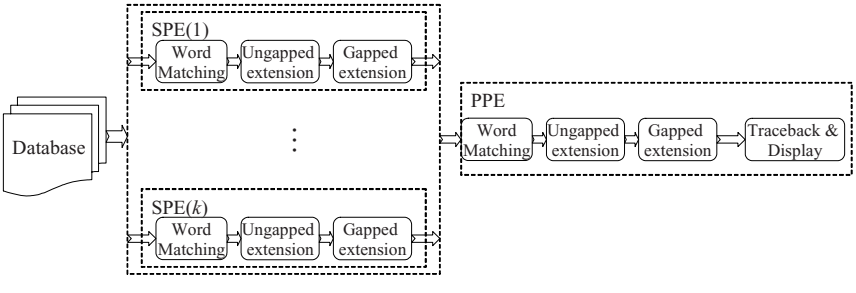


Fig. 3. Mapping of the different stages of the BLASTP algorithm onto the Cell BE

PPE. Upon receiving this information, the PPE completes Stages 1-4 for these subject sequences. The reason why not only Stage 4 is performed on the PPE is that this stage requires additional information from the previous stages and storing this on the SPEs would be too memory-intensive. However, since this redundant computation is merely performed for very few subject sequences the additional runtime is negligible (see Section 5 for details).

As mentioned above, the size of the codeword lookup data structure used by NCBI BLAST is too large for the local store of the SPEs. Therefore, we are using a more memory-efficient data structure for Stage 1. The utilized data structure is a compressed *deterministic finite-state automaton* (DFA), which is similar to the approach used by FSA-BLAST [3, 4]. The compressed DFA for w=3 is illustrated in Figure 4.

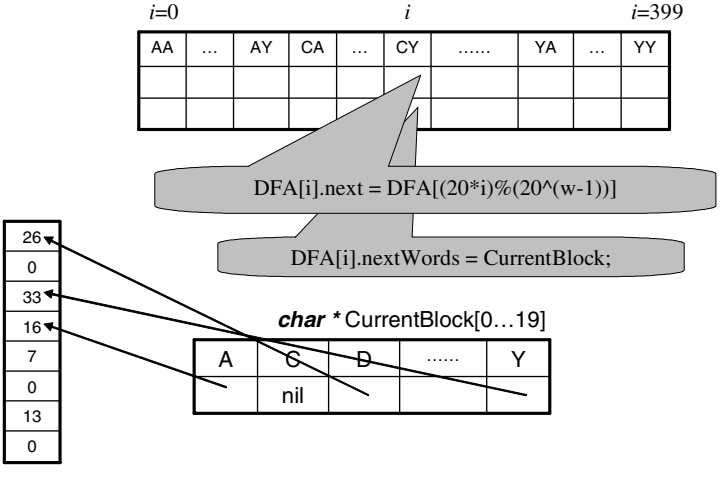


Fig. 4. Illustration of the compressed FSA data structure for w=3

Each possible prefix of lengths $w-1$ is represented by a state; i.e. for w=3 there are 400 states representing the prefixes AA to YY, which are stored in the array DFA[i] in Figure 4. Each state has two transitions: one to the next state (DFA[i].next) and one to a list of 20 words (DFA[i].nextWords). Each entry in this list (currentBlock[0..19]) contains a pointer to an array of query positions. These query positions represent the neighborhood $N(w,T)$ of the associated w-mer. This data structure allows the compression of frequently used query positions that are in neighborhoods of similar w-mers. For example in Figure 4, $N($'CYC',$T) = \{33, 16, 7\}$ and $N=\{$'CYA',$T\} = \{16, 7\}$. By storing these positions in subsequent order terminated by "0" it is possible to re-use memory for both neighborhoods. Our experiments have shown that the size the compressed DFA is only 43.8 KByte on average. Hence, it is possible to store the complete data structure on each SPE for most queries.

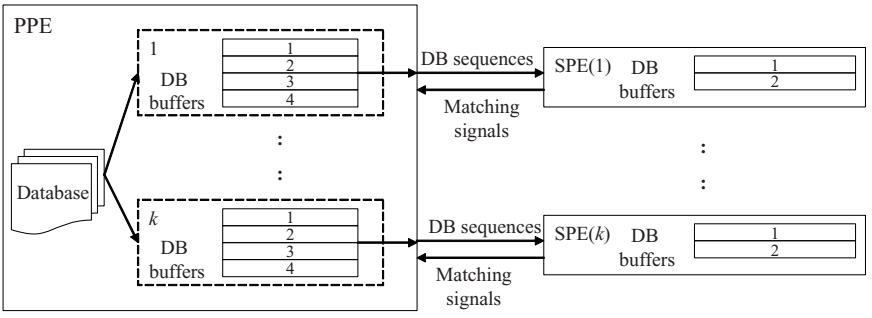


Fig. 5. Buffering scheme

The DFA is transferred into each SPE. The PPE then reads sequences from the database and transfers them to the SPEs by Direct Memory Access (DMA). In order to hide latencies and achieve good load balancing, we have implemented four buffers on the PPE per SPE and two buffers on each SPE (see Figure 5). Our double buffering scheme allows SPEs to receive a new subject sequence through DMA while processing another previously received sequence. The PPE continuously prepares sequence data for free buffers. Once a buffer is filled, the PPE sends a mailbox notification to the corresponding SPE. The number of buffer in the PPE for each SPE is therefore restricted by the size of the SPE's Read Inbound Mailbox (which is four). Furthermore, the PPE dynamically assigns protein sequences to buffers depending on their lengths and the available memory. The maximum number of sequences inside a buffer is 32.

All sequences inside a buffer are filtered by Stages 1-3 on one of the SPEs. If a sequence passes all these stages, the corresponding bit in the matching signal (32 bits) is set. After all sequences are processed, this matching signal is sent back to the PPE via a mailbox. The PPE then identifies all sequences that have passed Stages 1-3 on SPE and perform Stages 1-4 on them.

Pseudocodes of the programs running on the PPE and each SPE are shown in Figures 6 and 7. Because of the limited storage of each SPE (256 KBytes) it is important to analyze the associated memory consumption. The size of SPE program is 100KByte. Thus, we have at most 156KByte for storing the DFA data structure, the two buffers as well as other parameters and intermediate results. Hence, we have assigned 10KByte to each buffer and up to 80KByte to the DFA. 80KByte is sufficient for DFAs for query sequences of up to 2000 base-pairs (bps). In our experiment, the average DFA size is 43.8KByte. If the length of a subject sequence is over 10Kbps, it will be put directly into the sequence queue of the PPE without sending it to an SPE. Furthermore, some database sequences exceed a certain memory threshold during they are processed on the SPE. Such sequences will be marked and passed to the PPE for further processing. Although, this creates additional work, the number of such sequences is usually negligible. It is also another reason why the PPE performs all stages of the BLASTP algorithm instead of only Stage 4. Further note, that we do not return results of matching sequences from SPEs because we do not want to increase SPE code size by increasing program complexity to return the search results.

```
1. Initialization
2. Create DFA
3. Start SPEs and send parameters and DFA lookup table to SPEs
4. Check whether there is mail from SPEs
   If there is a mail
      Collect information of sequences that passed stages 1-3 and keep in a queue
      Mark the corresponding buffer as free
5. Check whether there is a free buffer
   If a free buffer is found
      Prepare data into it and mark it as occupied
   Else
      Do BLASTP searching stages 1-2 for sequences in the queue
6. Repeat 4-5 until there is no sequence in database
7. Send commands to SPEs to complete last buffered sequences
8. Wait until all buffers are marked as free
9. Do BLASTP stages 3-4
```

Fig. 6. Pseudocode of the program running on the PPE

```
1. Initialization
2. Receiving parameters and DFA from PPE
3. Receiving mail with command from PPE
4. If command is new-data-available
      DMA the new data
      If this is the 1st data
         Goto 3
      Else
4.5      Wait for last data to be completely DMA transferred
         Do Stages 1-3 for sequences in the last data
4.7      Return matching signal to PPE through SPU Write Outbound Mailbox
         Goto 3
5. If command is finish-last-sequence
      Do Stages 1-3 for sequences in the last data
      Return matching signal to PPE through SPE Write Outbound Mailbox
      Exit
```

Fig. 7. Pseudocode of the program running on the SPE

5 Performance Evaluation

We have implemented the described Cell BE BLASTP program using CELL BE SDK 3.0 and evaluated it on a PlayStation®3 (PS3), which contains a Cell BE as its main processor. In order to evaluate the performance on a PS3, we have installed LINUX version 2.6.23-rc3 (gcc version 4.1.1 20061011 (Red Hat 4.1.1-30)). Please note that on the PS3 two of eight SPEs are used by the operating system running. Therefore, our experiments can only use up to six SPEs.

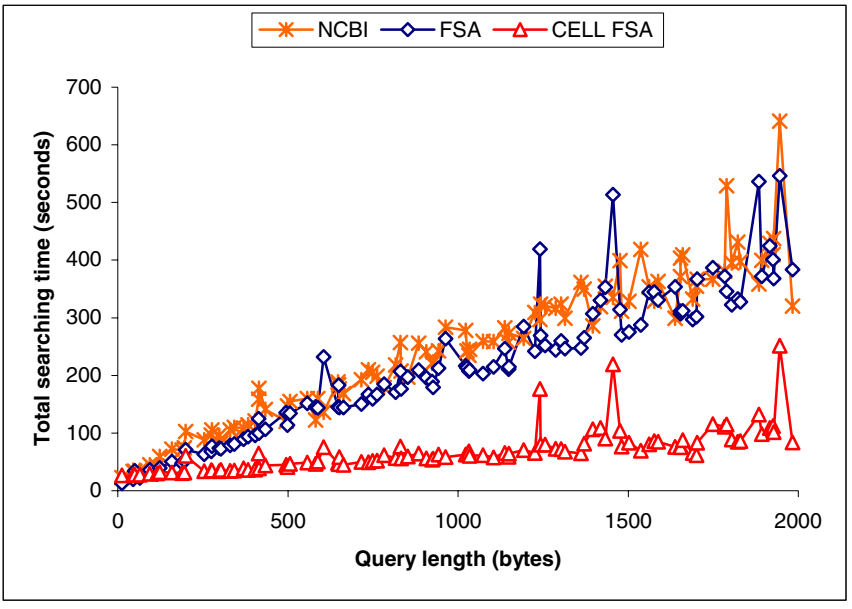


Fig. 8. Runtime comparison between FSA-BLASTP on a dual-core P4 3GHz and Cell BE BLASTP on a PS3 for varying query sequence lengths

We have compared the performance of our Cell BE BLASTP program to FSA-BLASTP (available form www.fsa-blast.org) and NCBI-BLASTP (www.ncbi.nlm.nih.gov/BLAST/developer.shtml). FSA-BLAST uses an optimized sequential algorithm and is around 15% faster than NCBI-BLASTP with no loss in accuracy [3, 4]. FSA-BLASTP and NCBI-BLASTP are tested on a HP workstation xw4200 with Dual-core Pentium®4 (P4) CPU 3GHz, 2GB of RAM. Two-hit model [2] is used for all BLASTP programs. Default values of W=3 and T=11 are adopted. The produced matching results by FSA-BLASTP and Cell BE BLASTP are exactly the same.

The protein sequence database we used in our experiments is the GenBank Non-Redundant Protein Database (downloaded from ftp://ftp.ncbi.nih.gov/blast/db/FASTA/nr.gz), which contains 6,375,605 protein sequences. We have chosen 100 random sequences from the database as queries. The lengths of the query sequences are distributed uniformly between 1 and 2000bps.

A performance comparison of the presented parallel Cell BE BLASTP program to the sequential FSA-BLASTP and NCBI-BLASTP programs are shown in Figure 8. It can be seen that Cell BE BLASTP is faster than FSA-BLAST in most cases. The average searching times are 217.5s for FSA-BLASTP, 244.75s for NCBI-BLASTP, and 67.97s for Cell BE BLASTP. This corresponds to an average speedup of 3.2 and 3.6 respectively.

More detailed statistics are shown in Table 1. From Table 1, we can see that Cell BE BLASTP spends more time on Stage 4. This is because the PPE is a less powerful processor than a P4. The speedup of Cell BE BLASTP mostly comes from stage 1-3 which are running on six PPEs of the PS3 in parallel.

Table 1. More detailed runtime comparison (in seconds) between FSA-BLASTP and Cell BE BALSTP

Query length range	FSA-BLASTP				Cell BE BLASTP				Speedup
	Stages 1-2	Stage3	Stage4	Total	Stages 1-2	Stage 3	Stage 4	Total	
1-300	40.1	5.66	0.30	46.5	28.9	1.77	0.74	32.9	1.41
301-500	74.0	23.09	0.32	97.8	35.4	3.10	0.81	40.9	2.39
501-800	110.3	46.57	0.50	157.8	44.5	4.30	1.10	51.5	3.06
801-1100	151.0	50.98	0.92	203.4	52.8	4.74	1.83	61.1	3.33
1101-1400	183.0	76.32	1.80	261.6	61.8	10.25	4.18	79.0	3.31
1401-1700	216.9	109.01	3.22	329.6	67.2	15.19	7.98	92.4	3.57
1701-2000	241.8	141.53	2.02	385.9	83.9	18.77	4.57	109.0	3.54

Table 2. Average number of sequences processed by each stage of FSA-BLASTP on a P4 and by the PPE in Cell BE BLASTP

Query length	FSA-BLASTP			Cell BE BLASTP (only on PPE)			Matching output
	Stages1-2	Stage3		Stages1-2	Stage3		
		semi	gapped		semi	Gapped	
1-300	6,375,605	96954	9443	2113	2062	1731	328
301-500		334494	13749	2591	2570	1462	324
501-800		617225	19602	5480	5471	3713	443
801-1100		586139	24163	5408	5402	3569	471
1101-1400		761097	34028	7193	7189	5178	443
1401-1700		1096186	43616.1	15404	15402	12901	438
1701-2000		1206705	38761	6734	6733	4126	428

The numbers of sequences that are processed in each stage by FSA-BLASTP and in the PPE by Cell BE BLASTP are shown in Table 2. In FSA-BLASTP, every database sequence is processed by Stages 1-2. The PPE in Cell BE BLASTP only processes a very small faction of database sequences since most sequences have been filtered by SPEs in parallel. This reduced number of sequences contributes to the less total runtime of Cell BE BLASTP. However, the ideal speedup of around six is not reached since the parallel SPE filters add some data transfer and coordination overhead and the PPU is less powerful than a P4. It should also be noted that the speedup for shorter query sequences is generally lower since the runtime is too short to effectively compensate for the associated overheads.

Also note that for Cell BE BLASTP in Table 2, the number of database sequences is larger than the number of found matching sequences. This can be explained as follows. Firstly, if a sequence is too long to be sent to the SPE, it will be processed by the PPE directly. In the experiment, 72 sequences are longer than the maximum buffer length (10KByte). Secondly, some sequences in Stages 1-3 in the SPE exceed the maximum available memory space. These sequences are returned as matches and need further processing on the PPE.

Table 3. Runtime statistics (in seconds) of three exceptional sequences

Query length	Method	Time				
		Stages 1-2	Stage3		Stage4	Total
			semi	gapped		
605	FSA-BLAST	63.55	160.89	6.40	0.80	232.13
	Cell BE	58.65	11.63	1.85	1.88	75.61
1455	FSA-BLAST	138.97	348.58	0.38	24.96	513.43
	Cell BE	84.48	65.49	1.28	66.17	219.43
1945	FSA-BLAST	225.87	316.33	1.02	2.63	546.35
	CELL	132.11	109.53	1.36	6.98	251.52

Query length	Method	Number of sequences			Matching output
		Stages 1-2	Stage3		
			semi	gapped	
605	FSA-BLAST	6,375,605	1,890,358	33,061	500
	Cell BE	5,536	5,536	2,288	500
1455	FSA-BLAST	6,375,605	2,981,242	23,895	500
	Cell BE	8,344	8,344	4,115	500
1945	FSA-BLAST	6,375,605	1,555,474	170,541	500
	Cell BE	27,681	27,677	25,473	500

Figure 8 also shows that some query sequences require more processing time by both FSA-BLASTP and Cell BE BLASTP than queries of similar lengths. The statistics of the three such exceptional sequences is shown in Table 3. It can be seen that for these three queries, a bigger number of database sequences need to be processed than average. This increases both CPU and PPE workload.

6 Conclusion

In this paper, we have presented a parallel algorithm for accelerating BLASTP on a heterogeneous multi-core system. In order to exploit the characteristics of this type of architecture we have used a compressed deterministic finite state automaton for hit detection in order to reduce memory consumption and a double-buffered communication scheme. Our implementation achieves an average speedup of 3.2 compared to the optimized FSA-BLASTP and 3.6 compared to NCBI-BLASTP on a PS3, which is available for less than US$500 at most local computer outlets. The very rapid growth of biological sequence databases demands even more powerful high-performance solutions in the near future. Hence, our results are especially encouraging since high performance computer architectures are developing towards heterogeneous multi-core systems.

Acknowledgement

The work was supported by the A*Star BMRC Research Grant 04/1/22/19/375.

References

1. Altschul, S.F., Gish, W., Miller, W., Myers, E.W., Lipman, D.J.: Basic Local Alignment Search Tool. J. Mol. Biol. 215(3), 403–410 (1990)
2. Altschul, S.F., et al.: Gapped BLAST and PSI-BLAST: A New Generation of Protein Database Search Programs. Nucleic Acid Research 25(7), 3389–3402 (1997)
3. Cameron, M., Williams, H.E., Cannane, A.: A Deterministic Finite Automaton for Faster Protein Hit Detection in BLAST. Journal of Computational Biology 13(4), 965–978 (2006)
4. Cameron, M., Williams, H.E., Cannane, A.: Improved Gapped Alignment in BLAST. IEEE Trans. Computational Biology and Bioinformatics 1(3), 116–129 (2004)
5. Jacob, A., Lancaster, J., Harris, B., Buhler, J., Chamberlain, R.: Mercury BLASTP: Accelerating Protein Sequence Alignment. ACM Transactions on Reconfigurable Technology and Systems(to appear, 2008)
6. Kahle, J.A., et al.: Introduction the Cell multiprocessor. IBM Journal of Research and Development 49(4/5), 598–604 (2005)
7. Kent, W.J.: BLAT – The BLAST-like Alignment Tool. Genome Research 12(4), 656–664 (2002)
8. Kistler, M., Perrone, F., Petrini, F.: Cell multiprocessor communication network: built for speed. IEEE Micro. 26(3), 10–23 (2006)
9. Li, M., Ma, B., Kisman, D., Tromp, J.: Patternhunter II: Highly Sensitive and Fast Homology Search. J. Bioinformatics and Computational Biology 2(3), 417–439 (2004)
10. Oehmen, C., Nieplocha, J.: ScalaBLAST: A scalable implementation of BLAST for high-performance data-intensive bioinformatics analysis. IEEE Transactions on Parallel and Distributed Systems 17(8), 740–749 (2006)
11. Pham, D., et al.: The design and implementation of a first-generation Cell processor. In: Proceedings IEEE ISSCC 2005, San Francisco, CA, pp. 184–185 (2005)
12. Sachdeva, V., Kistler, M., Speight, E., Tzeng, T.H.K.: Exploring the viability of the Cell Broadband Engine for bioinformatics applications. In: 6th IEEE International Workshop on High Performance Computational Biology (HiCOMB 2007), Long Beach, CA (2007)
13. Smith, T.F., Waterman, M.S.: Identification of Common Molecular Subsequences. J. Mol. Biol. 147, 195–197 (1981)
14. Wirawan, A., Kwoh, C.K., Schmidt, B.: Parallel DNA Sequence Alignment on the Cell Broadband Engine. In: Wyrzykowski, R., Dongarra, J., Karczewski, K., Waśniewski, J. (eds.) PPAM 2007. LNCS, vol. 4967. Springer, Heidelberg (to appear, 2008)

Author Index

Acharya, Raj 334
Akutsu, Tatsuya 66
Ammar, Reda 132
Anand, Ashish 154

Bailey, James 165, 400
Balakrishnan, Narayanaswamy 13
Bartlett, Kim 349
Bauer, Denis C. 28
Bedo, Justin 288
Bodén, Mikael 28
Bonilla Huerta, Edmundo 250
Boussioutas, Alex 400
Bulach, Dieter 201
Buske, Fabian A. 28

Carneiro, Laura D.G. 359
Chan, Keith C.C. 225
Chaturvedi, Iti 214
Chen, Chen 448
Chen, Luonan 178
Chetty, Madhu 41, 201, 237, 311
Chomilier, Jacques 54
Coppel, Ross 373

Diniz, Michely C. 359
Dooley, Laurence S. 373
Duval, Béatrice 250

Farias, Kaio M. 359
Fogel, Gary B. 154

Gadelha, Carla R.F. 359
Girão, Karen T. 359
Gondal, Iqbal 373
Gromiha, M. Michael 1
Gustafsson, Daniel 400

Han, Xiaoxu 388
Hao, Jin-Kao 250
Hardison, Ross 334
Haviv, Izhak 400
Heringa, Jaap 187
Hoque, Md Tamjidul 41
Huang, Liang-Tsung 1

Imada, Keisuke 87

Jancura, Pavol 187
Ji, Xiaonan 165

Kamimura, Michel T. 359
Karuturi, R. Krishna Murthy 323
Kasturi, Jyotsna 334
Kato, Yuki 66
Koizumi, Satoshi 87
Kokol, Peter 121
Kowalczyk, Adam 400
Küçükural, Alper 412
Kumar, Chetan 13
Kumar, Nishith 13

Lacroix, Zoé 54
Lai, Lien-Fu 1
Lewis, Andrew 41
Li, QiPeng 78
Lonquety, Mathieu 54
Luo, Huaien 323

Macintyre, Geoff 400
Maia, Italo M.C. 359
Marchiori, Elena 187
McGarry, Ken 349
Medvés, Lehel 110
Meydan, Cem 412
Milanova, Mariofanna 299
Miller, Lance D. 323
Müller-Wittig, Wolfgang 448, 460
Mundra, Piyushkumar A. 144

Nagarajan, Radhakrishnan 299
Ngom, Alioune 262

Ohkawa, Takenao 87
Oliveira, Diana M. 359
Oliveira, Fátima C.E. 359
Ooi, Chia Huey 311
Ott, Michael 424
Ozaki, Tomonobu 87

Pacheco, Ana C.L. 359
Pan, Quan 78

Pfeiffer, Gerd 436
Pittelkow, Yvonne E. 276
Pourfarzam, Morteza 349
Pugalenthi, Ganesan 154

Rajapakse, Jagath C. 144, 214
Rajasekaran, Sanguthevar 132
Ram, Ramesh 201, 237
Ramamohanarao, Kotagiri 165
Rangarajan, Sarani 13
Rodriguez, Juan J. 121
Rueda, Luis 262

Sattar, Abdul 41
Scazzero, Joseph 388
Schimmler, Manfred 436
Schmidt, Bertil 448, 460
Schröder, Jan 436
Seesi, Sahar Al 132
Sehgal, Muhammad Shoaib B. 373
Sekar, Kanagaraj 13
Seki, Hiroyuki 66
Sezerman, O. Uğur 412
Silva, Maria C. 359
Silva, Samara C. 359
Stamatakis, Alexandros 424
Stiglic, Gregor 121
Sudibyo, Yuliansa 323
Suganthan, P.N. 154
Szilágyi, László 110
Szilágyi, Sándor M. 110

Teng, Shyh Wei 311

Upreti, Meenakshi 299

Wang, Lili 262
Wang, Zhenghua 225
Weiguo, Liu 448
Wienbrandt, Lars 436
Wilson, Susan R. 276

Yang, Pengyi 98
Yörükoğlu, Deniz 412

Zhang, Huiliang 460
Zhang, Shao Wu 78
Zhang, Zili 98
Zhao, Xing-Ming 178
Zhou, Tingting 225