Roque Marín
Eva Onaindía
Alberto Bugarín
José Santos (Eds.)

LNAI 4177

Current Topics in Artificial Intelligence

11th Conference of the Spanish Association for Artificial Intelligence, CAEPIA 2005
Santiago de Compostela, Spain, November 2005
Revised Selected Papers

Springer

Lecture Notes in Artificial Intelligence 4177

Edited by J. G. Carbonell and J. Siekmann

Subseries of Lecture Notes in Computer Science

Roque Marín Eva Onaindía
Alberto Bugarín José Santos (Eds.)

Current Topics in Artificial Intelligence

11th Conference of the Spanish Association
for Artificial Intelligence, CAEPIA 2005
Santiago de Compostela, Spain, November 16-18, 2005
Revised Selected Papers

Springer

Series Editors

Jaime G. Carbonell, Carnegie Mellon University, Pittsburgh, PA, USA
Jörg Siekmann, University of Saarland, Saarbrücken, Germany

Volume Editors

Roque Marín
Universidad de Murcia, Campus de Espinardo, s/n.
30071 Murcia, Spain
E-mail: roque@dif.um.es

Eva Onaindía
Universidad Politécnica de Valencia, Camino de Vera, s/n.
46022 Valencia, Spain
E-mail: onaindia@dsic.upv.es

Alberto Bugarín
Universidade de Santiago de Compostela, Monte da Condesa, s/n.
15782 Santiago de Compostela, Spain
E-mail: alberto@dec.usc.es

José Santos
Universidade da Coruña, campus de Elviña, s/n.
15071 A Coruña, Spain
E-mail: santos@uds.es

Library of Congress Control Number: 2006933060

CR Subject Classification (1998): I.2, F.4.1, F.1

LNCS Sublibrary: SL 7 – Artificial Intelligence

ISSN 0302-9743
ISBN-10 3-540-45914-6 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-45914-9 Springer Berlin Heidelberg New York

Springer is a part of Springer Science+Business Media

springer.com

Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 11881216 06/3142 5 4 3 2 1 0

Preface

Since its foundation in 1983, the Spanish Association for Artificial Intelligence (AEPIA) has helped to guide and gather together Spanish researchers working on AI. To this aim, it was agreed in 1985 that AEPIA would promote a national conference every two years. Since then, the AEPIA Conference, known as CAEPIA, has taken place in Madrid, Alicante, Málaga, Murcia, Gijón and Donostia.

Two other foundational objectives of AEPIA are to establish and keep relationships with other national and international organizations in the AI field and to favour the exchange of information and/or experiences among AI researchers. The first objective was accomplished when AEPIA became a member of ECCAI (European Coordinating Committee for Artificial Intelligence) and a founder member of IBERAMIA (Iberoamerican Conference on Artificial Intelligence). The second objective was attained by raising the quality of CAEPIA to usual international standards, focusing on international committees, participants and invited speakers. In 2003, an important step in this direction was taken with the publication of an English volume of selected papers presented at the conference, aiming to boost a more fruitful exchange of ideas within the international AI scientific community.

In November 2005, the 11th Conference of the Spanish Association for Artificial Intelligence (CAEPIA 2005) was hosted by the city of Santiago de Compostela, giving the participants the opportunity of enjoying its superb historical and cultural atmosphere. The conference was jointly organized by the University of Santiago de Compostela and the University of A Coruña. The goals of CAEPIA 2005 were to create conditions for participants to disseminate their research work, to strengthen the relationships among international AI research groups, to facilitate contact between new researchers and already consolidate existing groups, and to help the new developments be spread to society.

The participation rate of CAEPIA 2005 shows the maturity of current research on AI in Spain and the strong links of international collaboration between Spanish and foreign researchers. CAEPIA 2005 received 147 submissions from 17 different countries, with 24.5% of the authors coming from abroad.

The papers were reviewed by an international committee composed of 76 members from 14 different countries. Each paper was sent to three different reviewers, with an average of 2.67 reviews per paper. The best contributions to CAEPIA 2005 were selected for a second review process that produced 48 papers out of 147 (32.7 % of the total), to be published in this post-proceedings volume.

The editors would like to acknowledge the work of the members of the Program Committee in reviewing and selecting the papers. Let us also express our deepest gratitude to the invited lecturers and speakers and to the researchers, who contributed with their valuable work towards achieving a high scientific

level for the conference. The members of the local Organization Committee at the University of Santiago de Compostela and the University of A Coruña, as well as the members of the supporting group for the Program Committee at the University of Murcia also deserve our acknowledgement. Special thanks are due to Springer for publication of this volume and to the staff at Springer for the valuable support they provided throughout the publication process.

The 11th Conference of the Spanish Association for Artificial Intelligence (CAEPIA 2005) took place shortly before the 50th anniversary of what is considered to be the historical birth of the field of AI, the two-month meeting at Dartmouth College during the summer of 1956. This anniversary offers us the opportunity to look back and take a critical glance at the present state of AI in the world. We hope this volume, containing a selection of the best papers presented at CAEPIA 2005, constitutes a small contribution in this direction.

May 2006

Roque Marín
Eva Onaindía
Alberto Bugarín
José Santos

Organization

The 11th Conference of the Spanish Association for Artificial Intelligence (CAEPIA 2005) was jointly organized by the University of Santiago de Compostela and the University of A Coruña, in Santiago de Compostela, Spain, November 16−18, 2005.

Program Committee Chair

Roque Marín, Universidad de Murcia (Spain)

Program Committee Vicechair

Eva Onaindia, Universitat Politècnica de València (Spain)

Program Committee

Carlos Alonso, Universidad de Valladolid (Spain)
Luis Alvarez, Universidad de Las Palmas de Gran Canaria (Spain)
José Angel Bañares, Universidad de Zaragoza (Spain)
Emilia Barakova, Brain Science Institute (Japan)
Federico Barber, Universitat Politècnica de València (Spain)
Alvaro Barreiro, Universidade da Coruña (Spain)
Senén Barro, Universidade de Santiago de Compostela (Spain)
Beatriz Barros, UNED (Spain)
Daniel Borrajo, Universidad Carlos III de Madrid (Spain)
Vicent Botti, Universitat Politècnica de València (Spain)
Alberto Bugarín, Universidade de Santiago de Compostela (Spain)
Nuria Castell, Universitat Politècnica de Catalunya (Spain)
Helder Coelho, Universidade Nova de Lisboa (Portugal)
Ricardo Conejo, Universidad de Málaga (Spain)
Juan M. Corchado, Universidad de Salamanca (Spain)
Ulises Cortés, Universitat Politècnica de Catalunya (Spain)
Juan José del Coz, Universidad de Oviedo (Spain)
Angel P. del Pobil, Universitat Jaume I (Spain)
Alvaro del Val, Universidad Autónoma de Madrid (Spain)
Miguel Delgado, Universidad de Granada (Spain)
Yves Demazeau, Laborative Leibniz, Institut IMAG (France)
Rose Dieng, INRIA Sophia-Antipolis Unit (France)
Ed Durfee, University of Michigan (USA)

Francisco Escolano, Universitat d'Alacant (Spain)
Wolfgang Faber, Università della Calabria (Italy)
Luis Fariñas del Cerro, Université Paul Sabatier (France)
Juan Pedro Febles Rodríguez, Centro Nacional de Bioinformática (Cuba)
Isabel Fernández de Castro, EHU/Universidad del País Vasco (Spain)
José Antonio Gámez, Universidad de Castilla-La Mancha (Spain)
Ana García-Serrano, Universidad Politécnica de Madrid (Spain)
Francisco Garijo, Telefonica I+D (Spain)
Hector Geffner, Universitat Pompeu Fabra (Spain)
Lluís Godo, Institut d'Investigació en Intel.ligència Artificial (Spain)
Antonio Gómez Skarmeta, Universidad de Murcia (Spain)
Asunción Gómez-Pérez, Universidad Politécnica de Madrid (Spain)
Manuel Hermenegildo, Universidad Politécnica de Madrid (Spain)
José Hernández Orallo, Universitat Politècnica de València (Spain)
Miguel Angel Jaramillo, Universidad de Extremadura (Spain)
Elena Lazkano, EHU/Universidad del País Vasco (Spain)
Peter Lucas, Radboud Universiteit Nijmegen (The Netherlands)
Lawrence Mandow, Universidad de Málaga (Spain)
Roque Marín, Universidad de Murcia (Spain)
Fernando Martín, Universidad de Murcia (Spain)
Mark Maybury, MITRE Corporation (USA)
Gaspar Mayor, Universitat de les Illes Balears (Spain)
Erica Melis, Universität des Saarlandes (Germany)
José Mira, UNED (Spain)
Serafín Moral, Universidad de Granada (Spain)
Eduardo Morales Manzanares, ITESM, Morelos (Mexico)
Juan José Moreno Navarro, Universidad Politécnica de Madrid (Spain)
José A. Moreno Pérez, Universidad de La Laguna (Spain)
Pablo Noriega, Institut d'Investigació en Intel.ligència Artificial (Spain)
Nuria Oliver, Microsoft Corporation (USA)
Eva Onaindia, Universitat Politècnica de València (Spain)
Sascha Ossowski, Universidad Rey Juan Carlos (Spain)
Ramón P. Otero, Universidade da Coruña (Spain)
José Tomás Palma, Universidad de Murcia (Spain)
David Pearce, Universidad Rey Juan Carlos (Spain)
Luigi Portinale, Università del Piemonte Orientale (Italy)
Maria Cristina Riff, Universidad Técnica Federico Santa María (Chile)
José Cristóbal Riquelme, Universidad de Sevilla (Spain)
Ramón Rizo, Universitat d'Alacant (Spain)
Jesús María Rodríguez Presedo, Universidade de Santiago de Compostela (Spain)
Camino Rodríguez Vela, Universidad de Oviedo (Spain)
José Santos Reyes, Universidade da Coruña (Spain)
Abdul Sattar, Griffith University (Australia)
Humberto Sossa, Instituto Politécnico Nacional (IPN) (Mexico)

Luis Enrique Súcar, ITESM, Morelos (Mexico)
María Jesús Taboada, Universidade de Santiago de Compostela (Spain)
Miguel Toro, Universidad de Sevilla (Spain)
Maite Urretavizcaya, EHU/Universidad del País Vasco (Spain)
Felisa Verdejo, UNED (Spain)
José M. Vidal, University of South Caroline (USA)
Enrique Vidal, Universitat Politècnica de València (Spain)
Gerson Zaverucha, Universidade Federal do Rio de Janeiro (Brazil)
Changjiu Zhou, Singapore Polytechnic (Singapore)

Conference Co-chairs

Alberto J. Bugarín Diz, Universidade de Santiago de Compostela (Spain)
José Santos Reyes, Universidade da Coruña (Spain)

Additional Reviewers

Eva Armengol Voltas
Roi Blanco González
Alberto J. Bugarín Diz
Manuel Campos Martínez
Purificación Cariñena Amigo
Carlos Carrascosa Casamayor
José M. Casanova Crespo
José Luis Correa Pombo
Carmelo del Valle Sevillano
Félix Díaz Hermida
Vicent Estruch Gregori
Paulo Félix Lamas
Antonio Fernández-Caballero
Manuel Fernández-Delgado
Cèsar Ferri
Pablo García Tahoces
Attilio Giordana
Laura Giordano
Adriana Giret Boggino
Giovambattista Ianni
Roberto Iglesias Rodríguez
José Manuel Juárez Herrero
Vicente Julián Inglada
Manuel Lama Penín
Carlos Linares López
Jim Lipton
Pedro López García
David E. Losada Carril
Antonio Lova
Marco Maratea
Julio Mariño Carballo
Rafael Martínez Gasca
Jesús Medina Moreno
Andreas Meier
Stefania Montani
Manuel Mucientes Molina
Isabel Navarrete Sánchez
Hector Palacios Verdes
Terenziani Paolo
José Luis Pérez de la Cruz Molina
Simona Perri
Jorge Puente Peinador
Jose Miguel Puerta Callejon
María José Ramírez Quintana
Miguel A. Rodríguez González
Horacio Rodriguez Hontoria
Miguel A. Salido
Eduardo M. Sánchez Vila
Laura Sebastiá Tarín
Basilio Sierra Araujo

Giorgio Terracina
Alicia Troncoso Lora
Mercedes Valdés Vela
Carlos Vázquez Regueiro
Xosé A. Vila Sobrino
Alicia Villanueva
Juan A. Botía Blaya
Luis D. Hernández Molinero

Invited Speakers

Gerhard Brewka, Universität Leipzig (Germany)
Peter Lucas, Radboud Universiteit Nijmegen (The Netherlands)
Stephen H. Muggleton, Imperial College London (UK)
Vicente Botti, Universitat Politècnica de València (Spain)

Sponsoring Institutions

Universidade de Santiago de Compostela
Universidade da Coruña
CICYT. Ministerio de Educación y Ciencia
Dirección Xeral I+D. Xunta de Galicia
Dirección Xeral Universidades. Xunta de Galicia

Table of Contents

Invited Talk

Selected Papers from the 11th Conference of the Spanish Association for Artificial Intelligence (CAEPIA 2005)

Preferences in Answer Set Programming

Gerhard Brewka

University of Leipzig, Dept. of Computer Science
Augustusplatz 10-11, 04109 Leipzig, Germany
brewka@informatik.uni-leipzig.de

Abstract. Preferences play a major role in many AI applications. We give a brief overview of methods for adding qualitative preferences to answer set programming, a promising declarative programming paradigm. We show how these methods can be used in a variety of different applications such as configuration, abduction, diagnosis, inconsistency handling and game theory.

1 Introduction

All our decisions depend on preferences. Preferences are just everywhere: an agent may prefer coffee over tea, driving by car over taking the bus, Madonna over Britney Spears, Deportivo La Coruña over Bayern München, staying single over getting married, sleeping over listening to talks, etc. Preferences also play a major role in numerous AI applications, for instance in diagnosis (prefer more plausible hypotheses), configuration (prefer solutions satisfying more important constraints), planning (prefer cheaper/less risky plans), revision (give up less preferred/entrenched beliefs), reasoning about action (prefer histories with fewer unexplained changes), ontologies (prefer more specific information), legal/deontic reasoning (prefer more recent law), and many others.

In dealing with preferences one has to address at least the following issues:

- how to *represent the space of alternatives*: a standard approach to represent alternative choices/solutions to a problem is based on constraints; here we want to advocate the use of answer set programming.
- how to *represent preferences*: traditionally, numbers are used to represent utilities; here we focus on qualitative preferences: numbers are often difficult to obtain from users – and they are not always necessary.
- how to *interpret preferences*: different interpretations of preference statements are possible: preferences can be viewed as strict vs. defeasible, and they can be given a ceteris paribus (other things being equal) interpretation.
- how to *represent (in)dependencies*: in realistic settings, preferences are almost always context dependent; such dependencies need to be represented adequately.

In this paper we describe two related approaches to qualitative preference handling in the context of answer set programming, and we demonstrate how they can be used for several typical AI problems. Let us first describe what we mean by answer set programming – and why we think it is useful in this context.

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 1–10, 2006.

2 Answer Set Programming

Answer sets [13] define the semantics of logic programs with strict and default negation (extended LPs). Such programs are collections of rules of the form:

$$a \leftarrow b_1, \ldots, b_n, not\ c_1, \ldots, not\ c_m \quad (1)$$

where a, b_i, c_j are literals, that is, the classical negation symbol $\neg$ can appear in front of an atom. *not* stands for default negation, $\neg$ for strict negation. We also allow variables in rules, assuming that a rule with variables is just a shorthand notation for all ground instances.

Intuitively, the answer sets of a program represent acceptable sets of beliefs one may adopt based on the information in the program. To qualify as an answer set, a set of literals S has to satisfy two requirements:

- *closedness*: all "applicable" rules must be used to generate S: if for a rule of form (1) all $b_i \in S$ and no $c_j \in S$, then we must have $a \in S$.
- *groundedness*: whenever $a \in S$, a must have a (non-circular) derivation based on undefeated rules only, i.e., rules whose *not* -preconditions are not in S.

To check whether S is an answer set of P one can use the following test[1]

1. remove from P all S-defeated rules (rules with literal *not* L in body such that $L \in S$),
2. remove all *not* literals from the remaining rules,
3. check whether S is the smallest set closed under the reduced program.

Answer sets have proven to be extremely useful for solving a variety of AI problems. Two important developments were essential for this success:

1. the development of highly efficient answer-set provers, the most advanced among them being *Smodels* [18] and *dlv* [10];
2. a shift from a theorem proving to a constraint programming perspective [17],[16].

It turned out that many problems, for instance in reasoning about actions, planning, diagnosis, belief revision and product configuration, have elegant formulations in terms of logic programs if the answer sets (rather than proofs of queries) describe problem solutions [15,24,1,11]. This particular way of using logic programs is now called answer set programming (ASP). Although the language used in rules is very simple, it is quite expressive. For instance, we can represent transitive closure (which is not first order expressible), default rules (like $flies(X) \leftarrow not\ ab(X), bird(X)$), and we can make simple epistemic distinctions which are particularly useful for preference reasoning. For instance, when buying a car it will make a difference whether we know the car is safe (*safe*), or whether we just do not know whether it is not safe (*not* $\neg$*safe*). Such distinctions can conveniently be expressed using the two negations.

[1] This test is actually the definition used in [13].

In ASP it is common to use rules without head to specify constraints to be satisfied by an answer set. The rule ($\leftarrow$ *body*) abbreviates

$$false \leftarrow body, not\ false$$

where *false* is an arbitrary symbol not appearing elsewhere in the program. The effect of this rule is that answer sets satisfying *body* cannot exist.

A simple yet illustrative example is graph coloring. The task is to assign a number of different colors (3 in our example) to the nodes in a graph such that neighboring nodes have different colors. We can represent this as follows (r, g, b stand for red, green, bue, respectively, c for color, *col* for colored):

$$\begin{aligned} col(X, r) &\leftarrow node(X), not\ col(X, b), not\ col(X, g) \\ col(X, b) &\leftarrow node(X), not\ col(X, r), not\ col(X, g) \\ col(X, g) &\leftarrow node(X), not\ col(X, b), not\ col(X, r) \\ &\leftarrow node(X), node(Y), c(Z), col(X, Z), col(Y, Z), edge(X, Y), X \neq Y \end{aligned}$$

Given the description of a graph G, using predicate symbols *node* and *edge*, and a declaration of colors $c(r)$, $c(g)$ and $c(b)$, each answer set describes exactly one legal coloring of G.

3 Qualitative Preferences in ASP

There are many ways of adding preferences to logic programming. Approaches can roughly be categorized based on the following two main criteria:

1. preferences among rules vs. preferences among literals/formulae
2. fixed preferences vs. context dependent preferences

In this paper, we focus on context dependent literal/formula preference. Preferences almost always depend on context. For instance, whether you prefer red wine over white wine will most certainly depend on the main course you are having. Fixed preferences simply are too inflexible to model this. Context dependent rule preferences have been investigated (see for instance [5]). They are somewhat tedious to handle since they require naming techniques to refer to rules in programs. Context dependent preferences on formulas are much simpler to deal with and seem sufficient for a large variety of applications.

We present two related, yet slightly different approaches. The first, logic programs with ordered disjunction (*LPOD*s), is described in more detail in [3,7]. The second approach, answer set optimization programs (*ASO* programs, for short), was introduced in [6]. Both have their merits, the second being more modular and general. However, in some cases the former admits very concise and elegant representations. For this reason we believe they are both interesting.

3.1 LPODs

*LPOD*s are programs using ordered disjunction $\times$ in the head of rules to express preferences among literals in the head: the rule

$$A_1 \times \ldots \times A_n \leftarrow body \tag{2}$$

reads: if *body* is satisfied then some A_i must be in the answer set, most preferably A_1, if this is impossible then A_2, etc. Answer sets are defined through split programs [22]:

Definition 1. *Let r be a rule of form (2). For $k \leq n$ we define the kth option of r, denoted r^k, as*

$$C_k \leftarrow body, not\ C_1, \ldots, not\ C_{k-1}.$$

Definition 2. *Let P be an LPOD. P' is a split program of P if it is obtained from P by replacing each rule in P by one of its options.*

Definition 3. *Let P be an LPOD. A set of literals A is an answer set of P if it is a consistent answer set of a split program P' of P.*

An answer set S can satisfy rules to different degrees, where smaller degrees are better:

Definition 4. *Let S be an answer set of an LPOD P. S satisfies the rule*

$$C_1 \times \ldots \times C_n \leftarrow A_1, \ldots, A_m, not\ B_1, \ldots, not\ B_k$$

- *to degree 1 if $A_j \notin S$, for some j, or $B_i \in S$, for some i,*
- *to degree j $(1 \leq j \leq n)$ if all $A_j \in S$, no $B_i \in S$, and $j = min\{r \mid C_r \in S\}$.*

Intuitively, if *body* is satisfied, then the satisfaction degree is the smallest index i such that $A_i \in S$. Otherwise, the rule is irrelevant. Since there is no reason to blame an answer set for not applying an inapplicable rule, we also define the degree to be 1 in this case. The degree of r in S is denoted $deg_S(r)$.

Based on the satisfaction degrees of single rules, a global preference ordering on answer sets is defined. This can be done through a number of different combination strategies. Let $S^k(P) = \{r \in P \mid deg_S(r) = k\}$. For instance, we can use one of the following conditions to define that S_1 is strictly preferred to S_2:

1. there is a rule satisfied better in S_1 than in S_2, and no rule is satisfied better in S_2 than in S_1 (Pareto);
2. at the smallest degree j such that $S_1^j(P) \neq S_2^j(P)$ we have $S_1^j(P) \supset S_2^j(P)$ (inclusion);
3. at the smallest degree j such that $|S_1^j(P)| \neq |S_2^j(P)|$ we have $|S_1^j(P)| > |S_2^j(P)|$ (cardinality);
4. the sum of the satisfaction degrees of all rules is smaller in S_1 than in S_2 (penalty sum).

Note that S_1 is inclusion preferred to S_2 whenever it is Pareto preferred to S_2, and cardinality preferred to S_2 whenever it is inclusion preferred to S_2.

3.2 ASO Programs

In the *LPOD* approach the construction of answer sets is amalgamated with the expression of preferences. Optimization programs, on the other hand, separate the two consequently to allow for greater modularity, flexibility and generality.

Definition 5. *An* answer-set optimization *(or* ASO*) program over the set of atoms At is a pair* (P_{gen}, P_{pref})*, where* P_{gen} *is a logic program (over At), called a* generator program*, and* P_{pref} *is a collection of preference rules (over At), called a* preference program*.*

A preference rule is of the form

$$C_1 > \ldots > C_n \leftarrow body \tag{3}$$

where the C_i are boolean combinations of literals built from $\wedge, \vee, \neg$ and *not* . A *boolean combination* over a set of atoms A is a formula built of atoms in A by means of disjunction, conjunction, strong ($\neg$) and default (*not*) negation. For simplicity we assume that strong negation is allowed to appear only in front of atoms, and default negation only in front of literals. A satisfaction relation $\models$ can be defined in a straightforward manner:

$S \models l$ (l literal)	iff	$l \in S$
$S \models not\ l$ (l literal)	iff	$l \notin S$
$S \models C_1 \vee C_2$	iff	$S \models C_1$ or $S \models C_2$
$S \models C_1 \wedge C_2$	iff	$S \models C_1$ and $S \models C_2$.

Given this relation we can define the satisfaction degree of a preference rule in an answer set:

Definition 6. *Let S be a set of literals and r a preference rule of form* (3)*. The* satisfaction degree *of r in S (denoted by* $deg_S(r)$*) is given as follows. If* $S \models body$ *and, for some i,* $1 \leq i \leq n$*,* $S \models C_i$*, then* $deg_S(r)$ *is the smallest index of a satisfied boolean combination in the head of r. Otherwise the rule is irrelevant and* $deg_S(r)$ *is denoted by a special symbol I.*

As before we do not want to blame S for irrelevant rules and consider irrelevance as good a satisfaction degree as 1.

Again, the combination strategies described above - and many others - can be used to generate the global preference order. It is also possible to introduce meta preferences among the preference rules themselves by grouping them into subsets with different ranks.

In the rest of this paper, the inclusion-based combination strategy, that is, strategy 2 in the list above, will be our default strategy. As we will see, subset minimality is useful in many cases. Note that inclusion and Pareto preference coincide whenever all preference statements (ordered disjunctions or heads of preference rules) speak about no more than two options.

4 Applications

4.1 Configuration

Configuration is one of the prototypical applications of qualitative preference handling. Configuration problems can often be represented as AND/OR trees,

where AND nodes represent the refinement of a component into several subcomponents, OR nodes represent alternative choices. AND/OR trees can nicely be represented using *Smodels* cardinality constraints [23]: an expression of the form

$$L\{a_1, \ldots, a_n\}U$$

reads: at least L and at most U of the a_is must be true. L and U are integers representing a lower and an upper bound. As an example, consider the following representation of a dinner configuration problem.

$$\begin{aligned} 4\{starter, main, dessert, drink\}4 &\leftarrow dinner \\ 1\{soup, salad\}1 &\leftarrow starter \\ 1\{fish, beef, lasagne\}1 &\leftarrow main \\ 1\{wine, beer\}1 &\leftarrow drink \end{aligned}$$

This corresponds to an AND/OR tree with AND node *dinner* and OR nodes *starter*, *main* and *drink*. If we add the description of the current situation (e.g. $\neg salad$ if salad is unavailable) and preferences, e.g.

$$\begin{aligned} fish \vee beef &> lasagne \\ beer &> wine \leftarrow beef \\ wine &> beer \leftarrow not\ beef \end{aligned}$$

then the maximally preferred answer sets represent optimal configurations.

4.2 Abduction and Diagnosis

Abduction is the generation of explanations for a set of observations. Most formal approaches to abduction are based on a fixed set of formulas H of possible explanations, a set of formulas K representing background knowledge, and a set of formulas O describing the observations to be explained [14,21,19,8]. An explanation then is a minimal subset H' of H satisfying:

1. $H' \cup K$ is consistent and,
2. $H' \cup K \models O$.

In the context of ASP we assume that K is a logic program, H and O are sets of literals, and $\models$ is the credulous inference relation $\models_c$ under answer set semantics, that is, for the program K and observations O we have $K \models_c O$ iff there is an answer set S of K such that $O \subseteq S$ [11].

An explanation for O can be computed via the following LPOD $P_{abd}(K, H, O)$:

$$K \cup \{\leftarrow not\ o \mid o \in O\} \cup \{\neg ass(h) \times ass(h) \mid h \in H\} \cup \{h \leftarrow ass(h) \mid h \in H\}.$$

Intuitively, $ass(h)$ reads: h is assumed. We have the following result [4]:

Proposition 1. *Let H and O be sets of literals and let K be a logic program. H' is an explanation for O given K and H iff there is a consistent maximally preferred answer set S of $P_{abd}(K, H, O)$ such that $H' = \{h \in H \mid ass(h) \in S\}$.*

Note that whenever hypotheses do not interact (no hypothesis can be derived via K and other hypotheses) the literals ass_h are not needed and we can directly use $\neg h \times h$. Here is an example of abductive diagnosis in a medical domain:

$$\begin{array}{ll} H: & measles, flu, migraine \\ O: & headache,\ fever \\ K: & fever \leftarrow measles \\ & red\text{-}spots \leftarrow measles \\ & headache \leftarrow migraine \\ & nausea \leftarrow migraine \\ & fever \leftarrow flu \\ & headache \leftarrow flu \end{array}$$

The program we need to consider is $K \cup P \cup C$ where

$$P = \{\neg measles \times measles, \neg flu \times flu, \neg migraine \times migraine\}$$

and

$$C = \{\leftarrow not\ headache, \leftarrow not\ fever\}.$$

The inclusion preferred diagnoses obtained through this program (by intersecting its maximal preferred answer sets with H) are $\{migraine, measles\}$ and $\{flu\}$.

A somewhat different form of diagnosis, called model based or consistency based, requires a model M describing the normal behavior of the system, using abnormality predicates. A diagnosis is a minimal subset C' of the system components C such that assuming elements of C' to be abnormal and the others to be normal explains the observations. This form of diagnosis can be modeled using *LPOD*s as well [4].

4.3 Inconsistency Handling

In classical logic there are two major ways of handling inconsistency: (1) one can weaken the set of premises, that is, disregard some of the premises entirely or replace them with weaker formulas such that the inconsistency disappears, or (2) one can weaken the inference relation to make sure that inconsistent premises will not lead to the derivation of arbitrary formulas. The former approach is often based on the notion of maximal consistent subsets of the premises, the latter uses paraconsistent logics which are often based on multi-valued logic.

In nonmonotonic reasoning and logic programming there is a third alternative: one can add information to the effect that a rule giving rise to inconsistency is blocked. A simple example is the program $\{p \leftarrow not\ p\}$ which can be made consistent by adding the fact p.

An excellent overview of paraconsistent logic programming is [9]. We will stick here to the other approaches and show how optimization techniques can be used to make sure the effect of weakening or of adding new information is minimized.

Let P be a, possibly inconsistent, logic program. Let R be a set of rules some of which may be added to P to remove inconsistency. Borrowing terminology from

[20] we call them contradiction removal rules (see also [2]). Let $n : P \cup R \rightarrow N$ be a function assigning unique names taken from some set N to each of the rules in $P \cup R$. Technically speaking, the names are just new atoms not appearing in $P \cup R$. We first replace $P \cup R$ by the weakened program $weak_n(P \cup R)$:

$$\{head \leftarrow body, r \mid head \leftarrow body \in P \cup R, n(head \leftarrow body) = r\} \cup$$
$$\{r \leftarrow not\ \neg r \mid r \in N\} \cup$$
$$\{\neg r \leftarrow not\ r \mid r \in N\}$$

The new program is very weak. Basically, the new representation allows us to switch rules in P and in R on and off arbitrarily. Existence of a consistent answer set is guaranteed for the weakened program. Of course, we do not want to consider arbitrary subsets of the original programs. We want to turn off a minimal subset of rules in P and a maximal subset of rules in R. Using optimization programs, this can be achieved by using the set of preference rules:

$$\{r > \neg r \mid r \in N_P\} \cup \{\neg r > r \mid r \in N_R\}.$$

Here, N_P are the names of rules in P, N_R names of rules in R.

Obviously this is still a very simple strategy. Optimization programs offer the possibility to express much more fine grained preferences among the rules to be taken into account. For instance, we may want to distinguish between a domain constraint and an observation made by a potentially unreliable observer. This can conveniently be done by adding additional preference rules. We can also add meta preferences. If we give rules in the first of the two preference rule sets above preference over the rules in the second set, then the contradiction is removed by removal rules alone whenever this is possible. Only if the removal rules are insufficient, information from the original program is disregarded.

4.4 Game Theory

Consider the famous prisoners' dilemma, a 2 player normal form game which is widely discussed in the game theory literature because it shows that a Nash equilibrium is not necessarily Pareto-optimal. Both players can cooperate or defect. The respective utilities of the different outcomes for each player are represented in the following table:

	Coop.	Defect
Coop.	3,3	0,5
Defect	5,0	1,1

The single Nash equilibrium is obtained when both players defect. The pair of utilities in this case is $(1, 1)$. It is not difficult to represent this game as an *LPOD*. We use an index to indicate the respective player and abbreviate cooperate and defect with C and D, respectively. We have to represent the preferences of each player given a particular choice of the other player. Note that the actual numbers do not play a role at all here. The corresponding *LPOD* is as follows:

Player 1:	Player 2:
$D_1 \times C_1 \leftarrow C_2$	$D_2 \times C_2 \leftarrow C_1$
$D_1 \times C_1 \leftarrow D_2$	$D_2 \times C_2 \leftarrow D_1$

In addition, we need a move clause of the form $1\{C_i, D_i\}1$ for $i = 1$ or $i = 2$ which guarantees that an action is taken at all. It is not difficult to verify that the single preferred answer set of this program is exactly the single Nash equilibrium of the game. As a general result, Nash equilibria of normal form games, represented as *LPOD*s in this way, correspond exactly to those answer sets where each rule is satisfied to degree 1. For more details see [12].

5 Conclusions

In this paper we demonstrated that optimization techniques based on qualitative preferences can be highly fruitful for various problem solving tasks. We discussed several such tasks: configuration, abduction and diagnosis, inconsistency handling and game theory. In each case adding qualitative preferences turned out to be very fruitful. We believe that combining current ASP languages constructs for qualitative optimization will greatly increase the impact of ASP on AI and computer science in general.

Acknowledgements

A large part of the work reported here was done in cooperation with I. Niemelä and M. Truszczyński, a cooperation I would not want to miss.

References

1. C. Baral. *Knowledge representation, reasoning and declarative problem solving*, Cambridge University Press, 2003.
2. M. Balduccini, M. Gelfond, Logic programs with consistency-restoring rules, In P. Doherty, J. McCarthy, M.-A. Williams (editors), *International Symposium on Logical Formalization of Commonsense Reasoning*, AAAI 2003 Spring Symposium Series, 9-18, 2003.
3. G. Brewka, Logic programming with ordered disjunction, In *Proc. AAAI-02*, Morgan Kaufmann, 100-105, 2002.
4. G. Brewka, Answer sets and qualitative optimization, *Logic Journal of the IGPL*, to appear 2006.
5. G. Brewka and T. Eiter, Preferred answer sets for extended logic programs, *Artificial Intelligence* 109: 297-356, 1999.
6. G. Brewka, I. Niemelä, and M. Truszczynski, Answer set optimization, *Proc. IJCAI-03*, 867-872, Acapulco, 2003.
7. G. Brewka, I. Niemelä, and T. Syrjänen, Logic programs with ordered disjunction, *Computational Intelligence*, Vol 20 No 2, 335-357, 2004.
8. L. Console, D.T. Dupre, P. Torasso, On the relation between abduction and deduction, *Journal of Logic and Computation* 1(5):661-690, 1991.

9. C. V. Damasio and L. M. Pereira, A survey on paraconsistent semantics for extended logic programas, in: D.M. Gabbay and Ph. Smets (eds.), *Handbook of Defeasible Reasoning and Uncertainty Management Systems*, vol. 2, pp 241-320, Kluwer Academic Publishers, 1998.
10. T. Eiter, N. Leone, C. Mateis, G. Pfeifer, and F. Scarcello. The KR system dlv: progress report, comparisons and benchmarks, *Proc. Principles of Knowledge Representation and Reasoning, KR-98*, Morgan Kaufmann, 86-97, 1998.
11. T. Eiter, W. Faber, N. Leone, G. Pfeifer, The diagnosis frontend of the dlv system, *AI Communications* 12(1-2): 99-111, 1999.
12. N. Foo and T. Meyer and G. Brewka, LPOD answer sets and Nash equilibria, *Proc. 9th Asian Computing Science Conference, ASIAN-04*, 343-351, 2004.
13. M. Gelfond and V. Lifschitz, Classical negation in logic programs and disjunctive databases, *New Generation Computing*, 9:365–385, 1991.
14. K. Konolige, Abduction versus closure in causal theories, *Artificial Intelligence* 53: 255-272, 1992.
15. V. Lifschitz, Answer set programming and plan generation, *Artificial Intelligence*, 138(1-2):39–54, 2002.
16. V. Marek and M. Truszczynski, Stable models and an alternative logic programming paradigm, in: *The Logic Programming Paradigm: a 25-Year Perspective*, Springer Verlag, 375-398, 1999.
17. I. Niemelä, Logic programs with stable model semantics as a constraint programming paradigm, *Annals of Mathematics and Artificial Intelligence*, 25(3,4):241-273, 1999.
18. I. Niemelä and P. Simons, Smodels - an implementation of the stable model and well-founded semantics for normal logic programs, *Proc. 4th Intl. Conference on Logic Programming and Nonmonotonic Reasoning*, Springer Verlag, 420-429, 1997.
19. Y. Peng and J. Reggia, Abductive inference models for diagnostic problem solving, *Symbolic Computation - Artificial Intelligence*, Springer Verlag, 1990.
20. L. M. Pereira, J. J. Alferes and J. Aparicio. Contradiction removal within well founded semantics, in: A. Nerode, W. Marek, V. S. Subrahmanian (editors), *Logic Programming and Nonmonotonic Reasoning*, MIT Press, 105-119, 1991.
21. D. Poole, An architecture for default and abductive reasoning, *Computational Intelligence*, 5(1): 97-110, 1989.
22. C. Sakama and K. Inoue, Prioritized logic programming and its application to commonsense reasoning, *Artificial Intelligence*, 123(1-2):185–222, 2000.
23. P. Simons, I. Niemelä, and T. Soininen, Extending and implementing the stable model semantics, *Artificial Intelligence*, 138(1-2):181–234, 2002.
24. T. Soininen, *An approach to knowledge representation and reasoning for product configuration tasks*, PhD thesis, Helsinki University of Technology, Finland, 2000.

3D Robot Mapping: Combining Active and Non Active Sensors in a Probabilistic Framework*

F. Aznar, M. Sempere, M. Pujol, R. Rizo, and R. Molina

Department of Computer Science and Artificial Intelligence
University of Alicante
{fidel, mireia, mar, rizo, rmolina}@dccia.ua.es

Abstract. Map reconstruction and robot location are two essential problems in the field of robotics and artificial intelligence. A robot could need a model of the environment that can be incomplete and therefore the robot must work considering the uncertainty. Bayesian Units consider the uncertainty and allow the fusion of information from different sensors. In this paper a map reconstruction system in 3D based on Bayesian Units is presented. The reconstruction is carried out integrating the data obtained by a laser and by an omnivision system. In addition, to improve the quality of the reconstruction, the fusion of several Bayesian Units is defined using a competitive fusion operator. Finally, the obtained results as well as the validity of the system are shown.

Keyword: Bayesian Units, Laser/Omnivision Fusion, 3D Mapping.

1 Introduction

Map reconstruction and robot location are essential in robotics and artificial intelligence. They are two research fields widely studied [1]. Robust location of a robot in a given environment depends on several factors. The environment map used by the robot is one of the most important ones [2]. Therefore, map construction has been considered as one of the most important problems for the design of autonomous robots.

However, when a model of the environment is used, we must notice that it can be incomplete due to the existence of hidden variables, not considered in the model, that influence the model. Probabilistic learning and inference try to solve this problem using a formal base. Bayesian programming [3] [4] is a formalism proposed as a solution to deal with problems that entail uncertainty and incompleteness. This formalism has been used successfully for programming autonomous robots [5].

Moreover, the information from different sensors can be fused to increase, among other features, the robustness and the security of the system [6]. In the same way, the consistence of the mapping and the robot localization are improved [7]. Bayesian processing units [8] are a fusion model based on Bayesian programming, oriented to the area of mobile robots.

* This work has been financed by the Generalitat Valenciana project GV04B685.

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 11–20, 2006.

In this paper, a 3D reconstruction system using the information obtained from a laser and from an omnivision system is presented. This map reconstruction has different phases. In the first phase, a SICK LMS 200 laser placed vertically on the top of the robot is used (see figure 1a). The way the laser and omnivision systems are arranged allows us to obtain a section of the environment for each laser capture. Besides, an omnidirectional image is obtained, using a low resolution video camera (640x480) on a hyperbolic mirror, when a laser capture is registered.

Two Bayesian Units have been defined to carry out the basic fusion process of the omnidirectional images and of the laser captures. The first unit task is to obtain the space information from the laser measures. The second unit will joint this information with the omnivision image in order to add the colour information to the generated 3D map. In order to achieve more resolution and robustness in the mapping process, several colour units will be combined to finish the fusion process.

Some examples of obtained maps using this approach will be shown to verify the correctness of the mapping and fusion processes.

2 Bayesian Units and Fusion

A processing unit u is a description that defines the following probabilistic distribution: $P(I \otimes S \otimes O \otimes |u)$, where I is an input variable that specifies the information to be processed, S is an state variable that represents the situation of the process unit and O is an output variable that specifies the new information to be generated. Variables I, S and O do not need to be atomic and can be composed by several random variables that will be assumed discreet.

The decomposition of this probabilistic distribution and its form are not limited. So, the variable decomposition or input variables I can be defined using questions to other processing units.

The state variable S represents the situation of the processing unit. Nevertheless in this paper only reactive units will be used (with $|S| = 0$).

As it was previously specified, u is made up of a description of a set of variables. Therefore, it is simple to ask any question to this distribution.

A processing unit is considered to be functional when it answers in informative way (its entropy is sufficiently separated from its maximum) some questions that can be applied to its distribution. Initially, the probability of a first state S_0 is obtained from the input variables. Using this state, all necessary questions must be answered until the distribution that allows the system to obtain the output variables is defined. In some of this intermediate steps the information about a previous state and an output variable in each previous distribution can be useful to provide more information to the unit. A Bayesian Unit is considered useful, if it answers in an informative way to all the previously defined questions. A Bayesian Unit is described using the Bayesian programming formalism [3]:

$$
\text{UB}\left\{\begin{array}{l}
\text{Description}\left\{\begin{array}{l}
\text{Especification}\left\{\begin{array}{l}
\textbf{Variables}: \\
I: \text{Input data} \\
\text{S: Internal state} \\
\text{O: Output information} \\
\textbf{Decomposition}: \text{Any} \\
\textbf{Parametrical Forms}: \text{Any}
\end{array}\right. \\
\text{Identification: Any}
\end{array}\right. \\
\text{Questions: Any} \\
\quad \text{Need to be informative:} \\
\quad \textbf{if } S = \emptyset \\
\quad \{P(O|I) \\
\quad \textbf{else} \\
\quad \left\{\begin{array}{l}
S = \{S_0, S_1, \ldots, S_n\}, \\
A_i = \{a_{1,i}, a_{2,i}\}; a_{1,i} \subseteq O, a_{2,i} \subseteq S; \\
P(S_0|I), (P(S_1|S_0 \otimes A_0), P(S_2|S_1 \otimes A_1), ..., P(S_n|S_{n-1} \otimes A_{n-1})), \\
P(O|S_n \otimes a_{2,n})
\end{array}\right.
\end{array}\right. \tag{1}
$$

2.1 Competitive Fusion Operator

Although several fusion operators that allow the system to combine Bayesian Units can be defined, in this paper only a competitive fusion operator for reactive units is specified. A configuration of processing units is considered competitive or redundant if each unit obtains independent measures of the same property. Two units u_1 and u_2 with the capability of answering in an appropriate way their associate questions are available. A new unit u that combines the information of both units is the aim of the system. Although the number of states will influence the fusion process of two units, in this case, only the operator capable of fusing two units with no states are presented. In addition, the combination of the outputs of u_1 and u_2 is assumed to be unknown.

Competitive Fusion, Reactive Units, $|S| = 0$. In the case of fusion of reactive units (units with no state), the variables of the new unit u are defined as:

$$
\begin{array}{l}
I = I_1 \wedge I_2, |I| = |I_1| + |I_2| \\
O = O_1 \cup O_2, |O| = |O_1| + |O_2| - |O_1 \cap O_2| \\
S = \emptyset, |S| = 0
\end{array} \tag{2}
$$

So, input variable I will be composed by the inputs of the subunits. As the way to combine the outputs of the subunits in the fusing process is unknown, the output will be composed by all the possible outputs of the subunits. The following decomposition is proposed based on the previous probability distribution:

$$
\begin{array}{l}
P(I \otimes O) = P(I_1 \otimes I_2 \otimes O) = \\
\quad = P(O) \times P\left(I_1 | O\right) \times P\left(I_2 | O \otimes I_1\right) = \\
\quad = P(O) \times P\left(I_1 | O\right) \times P\left(I_2 | O\right)
\end{array} \tag{3}
$$

where the second equality results from the conditional independence between I_1 and I_2. The probability of a certain output $P(O)$ is considered uniform. Moreover, both distributions $P(I_1|O)$ and $P(I_2|O)$ have to be specified by the programmer, so that for each output the inputs that generated it can be determined.

To considerate the definition of the unit to be informative, the question $P(O|I)$ must be answered. The fusion of the units informatively answers this question, although the demonstration will not be included in this paper.

3 Proposed Bayesian Units

3.1 Obtaining Points in the 3D Space

The first stage of mapping is to obtain the set of points that represents the environment where the robot will move. As it is said, a laser is placed vertically and it is capable of obtaining a section of 180º from each position. A set of sections is obtained moving the robot through the environment. Joining these sections an environment description is calculated. Therefore, the Bayesian Unit that obtains the 3D points of the environment will have the following variables:

$$(I; S; O) : (A_L, D_L, P_R; \emptyset; P_W)$$

So, input variables are composed by: the angles of each laser capture A_L, which are known and range from 0º to 180º at intervals of 0.5, the distance measured by the laser D_L for each angle (from 0 to 10m with precision of 10mm) and the current robot position P_R. The state is not needed in this unit because it is a reactive process where an output will be directly generated from an input. The set of points that we want to obtain from input data is P_W.

A joint distribution is generated from these variables and a decomposition of this distribution is proposed:

$$P(A_L \otimes D_L \otimes P_R \otimes P_W) = P(P_R) \times P\,(A_L) \times P\,(D_L|\, A_L \otimes P_R) \times P\,(P_W|\, A_L \otimes D_L \otimes P_R)$$

$P(P_R)$ can be supposed to be uniform if it is not needed the robot to start mapping process at a certain location, as in this case, all the positions that the robot can visit in the environment have the same probability of being visited. In the same way, $P(A_L)$ is considered uniform because all the angles of the laser capture A_L will be used, independently from the position of the robot. Besides, $P(D_L|A_L \otimes P_R)$ is determined by both the laser measures in a certain position and angle, and the laser specifications. In this case, the readings of the LMS 200 laser have a normal error of $err_L = \pm 15mm$. $P(P_W|A_L \otimes D_L \otimes P_R)$ is exactly the question about the joint distribution to be obtained: given the angle and the distance measured by the laser in a certain position, which point in the 3D space does it match with? In equation 4 (where z is forward direction, y is the vertical axis and x is the Horizontal axis) it is shown that a point can be easily obtained applying the appropriate geometric transforms. Nevertheless, it is important to consider that any distance D_L is represented by a normal distribution. This way, for any angle and distance captured by the laser in a certain position, not only a point is obtained, but a Gaussian for each coordinate (x, y, z). This Gaussian will show the set of points that can represent the laser measures and their probability.

$$P_W = (P_{W_X}, P_{W_Y}, P_{Wz}) = \begin{pmatrix} G(\mu(D_L \cdot \cos(A_L) + P_{R_{\mu X}}), \sigma(err_L + P_{R_{\sigma X}})), \\ G(\mu(D_L \cdot \sin(A_L)), \sigma(err_L)), \\ G(\mu(P_{R_{\mu Z}}), \sigma(P_{R_{\sigma Z}})), \end{pmatrix} \tag{4}$$

The floor is assumed to be plane and even, therefore the robot coordinates P_{R_Y} are always 0. The robot position represented by P_{R_μ} and P_{R_σ} is also supposed to be known.

Another process unit can be used in order to locate the robot using previous measures (this unit will not be presented in this paper). Many methods can be developed to obtain this location in both indoor and outdoor (not structured environments) using a laser. An approximation to probabilistic location can be used, such as Kalman filter, in order to estimate the robot position. A Kalman filter can de defined using the same paradigm as Bayesian Units, the Bayesian programming, as it can be seen in [4].

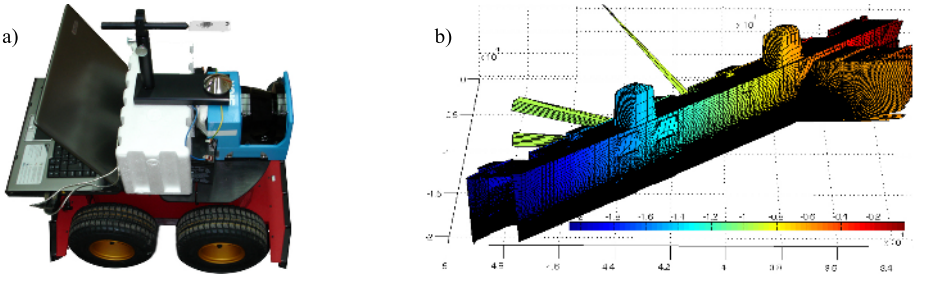


Fig. 1. a) Pioneer Robot P3-AT with laser and omnivision systems. b)Points map obtained by the first Bayesian Unit. It only represents $\mu(P_W)$. Bar line indicates the plane depth. A corridor of the university department is shown. Moving from the origin through the corridor, a room can be seen from 2 to 6m and two office doors with a window on the top of the door at 14m. Two skylights that provide zenithal light to the corridor can be observed.

3.2 Establishment of Point's Colour

When only one image is obtained from a three-dimensional environment, it is important to perform a reconstruction because there is a spatial information loss. Combining an active sensor with the captured images, an easy method to obtain the colour information for a certain point is provided. In this case, an omnivision image that saves the environment colour is jointly captured with each laser reading. The omnidirectional image provides visual information over the 180º and, therefore, for all the points captured by the laser.

A Bayesian processing unit with the capacity for obtaining each point colour, must have the following input variables:

$$(I; O) = (P_W, Omni; P_C)$$

where P_W represents the three-dimensional points obtained by previous unit. $Omni$ is the omnidirectional image and P_C is the colour that belongs to each P_W value. In this way, we can decompose the joint variables as:

$$P(P_W \otimes Omni \otimes P_C) = P(P_W) \times P(Omni|P_W) \times P(P_C|P_W \otimes Omni)$$

Where the probability of a point in the space $P(P_W)$ is uniform because we suppose that we do not have information about the environment structure. $P(Omni|P_W)$ is uniform because a point in the space can have any colour. Moreover, the distribution $P(P_C|P_W \otimes Omni)$ is related with the projection of a point P_W in the hyperbolic mirror used in the generation of the image $Omni$.

$$R: \begin{cases} x = p_x + t(q_x - p_x) \\ y = p_y + t(q_y - p_y) \\ z = p_z + t(q_z - p_z) \end{cases} \qquad H: \begin{cases} \frac{z^2}{c^2} - \frac{x^2+y^2}{a^2} = 1 \\ a^2 = b^2 = 548.1440; \mathrm{c}^2 = 789.3274 \\ \mathrm{q} = \sqrt{\mathrm{a}^2 + b^2} \end{cases}$$

R is the straight line from point p to hyperbole focus q. The projection of p on the hyperbole is defined as the intersection of hyperbole H and the straight line R, as:

$$\mathrm{Proj}_H(p): \begin{cases} \left(a^2(q_z - p_z)^2 - c^2(q_x - p_x)^2 - c^2(q_y - p_y)^2\right) t^2 + \\ \left(a^2 2p_z(q_z - p_z) - c^2 2p_x(q_x - p_x) - c^2 2p_y(q_y - p_y)\right) t + \\ a^2 {p_z}^2 - c^2 p_x^2 - c^2 p_y^2 - a^2 c^2 = 0 \end{cases}$$

Where t is the unknown value to be found obtaining as a result the two points of intersection of the straight line and the hyperbole.

As it is was said before, P_W has no points, but Gaussian distributions. The projection of the Gaussian distribution onto a tangent plane to the hyperbole is approximated. Distribution $P(P_C|P_W \otimes Omni)$ is defined as a two-dimensional Gaussian based on the pixels of image $Omni$.

$$\begin{aligned} P(P_C|P_W \otimes Omni) &= G_{Omni_x, Omni_y}(\mu_1, \mu_2, \sigma_1, \sigma_2, [\rho = 0]) = \\ &= G_{Omni_x, Omni_y}(\mathrm{Proj}_H(P_W)_x, \mathrm{Proj}_H(P_W)_y, \mathrm{Proj}_H(\sigma_x), \mathrm{Proj}_H(\sigma_y)) = \\ &= \frac{e^{\left(-\frac{(x - \mathrm{Proj}_H(P_W)x)^2}{2\mathrm{Proj}_H(\sigma_x)^2} - \frac{(y - \mathrm{Proj}_H(P_W)y)^2}{2\mathrm{Proj}_H(\sigma_y)^2}\right)}}{2\pi \mathrm{Proj}_H(\sigma_x) \cdot \mathrm{Proj}_H(\sigma_y)} \end{aligned}$$

4 Fusion of Colour Processing Units

It can be easily noticed that the Bayesian Unit that establishes the colour of a point in the space, discards a lot of information that can be fused. On one hand, this unit only obtains the colour of the points returned by the current plane of the laser (discarding close readings that can also contribute relevant information). On other hand, for each plane obtained by the laser in a position z, only the relative image $Omni_z$ is used.

To improve the quality of the mapping process it could be advisable to introduce more information about the colour of a point in the space, fusing two colour processing units. This way, applying the equations 2 and 3 we obtain:

$$\begin{aligned} &U: (I_1; I_2; O) = (P_{W_1}, Omni_1; P_{W_2}, Omni_2; P_{C_1} \cup P_{C_2}) \\ &P(I_1 \otimes I_2 \otimes O) = P(O) \times P\left(P_{W_1} \otimes Omni_1 | O\right) \times P\left(P_{W_2} \otimes Omni_2 | O\right) \end{aligned}$$

$P(O)$ is uniform because we assume that all the colours have the same probability. $P\,(P_{W_1} \otimes Omni_1|\,O)$ and $P\,(P_{W_2} \otimes Omni_2|\,O)$ determine the probability that a certain colour represents a point in the space defined by P_W and $Omni$. Nevertheless, using the distribution $P(O|P_W \otimes Omni)$ would be more convenient because it is defined previously. It is easy to prove that:

$$P\,(O|\,P_{W_1} \otimes Omni_1) = \tfrac{1}{\Sigma} \times \sum_{P_{W_2}, Omni_2} \begin{pmatrix} P(O) \times P\,(P_{W_1} \otimes Omni_1|\,O) \times \\ P\,(P_{W_2} \otimes Omni_2|\,O) \end{pmatrix} =$$
$$= \tfrac{1}{\Sigma} \times P(O) \times P\,(P_{W_1} \otimes Omni_1|\,O) \times \sum_{P_{W_2}, Omni_2} P\,(P_{W_2} \otimes Omni_2|\,O) =$$
$$= \tfrac{1}{\Sigma'} \times P\,(P_{W_1} \otimes Omni_1|O)$$

$\frac{1}{\Sigma}$ is a normalization constant. The fusion of two units can be written as:

$$P(I_1 \otimes I_2 \otimes O) = P(O) \times P\,(O|\,P_{W_1} \otimes Omni_1) \times P\,(O|\,P_{W_2} \otimes Omni_2)$$

Where $P\,(O|\,P_{W_1} \otimes Omni_1)$ and $P\,(O|\,P_{W_2} \otimes Omni_2)$ have previously been defined in fused subunits when $[O \subseteq P_{W_1}]$ and $[O \subseteq P_{W_2}]$ and they are considered uniform in other case. So, unit combination is a two-dimensional Gaussian product.

Two kinds of fusion can be carried out using the combination of two units. On one hand, two fused units can share the same image $Omni$ but different points. On the other hand, points con be fixed and images can be changed for each subunit.

5 Experimentation and Results

The map reconstruction process is divided into three steps. In the first step, a set of points that represents the environment is obtained using the information from the laser. As it was said before, it is advisable to consider that the information provided by the laser for a certain angle is not a point, but a Gaussian. This Gaussian indicates the set of points that can represent the laser measures with

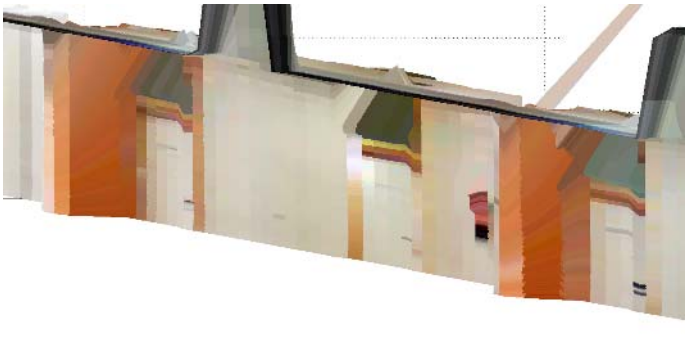


Fig. 2. Point map with colour information P_C obtained by a Bayesian colour processing unit for all map planes. A corridor section is presented, it is only shown the more probable colour of the distribution.

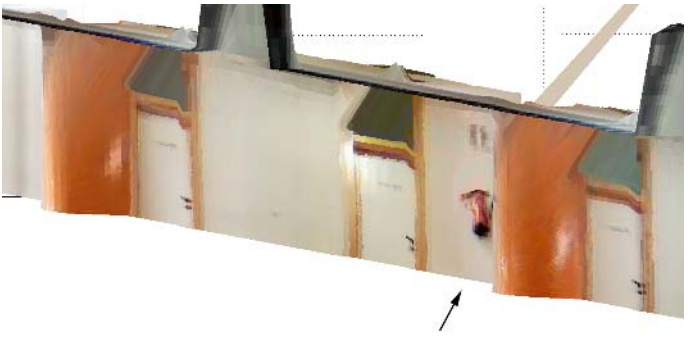


Fig. 3. Colour reconstruction P_C for each plane is obtained fusing two units information. A corridor section with only the more probable colour of the distribution is presented.

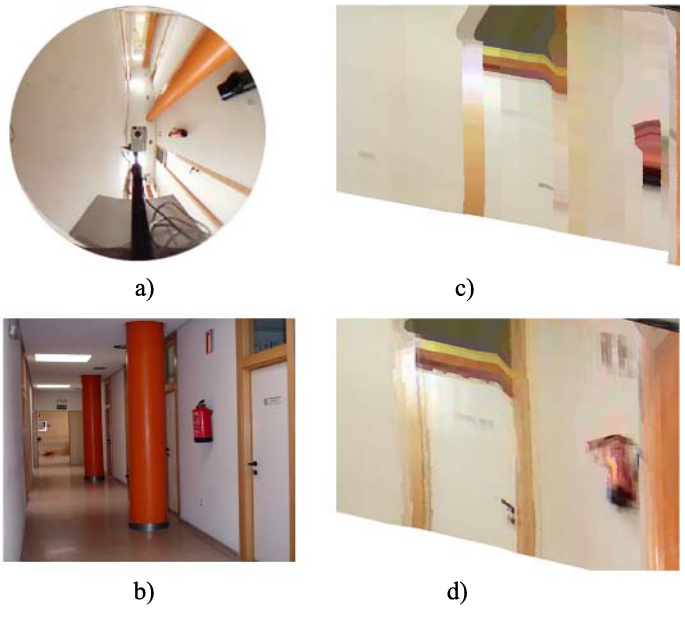


Fig. 4. a) Omnidirectional image obtained in a laser capture. b) Original corridor view. c) Detail of a reconstruction using only one unit. d) Reconstruction fusing two units. The arrow in figure 3 shows the plane where these shots have been obtained.

their probability. In figure 1b a 3D map reconstruction of a corridor of our department is obtained.

In a second step the colour is added. An omnidirectional image is captured for each laser reading (see fig. 4a). For each laser point a colour that represents it in the omnidirectional image is obtained (see fig. 2).

In the previous reconstruction a lot of information that can be fused to increase robustness and precision in the representation is discarded. Fusion can be done

in two ways. On one hand, interpolated planes between the data of plane pl_i and its next plane pl_{i+1} can be obtained. The colour of each point can also be calculated using a Bayesian colour unit. In this case, ten subplanes have been used for the image in figure 3. On the other hand, two units can be used for each interpolation between pl_i and pl_{i+1}, one for the image $Omni_i$ and another for $Omni_{i+1}$. This way, the fusion of these two units provides a more uniform representation.

Nevertheless, a third fusion method can be considered: using several units (each one with a different image $Omni$) for every plane or combine them. This method can introduce some noise in the system and therefore it can decrease the quality of generated images. It is important to consider that the Bayesian colour unit is very related with the projection of a point (x, y, z) in the hyperbole H. So, any odometric error can actually affect the projection of distant planes in a certain omnivision image providing wrong results.

Finally, it is important to highlight the difference between a reconstruction without fusion of Bayesian colour units (figure 4c) and a reconstruction using the fusion methods previously described (figure 4d).

6 Conclusions

In this paper, a model for 3D map reconstruction for an autonomous robot has been presented. The model of the environment used by the robot can be incomplete. Therefore, the robot should consider uncertainty. The proposed system is based on Bayesian Units. A Bayesian Unit is a fusion model, based on Bayesian programming, that explicitly considers uncertainty.

A process to 3D map reconstruction has been presented. This reconstruction process uses a laser and an omnivision system placed on the robot. Map reconstruction process can be divided in three steps, where every step is a Bayesian processing unit. In the first step a 3D point map of the environment is obtained from the laser information and from the robot position. Moreover, from omnidirectional images obtained by an omnivision system, a colour is assigned to every point obtained by the previous unit.

The use of a competitive fusion operator to carry out the fusion of several colour processing units is proposed. This way, we improve the quality of the reconstruction.

Results of the application of the fusion process, to verify this method, have been presented. Future works will be focused on using this maps in robotic localization tasks.

References

1. Thrun, S., Burgard, W., Fox, D.F.: A real-time algorithm formobile robot mapping with applications to multi-robot and 3d mapping. Proceedings of IEEE International Conference on Robotics and Automation (2000)
2. Pérez, J., Castellanos, J., Montiel, J., Neira, J., Tardós, J.: Continuous mobile robot localization: Vision vs. laser. Proceedings of IEEE International Conference on Robotics & Automation (1999)

3. Lebeltel, O., Bessière, P., Diard, J., Mazer, E.: Bayesian robots programming. Autonomous Robots **16** (2004) 49–79
4. Bessière, P., Group, I.R.: Survei: Probabilistic methodology and tecniques for artefact conception and development. INRIA (2003)
5. Julien Diard, P.B., Mazer, E.: A theorical comparison of probabilistic and biomimetic models of mobile robot navigation. Proceedings of the 2004 IEEE, International Conference on Robotics & Automation. New Orleans, LA (April 2004)
6. Elmenreich, W.: Sensor Fusion in Time-Triggered Systems. PhD thesis, Institud Für Technische Informatik 182 (October 2002)
7. Aboshosha, A., Zell, A.: Robust mapping and path planning for indoor robots based on sensor integration of sonar and a 2d laser range finder. Proceedings of IEEE 7th International Conference on Intelligent Engineering Systems (2003)
8. Aznar, F., Sempere, M., Pujol, M., Rizo, R.: A cognitive model for autonomous agents based on Bayesian programming. Proceedings of Brain, Vision and Artificial intelligence BV&AI (2005)

A Flipping Local Search Genetic Algorithm for the Multidimensional 0-1 Knapsack Problem

César L. Alonso[1,⋆], Fernando Caro[1], and José Luis Montaña[2,⋆⋆]

[1] Centro de Inteligencia Artificial, Universidad de Oviedo
Campus de Viesques, 33271 Gijón, Spain
calonso@aic.uniovi.es
[2] Departamento de Matemáticas, Estadística y Computación,
Universidad de Cantabria
montana@matesco.unican.es

Abstract. In this paper we present an evolutionary strategy for the multidimensional 0–1 knapsack problem. Our algorithm incorporates a flipping local search process in order to locally improve the obtained individuals and also, a heuristic operator which computes problem-specific knowledge, based on the surrogate multipliers approach introduced in [12]. Experimental results show that our evolutionary algorithm is capable of obtaining high quality solutions for large size problems and that the local search procedure significatively improves the final obtained result.

Keywords: Evolutionary computation, genetic algorithms, knapsack problem, linear 0–1 integer programming.

1 Introduction

The multidimensional 0–1 knapsack problem (MKP) is a NP-complete combinatorial optimization problem which captures the essence of linear 0–1 integer programming problems. It can be formulated as follows. We are given a set of n objects where each object yields p_j units of profit and requires a_{ij} units of resource consumption in the i-th knapsack constraint. The goal is to find a subset of the objects such that the overall profit is maximized without exceeding the resource capacities of the knapsacks.

The MKP is one of the most popular constrained integer programming problems with a large domain of applications. Many practical problems can be formulated as a MKP instance: the capital budgeting problem in economy or the allocation of databases and processors in a distributed computer system ([7]) are some examples. Most of the research on knapsack problems deals with the simpler one-dimensional case ($m = 1$). For this single constraint case, the problem is

⋆ Partially supported by the Spanish MCyT and FEDER grant TIC2003-04153.
⋆⋆ Partially supported by the Spanish MCyT, under project MTM2004-01167 and Programa de Movilidad PR2005-0422.

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 21–30, 2006.

not strongly NP-hard and some exact algorithms and also very efficient approximation algorithms have been developed for obtaining near-optimal solutions.

In the multidimensional case several exact algorithms that compute different upper bounds for the optimal solutions are known. For example that in [8]. But this method, based on the computation of optimal surrogate multipliers, becomes no applicable for large values of m and n and other strategies must be introduced. In this context heuristic approaches for the MKP have appeared during the last decades following different ideas: greedy-like assignment ([6], [12]); LP-based search ([2]); surrogate duality information ([12]) are some examples. Also an important number of papers using genetic algorithms and other evolutionary strategies have emerged. The genetic approach has shown to be well suited for solving large MKP instances. In [11] a genetic algorithm is presented where infeasible individuals are allowed to participate in the search and a simple fitness function with a penalty term is used. Thiel and Voss (see [15]) presented a hybrid genetic algorithm with a tabu search heuristic. Chu and Beasley (see [5]) have developed a genetic algorithm that searches only into the feasible search space. They use a repair operator based on the surrogate multipliers of some suitable surrogate problem. Surrogate multipliers are calculated using linear programming. Also in [1] an evolutionary algorithm based on the surrogate multipliers is presented, but in this case a good set of surrogate multipliers is computed using a genetic algorithm.

In the present paper we propose an evolutionary strategy for MKP which takes as starting point the surrogate multipliers approach appeared first in [12] and later in [5] and also in [1]. Our algorithm introduces a flipping search strategy that includes random walking in hypercubes of the kind $\{0,1\}^n$. Our experimental results show that the local search procedure increases the quality of the solutions with respect to those obtained by other evolutionary algorithms based on the surrogate problem. The rest of the paper is organized as follows. Section 2 deals with the surrogate problem associated to an instance of the MKP. In section 3 we present our evolutionary algorithm for solving the MKP that incorporates our local search procedure. In section 4 experimental results and comparisons over problems taken from the OR-library(see [4][1]) and a set of large instances (proposed by Glover and Kochenberger[2])are included. Finally, section 5 contains some conclusive remarks.

2 The MKP and the Surrogate Problem

This section resumes the mathematical background and the formulation of a genetic heuristic for computing surrogate multipliers.

Definition 1. *An instance of the MKP is a 5-tuple:*

$$K = (n, m, \overline{p}, A, \overline{b}) \tag{1}$$

[1] Public available on line at http://people.brunel.ac.uk/∼mastjjb/jeb/info.html
[2] Public available at http://hces.bus.olemiss.edu/tools.html

where $n, m \in \mathbb{N}$ are both natural numbers representing (respectively) the number of objects and the number of constraints; $\overline{p} \in (\mathbb{R}^+)^n$ is a vector of positive real numbers representing the profits; $A \in M_{m\times n}(\mathbb{R}^+ \cup \{0\})$ is $m \times n$-matrix of non-negative real numbers corresponding to the resource consumptions and $\overline{b} \in (\mathbb{R}^+)^m$ is a vector of positive real numbers representing the knapsack capacities.

Remark 2. Given an instance of the MKP $K = (n, m, \overline{p}, A, \overline{b})$ the objective is

$$\text{maximize } f(\overline{x}) = \overline{p} \cdot \overline{x}' \tag{2}$$

$$\text{subject to } A \cdot \overline{x}' \leq \overline{b}' \tag{3}$$

where $\overline{x} = (x_1, \ldots, x_n)$ is a vector of variables that takes values in $\{0, 1\}^n$. Notation $\overline{v}'$ stands for the transposition of the vector $\overline{v}$.

Definition 3. *Let $K = (n, m, \overline{p}, A, \overline{b})$ be an instance of the MKP. A bit-vector $\alpha \in \{0, 1\}^n$ is a feasible solution of instance K if it verifies the constraint given by equation 3. A bit-vector $\alpha_K^{opt} \in \{0, 1\}^n$ is a solution of instance K if it is a feasible solution and for all feasible solution $\alpha \in \{0, 1\}^n$ of instance K it holds*

$$f(\alpha) \leq f(\alpha_K^{opt}) = \sum_{j=1}^{n} p_j \alpha_K^{opt}[j] \tag{4}$$

Here $\alpha[j]$ stands for the value of variable x_j.

Along this section we shall deal with a relaxed version of the MKP where the vector of variables $\overline{x}$ can take values in the whole interval $[0, 1]^n$. We will refer to this case as the LP-relaxed MKP. An instance of the LP-relaxed MKP will be denoted by K^{LP}. A solution of some instance K^{LP} of the LP-relaxed MKP will be denoted by $\alpha_{K^{LP}}^{opt}$.

Definition 4. *Let $K = (n, m, \overline{p}, A, \overline{b})$ be an instance of MKP and let $\overline{\omega} \in (\mathbb{R}^+)^m$ be a vector of positive real numbers. The surrogate constraint for K associated to $\overline{\omega}$, denoted by $Sc(K, \overline{\omega})$, is defined as follows.*

$$\sum_{j=1}^{n} (\sum_{i=1}^{m} \omega_i a_{ij}) x_j \leq \sum_{i=1}^{m} \omega_i b_i \tag{5}$$

The vector $\overline{\omega} = (\omega_1, \ldots, \omega_m)$ is called the vector of surrogate multipliers.

The surrogate instance for K, denoted by $SR(K, \overline{\omega})$, is defined as the 0–1 one-dimensional knapsack problem given by:

$$SR(K, \overline{\omega}) = (n, 1, \overline{p}, \overline{\omega} \cdot A, \overline{\omega} \cdot \overline{b}') \tag{6}$$

We shall denote by $\alpha_{SR(K,\overline{\omega})}^{opt}$ the solution of the surrogate instance $SR(K, \overline{\omega})$.

As an easy consequence of the definition of the surrogate problem we can state that:

$$f(\alpha_{SR(K,\overline{\omega})}^{opt}) \geq f(\alpha_K^{opt}) \tag{7}$$

Remark 5. The best possible upper bound for $f(\alpha_K^{opt})$ using the inequality 7 is computed by finding the minimum value $min\{f(\alpha_{SR(K,\overline{\omega})}^{opt}) : \overline{\omega} \in (\mathbb{R}^+)^m\}$. Next proposition motivates the genetic strategy to approximate the optimal values of the surrogate multipliers $\overline{\omega}$. The interested reader can find the detailed proof in [1].

Proposition 6. *Let $K = (n, m, \overline{p}, A, \overline{b})$ be any instance of MKP. Then the following inequalities are satisfied.*

$$min\{f(\alpha_{SR(K,\overline{\omega})^{LP}}^{opt}) : \overline{\omega} \in (0,1]^m\} = min\{f(\alpha_{SR(K,\overline{\omega})^{LP}}^{opt}) : \overline{\omega} \in (\mathbb{R}^+)^m\} \quad (8)$$

$$min\{f(\alpha_{SR(K,\overline{\omega})^{LP}}^{opt}) : \overline{\omega} \in (0,1]^m\} \geq f(\alpha_K^{opt}), \quad (9)$$

Here $\alpha_{SR(K,\overline{\omega})^{LP}}^{opt}$ denotes the solution of the relaxed surrogate problem $SR(K,\overline{\omega})^{LP}$.

2.1 A Genetic Algorithm for Computing the Surrogate Multipliers

The following is a simple genetic algorithm (GA) to obtain approximate values for $\overline{\omega}$ (see [1]). In this GA the individuals will be binary 0–1 strings, representing $\overline{\omega} = (\omega_1, \ldots, \omega_m)$. Each ω_i will be represented as a q-bit binary substring, where q determines the desired precision of $\omega_i \in (0, 1]$.

Definition 7. *Given an MKP instance $K = (n, m, \overline{p}, A, \overline{b})$, $q \in \mathbb{N}$, and a representation $\gamma \in \{0,1\}^{qm}$ of a candidate vector of surrogate multipliers $\overline{\omega} = (\omega_1, \ldots, \omega_m)$ the fitness value of γ is defined as follows:*

$$fitness(\gamma) = f(\alpha_{SR(K,\overline{\omega})^{LP}}^{opt}) = \sum_{j=1}^{n} p_j \alpha_{SR(K,\overline{\omega})^{LP}}^{opt}[j] \quad (10)$$

Remark 8. The solution $\alpha_{SR(K,\overline{\omega})^{LP}}^{opt}$ can be obtained by means of the well known greedy algorithm for one-dimensional instances of the LP–relaxed knapsack problem.

Note that our objective is to minimize the $fitness$ function defined by equation 10. The remainder operators of the genetic algorithm have been chosen as follows: the roulette wheel rule as selection procedure, uniform crossover as recombination operator and bitwise mutation according to a given probability p_m (see [10] for a detailed description of these operators).

3 The Evolutionary Algorithm for the MKP

Given an instance of MKP, $K = (n, m, \overline{p}, A, \overline{b})$, and a set of surrogate multipliers, $\overline{\omega}$, computed by the genetic algorithm described in the previous section, we will run an steady state evolutionary algorithm for solving K that searches only into the feasible search space. So, it includes a repair operator for the infeasible individuals, that uses as heuristic operator the set of surrogate multipliers, and also an improvement technique. As main contribution, we incorporate to this algorithm a local search procedure in order to locally improve the individuals of the population.

3.1 Individual Representation and Fitness Function

We choose the standard 0–1 binary representation since it represents the underlying 0–1 integer values. A chromosome α representing a candidate solution for $K = (n, m, \overline{p}, A, \overline{b})$ is an n-bit binary string. A value $\alpha[j] = 0$ or 1 in the j-bit means that variable $x_j = 0$ or 1 in the represented solution. The fitness of chromosome $\alpha \in \{0,1\}^n$ is defined by

$$f(\alpha) = \sum_{j=1}^{n} p_j \alpha[i] \tag{11}$$

3.2 Repair Operator and Improve Procedure

The repair operator lies on the notion of utility ratios. Let $K = (n, m, \overline{p}, A, \overline{b})$ be an instance of MKP and let $\overline{\omega} \in (0,1]^m$ be a vector of surrogate multipliers. The utility ratio for variable x_j is defined by $u_j = \frac{p_j}{\sum_{i=1}^{m} \omega_i a_{ij}}$. Given an infeasible individual $\alpha \in \{0,1\}^n$ we apply the following repairing procedure.

Procedure DROP ([5], [1])
input: $K = (n, m, \overline{p}, A, \overline{b})$; $\overline{\omega} \in (0,1]^m$ and a chromosome $\alpha \in \{0,1\}^n$

```
begin
    for j=1 to n compute u_j
    P:=permutation of (1,...,n) with u_P[j] <= u_P[j+1]
    for j=1 to n do
        if (alpha[P[j]]=1 and infeasible(alpha)) then
            alpha[P[j]]:=0
end
```

Once we have transformed α into a feasible individual α', a second phase is applied in order to improve α'. This second phase is called the ADD phase.

Procedure ADD ([5], [1])
input: $K = (n, m, \overline{p}, A, \overline{b})$; $\overline{\omega} \in (0,1]^m$ and a chromosome $\alpha \in \{0,1\}^n$

```
begin
    P:=permutation of (1,...,n) with u_P[j] >= u_P[j+1]
    for j=1 to n do
        if alpha[P[j]]=0 then alpha[P[j]]:=1
        if infeasible(alpha) then alpha[P[j]]:=0
end
```

3.3 Genetic Operators

We use the roulette wheel rule as selection procedure, the uniform crossover as replacement operator and bitwise mutation with a given probability p_m. So when mutation must be applied to an individual α, a bit j from α is randomly selected and flipped from its value $\alpha[j] \in \{0,1\}$ to $1 - \alpha[j]$.

3.4 Local Search Procedure

Let $K = (n, m, \overline{p}, A, \overline{b})$ be an instance of MKP, $\overline{\omega} \in (0, 1]^m$ a vector of surrogate multipliers and $\alpha \in \{0, 1\}^n$ a chromosome representing a feasible solution of K. We describe below a local search procedure that can be applied to α. During the execution of this procedure, chromosomes representing infeasible solutions of K could be generated. So, the repair and improve operators described above should be applied. We will refer to $DROP_j$ as the operator that behaves as DROP but with the property that it does not modify the gen $\alpha[j]$ in chromosome α. We will refer to ADD_j in an analogous way.

Assuming that the iteration begins with a chromosome α, at each iteration step, the local search performs a random walk inside the hypercube $\{0, 1\}^n$ generating a permutation P of length n. Then, for each $j \in \{1, \ldots, n\}$ makes a flip in $\alpha[P[j]]$ if and only if there is a gain in the fitness. Note that after that the flip was made, the generated chromosome could represent an infeasible solution. If it is the case we apply first $DROP_{P[j]}$, and then the procedure ADD. In other case –when the flip does not turns the chromosome infeasible– we just apply the procedure $ADD_{P[j]}$. The described process iterates until there is no gain in the fitness.

Procedure LocalSearch
input: $K = (n, m, \overline{p}, A, \overline{b})$; $\overline{\omega} \in (0, 1]^m$ and a chromosome $\alpha \in \{0, 1\}^n$

```
begin
    repeat
        beta:=alpha
        P:=permutation of (1,...,n)
        for j=1 to n do
            alpha_aux:=alpha
            alpha:=flip(P[j],alpha)
            if f(alpha) <= f(alpha_aux) then alpha:=alpha_aux
    until alpha=beta
end
```

Where the pseudo–code of the procedure $flip\,(i, \alpha)$ is as follows:

```
begin
    if alpha[i]=1 then
        alpha[i]:=0
        alpha:=ADD_i(K,omega,alpha)
    else
        alpha[i]=1
        alpha:=DROP_i(K,omega,alpha)
        alpha:=ADD(K,omega,alpha)
end
```

Our evolutionary algorithm follows the performance of the steady state GA described in [1] but introducing the above local search procedure. In this sense we do not apply the local search procedure to all generated individuals but only

periodically, each time that a number t of generations were produced. In this case we apply local search to all individuals in the population. The period t is given by the user and also the local search procedure is applied to the initial and final populations. To maintain diversity of individuals during the execution, the strategy used while applying the local search procedure to a population $P(kt)$, $k \in \mathbb{N}$, is the following:

```
begin
    P(kt+1):=empty_set
    for each alpha in P(kt) do
        beta:=local_search(alpha)
        if (beta belongs to (P(kt) or P(kt+1))) then
            insert alpha in P(kt+1)
        else
            insert beta in P(kt+1)
end
```

Remark 9. Individuals of the initial population are constructed generating random permutations from $(1, \ldots, n)$, applying the ADD procedure to the chromosome $\alpha = (0, \ldots, 0)$ following the permutation order.

4 Experimental Results

We have executed our double genetic algorithm with local search (DGALS) on two sets of MKP instances. The first set of instances is included in the OR-Library proposed in [4][3]. They are randomly generated instances of MKP with number of constraints $m \in \{5, 10, 30\}$, number of variables $n \in \{100, 250, 500\}$ and tightness ratios $r \in \{0.25, 0.5, 0.75\}$. The tightness ratio r fixes the capacity of i-th knapsack to $r \sum_{j=1}^{n} a_{ij}$, $1 \leq i \leq m$. There are 10 instances for each combination of m, n and r giving a total of 270 test problems. The second set is a set of 11 instances proposed by Glover and Kochenberger[4]. These are large instances, up to $n = 2500$ and $m = 100$. After a previous experimentation we have set the parameters to the following values. The GA computing the surrogate multipliers uses population size 75, precision $q = 10$, probability of mutation 0.1 and 15000 generations to finish. The steady state GA solving the MKP instances uses population size 100, probability of mutation equal to 0.1 and the algorithm finishes when 1500000 evaluations have been performed. The period t to apply local search is set to 10^4 generations as a good trade-off between quality of the final solutions and computational effort.

For the first set of instances, since the optimal solutions values are unknown, the quality of a solution α is measured by the percentage gap of its fitness value with respect to the fitness values of the optimal solution of the LP-relaxed problem: $\%gap = 100\frac{f(\alpha_{K^{LP}}^{opt}) - f(\alpha)}{f(\alpha_{K^{LP}}^{opt})}$. We will compare our algorithm with that

[3] Public available on line at http://people.brunel.ac.uk/∼mastjjb/jeb/info.html
[4] Public available at http://hces.bus.olemiss.edu/tools.html

presented in [5] (CHUGA), and the percentage gap is also the quality measurement used in that work. For the second set of 11 instances, we compare the best known solutions with that obtained by our evolutionary algorithm and also we compute the %*gap* with respect to the fitness values of that best known solutions.

We measure the time complexity by means of the number of evaluations required to find the best obtained solution (E.B.S.), as the individual representation and fitness function are the same in CHUGA, DGA and DGALS. We have executed our evolutionary algorithm on a Pentium IV; 3GHz. Over this platform the execution time for a single run ranges from 5 minutes, for the simplest problems, to 10 hours, for the most complex ones.

The results of our experiments are displayed in Tables 1 and 2 based on 10 independent executions for each instance. In table 1, our algorithm (DGALS), executed over the set of 270 instances is compared with that of Chu et. al (CHUGA) ([5]) and also with our first version without local search (DGA) ([1]). The first two columns identify the 30 instances that corresponds to each combination of m, n. In the remainder columns we show for the compared algorithms the average %gap and the average number of evaluations required until the best individual was encountered (A.E.B.S). We have taken the values corresponding to CHUGA from [5] and [13]. As the authors have pointed out in their work these results are based on only one run for each problem instance whereas in the case of our genetic algorithms 10 runs were executed. In our experiments the variance for the %gap is zero until the third decimal digit for all set of instances corresponding to each combination of m, n. In table 2 we display the results concerning to the set of 11 large instances. In this case we present only results for DGALS compared with the best known solutions obtained by Glover and Kochenberger (GLOVERA) ([9]). We also display the average percentage gap of the fitness of our solutions with respect to the fitness of the best known solutions.

From table 1 we conclude that our DGALS performs better than CHUGA and DGA in terms of the average %gaps and it seems that this better performance

Table 1. Computational results for CHUGA, DGA and DGALS. Values for DGA and DGALS are based on 10 runs for each instance.

Problem		GHUGA		DGA		DGALS	
m	n	A. %gap	A.E.B.S	A. %gap	A.E.B.S	A. %gap	A.E.B.S
5	100	0.59	24136	0.58	58045	0.58	99549
5	250	0.16	218304	0.15	140902	0.14	381840
5	500	0.05	491573	0.06	185606	0.05	675086
10	100	0.94	318764	0.98	70765	0.94	175630
10	250	0.35	475643	0.32	153475	0.29	495312
10	500	0.14	645250	0.15	179047	0.13	705148
30	100	1.74	197855	1.71	106542	1.70	217794
30	250	0.73	369894	0.71	184446	0.67	532870
30	500	0.40	587472	0.44	233452	0.36	716827

Table 2. Computational results for the 11 large instances. We compare DGALS with GLOVERA. Values A.E.B.S. and A. %gap of DGALS are based on 10 runs for each instance.

Problem		GLOVERA	DGALS		
m	n	Best Sol.	Best Sol.	A.E.B.S.	A. %gap
15	100	3766	3766	98857	0
25	100	3958	3958	196649	0
25	150	5650	5656	502719	0.01
25	200	7557	7557	234675	0.01
25	500	19215	19211	875213	0.02
25	1500	58085	58078	1128350	0.01
50	150	5764	5764	273140	0
50	200	7672	7672	360145	0.03
50	500	18801	18796	950548	0.03
50	1500	57292	57295	1459603	0.05
100	2500	95231	95378	1487329	-0.14

increases with the instance size. We can see also that the use of our local search procedure obviously suppose an increase in the amount of computational effort in order to reach the best solution. Nevertheless this is a reasonable increment and, as we have mentioned, the found solutions are significatively better. Although it is not showed in a table, we have done some executions only with local search applied to random generated individuals, obtaining poor results. This also agrees with the fact that using local search in all generations does not guarantees better individuals after 1500000 evaluations.

The results presented in table 2 agree with our conjecture that DGALS performs better with large instances. We think that the reason that in some of these problems we have not reached the best known solution is because we would need more than 1500000 evaluations. Specially surprising are the results for the largest instances, where we have obtained considerably better solutions in all executions of DGALS.

5 Conclusive Remarks

In this paper we have presented an evolutionary algorithm for solving multidimensional knapsack problems based on genetic computation of surrogate multipliers incorporating a flipping local search procedure. On a large set of problems, we have shown that our evolutionary algorithm is capable of obtaining high-quality solutions for large problems of various characteristics. Clearly, the use of a local search procedure improves the quality of the individuals and directs the GA to the best solutions. Periodical application of the local search procedure reduces the amount of evaluations per generation and maintains diversity in the population. Both aspects seem to be crucial for the performance of our evolutionary algorithm.

References

1. Alonso, C. L., Caro, F., Montaña, J. L.: An Evolutionary Strategy for the Multidimensional 0–1 Knapsack Problem based on Genetic Computation of Surrogate Multipliers. Artificial Intelligence and Knowledge Engineering Applications: A Bioinspired Approach; LNCS 3562 (2005) 63–73.
2. Balas, E., Martin, C. H.: Pivot and Complement–A Heuristic for 0–1 Programming. Management Science 26 (1) (1980) 86–96
3. Balas, E., Zemel, E.: An Algorithm for Large zero–one Knapsack Problems. Operations Research 28 (1980) 1130–1145
4. Beasley, J. E.: Obtaining Test Problems via Internet. Journal of Global Optimization 8 (1996) 429–433
5. Chu, P. C., Beasley, J. E.: A Genetic Algorithm for the Multidimensional Knapsack Problem. Journal of Heuristics 4 (1998) 63–86
6. Freville, A., Plateau, G.: Heuristics and Reduction Methods for Multiple Constraints 0–1 Linear Programming Problems. Europena Journal of Operationa Research 24 (1986) 206–215
7. Gavish, B., Pirkul, H.: Allocation of Databases and Processors in a Distributed Computing System. J. Akoka ed. Management od Distributed Data Processing, North-Holland (1982) 215–231
8. Gavish, B., Pirkul, H.: Efficient Algorithms for Solving Multiconstraint Zero–One Knapsack Problems to Optimality. Mathematical Programming 31 (1985) 78–105
9. Glover, F., Kochenberger, G. A.: Critical event tabu search for multidimensional knapsack problems. Metaheuristics: The Theory and Applications. Kluwer Academic Publishers (1996) 407–427
10. Goldberg, D. E.: Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley (1989)
11. Khuri, S., Bäck, T., Heitkötter, J.: The Zero/One Multiple Knapsack Problem and Genetic Algorithms. Proceedings of the 1994 ACM Symposium on Applied Computing (SAC'94), ACM Press (1994) 188–193
12. Pirkul, H.: A Heuristic Solution Procedure for the Multiconstraint Zero–One Knapsack Problem. Naval Research Logistics 34 (1987) 161–172
13. Raidl, G. R.: An Improved Genetic Algorithm for the Multiconstraint Knapsack Problem. Proceedings of the 5th IEEE International Conference on Evolutionary Computation (1998) 207–211
14. Rinnooy Kan, A. H. G., Stougie, L., Vercellis, C.: A Class of Generalized Greedy Algorithms for the Multi-knapsack Problem. Discrete Applied Mathematics 42 (1993) 279–290
15. Thiel, J., Voss, S.: Some Experiences on Solving Multiconstraint Zero–One Knapsack Problems with Genetic Algorithms. INFOR 32 (1994) 226–242

A Hierarchical Pattern Matching Procedure for Signal Abstraction

A. Otero[1,⋆], P. Félix[1], S. Fraga[1], S. Barro[1], and F. Palacios[2]

[1] Dpto. de Electrnica e Computacin, Universidade de Santiago de Compostela. Santiago de Compostela. 15782, Spain
abraham@dec.usc.es
[2] Hospital Universitario de Getafe, Madrid. Spain

Abstract. The Multivariable Fuzzy Temporal Profile model enables human experts to project their knowledge of a signal pattern, described over a set of parameters, into a computable description. Here it is endowed with a hierarchical capability through the definition of algorithms that recognize a set of findings over a signal, and aggregate them into a set of abstraction levels. We also present a heuristics that takes advantage of the continuity properties of real signals to allow the matching process to meet real time requirements, even over high frequency signals.

1 Introduction

This work has its origin in the thesis that processes of abstraction over perception are organized into a hierarchical structure. Thus, the recognition and interpretation of observable events is carried out by successively aggregating pieces of information -findings- that are extracted from the data acquired. This operation results in a decrease in the volume of data handled, and an increase in their semantics. With this idea in mind, we have developed a hierarchical matching procedure for the Multivariable Fuzzy Temporal Profile (MFTP) model.

This model, presented in [5], makes use of the fuzzy set theory [6] to represent and handle the imprecision and uncertainty that are characteristic of human knowledge, and to allow the direct acquisition of knowledge on patterns from human experts. Constraint Satisfaction Problems (CSP) [1] are used to supply a structural description of signal patterns that enables the hierarchy of abstraction that is present in pattern recognition to be captured, making it possible to give detailed explanations of the matching results to the expert, as well as to describe sets of patterns that have common findings.

This paper is organized as follows: section 2 introduce certain fuzzy concepts on which the MFTP model is based, in order to then present in Section 3 a real example that we use to motivate and formalize the model, in Section 4. Section 5 explains how the hierarchical matching is carried out, and a heuristics

⋆ The authors wish to acknowledge the support from the Spanish CICyT and the Xunta de Galicia under the projects TIC2003-09400-C04-03 and PGIDIT04SIN206003PR, respectively.

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 31–41, 2006.

that exploits the hierarchical nature of the model to improve matching efficiency is presented in Section 6. Finally, we present the conclusions of this work.

2 Fuzzy Concepts

We will consider time as being projected onto a one-dimensional discrete axis $\tau = \{t_0, t_1, ..., t_i, ...\}$, where t_i represents a *precise* instant and t_0 represents the temporal origin. We consider that for every $i \in \mathbb{N}$, $t_{i+1} - t_i = \Delta t$, where the constant Δt is the minimum step of the temporal axis.

Given as discourse universe the set of real numbers $\mathbb{R}$, a **fuzzy number** A is a normal ($\exists\, v \in \mathbb{R}$, $\mu^A(v) = 1$) and convex ($\forall v, v', v'' \in \mathbb{R}$, $v' \in [v, v'']$, $\mu^A(v') \geq \min\{\mu^A(v), \mu^A(v'')\}$) fuzzy subset of $\mathbb{R}$. We obtain a fuzzy number A from a flexible constraint given by a possibility distribution π^A, which defines a mapping from $\mathbb{R}$ to the real interval $[0, 1]$. Given a precise number $v \in \mathbb{R}$, $\pi^A(v) \in [0, 1]$ represents the possibility of A being precisely v. By means of π^A we define a fuzzy subset A of $\mathbb{R}$, which contains the possible values of A.

We introduce the concept of **fuzzy increment** with the aim of representing quantities, such as the difference between two numbers, which may be fuzzy or not. Following Zadeh's extension principle [6], the fuzzy increment between a pair of fuzzy numbers A and B is given by D such $\pi^D(i) = \max_{i=t-s} \min\{\pi^A(t), \pi^B(s)\}$.

3 Clinical Example

We use an example from the clinical domain to introduce the MFTP model: the recognition of the sinusal P wave in a multichannel electrocardiogram (ECG) recording. The P wave is studied as a sign of auricular electrical activity, and is of great use in the diagnosis of arrhythmias and auriculo-ventricular conduction abnormalities. Nevertheless, its detection is still problematic in automatic ECG monitoring and currently there is no commercial bedside monitor that provides this information. The difficulty in their detection lies in the low relative voltage of these waves, the frequency with which small artifacts simulate them, as well as in the morphological variations produced even by breathing movements.

The events of the auricle-ventricular complex that enable or help in the diagnosis of arrhythmias and conduction abnormalities are the onset of the P wave and the onset of the QRS [3]. Among the multiple ECG leads of the monitoring, II and aV_R are chosen for their suitability (see Fig. 1). Over Lead II the pattern is defined as a gradual increase from a basal ECG value, corresponding with the onset of the P wave, and a very sharp decrease, corresponding with the onset of the QRS. Both events must be separated by approximately 120-200 ms. Over aV_R the P wave must be negative, thus there must be a gentle decrease in aV_R, practically simultaneously with the gentle increase over II. Non-compliance with these criteria precludes a sinusal origin for the auricular activity.

4 The MFTP Model

The MFTP model enables the identification of a pattern $\mathcal{M}$ of special significance over the temporal evolution of a set of parameters $\mathcal{P} = \{P^1, ..., P^s\}$, where each parameter P^j is obtained by means of an acquisition and sampling process: $P^j = \{(v^j_{[1]}, t^j_{[1]}), ..., (v^j_{[k]}, t^j_{[k]}), ...\}$. This pattern is described by a human expert and consists of a set of findings and relations between them.

The MFTP model is an extension of the Fuzzy Temporal Profile model [2], which allows a finding to be represented as a morphology described over a single physical parameter. The fact of being able to relate the occurrence of different findings among parameters is of great importance, as often the appearance of a finding over a single parameter, which on its own may not be a major determinant, may well be of interest if it appears to be related with other findings on other parameters which also do not seem to be definitive when considered in isolation. In our clinical example, the appearance of the pattern described over Lead II does not imply a sinusal beat if in aV_R P wave is positive.

Furthermore, the MFTP model is able to explicitly represent the hierarchy of abstraction levels that the expert employs to reason about the system. This enables detailed explanations of the reasons behind the occurrence or non-occurrence of a pattern to be given. Going back to the example, the independent representation of the onset of the P wave and of the QRS complex makes it possible to reason over whether the non-occurrence of the pattern was caused by an abnormal origin of the beat (P wave negative in II), or whether there has been a first-degree blockage (excessive distance between the P wave and the QRS complex) or a second-degree one (the P wave is not followed by a QRS complex).

The MFTP model is based on the CSP formalism and on the fuzzy set theory. An MFTP allows a pattern to be represented by means of a network of fuzzy constraints between a set of significant points.

Definition 1. *We define* ***significant point*** *on a physical parameter P^j, X^j_i, as the pair formed by a variable from the domain V^j_i and a temporal variable T^j_i. A significant point $X^j_i = < V^j_i, T^j_i >$ represents an unknown value V^j_i for P^j at an unknown temporal instant T^j_i. In the absence of constraints, V^j_i and T^j_i may take any precise value $v^j_{[k]}$ and $t^j_{[k]}$, respectively, where $(v^j_{[k]}, t^j_{[k]}) \in \mathcal{P}^j$.*

By $\mathcal{A}^j_i$ we denote the assignment of precise values from the evolution P^j to the variables of X^j_i; i.e., $\mathcal{A}^j_i = (v^j_{[k]}, t^j_{[k]})$. We define a general fuzzy constraint between a set of significant points, providing a computable support for soft descriptions of the form of a signal.

Definition 2. *A* ***fuzzy constraint*** *R between a set of significant points $\{X^{j_1}_{i_1}, X^{j_2}_{i_2}, ..., X^{j_g}_{i_g}\}$ is defined by means of a fuzzy relation $C = C(X^{j_1}_{i_1}, X^{j_2}_{i_2}, ..., X^{j_g}_{i_g})$.*

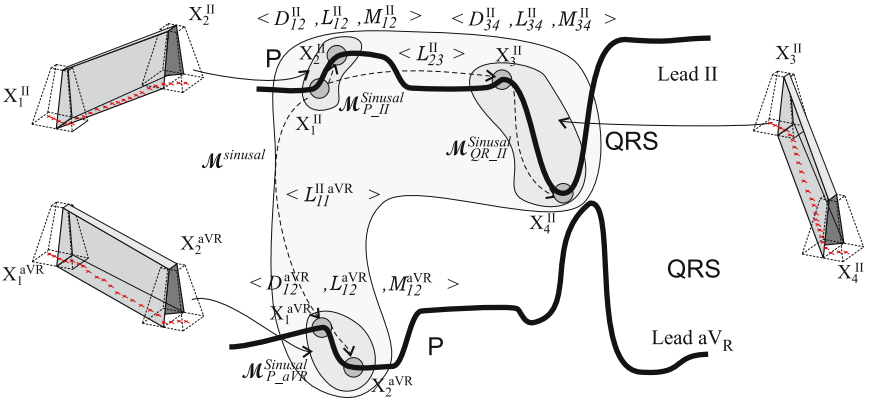


Fig. 1. Graph of the sinusal rhythm MFTP along with the graphical representation of the fuzzy trajectories that the MFTP defines for each finding

C is represented by means of a membership function μ^C*, which associates a degree of satisfaction of R to each assignment of precise values to* $X_{i_1}^{j_1}, X_{i_2}^{j_2}, ..., X_{i_g}^{j_g}$.

In principle, nothing restricts the form of the constraints that make up a MFTP. However, experience has shown that a set of constraints limiting the fuzzy increment, fuzzy temporal extension and fuzzy slope between a pair of significant points is able to capture a good number of nuances. Thus we define a constraint $L_{i_1 i_2}^{j_1 j_2}$ between two significant points $X_{i_1}^{j_1}$ and $X_{i_2}^{j_2}$ by means of a normal, convex possibility distribution $\mu^{L_{i_1 i_2}^{j_1 j_2}}(X_{i_1}^{j_1}, X_{i_2}^{j_2}) = \pi^{L_{i_1 i_2}^{j_1 j_2}}(h)$, $h \in \tau$, which represents the possibility of the fuzzy temporal extension between $X_{i_1}^{j_1}$ and $X_{i_2}^{j_2}$ being h. The assignations $T_{i_1}^{j_1} = t_{i_1}^{j_1}$ and $T_{i_2}^{j_2} = t_{i_2}^{j_2}$ are possible if $\pi^{L_{i_1 i_2}^{j_1 j_2}}(t_{i_2}^{j_2} - t_{i_1}^{j_1}) > 0$. In Fig. 1 L_{23}^{II} models the linguistic description "*approximately of 120-200 ms*".

A constraint $D_{i_1 i_2}^{j_1 j_2}$ between a pair of significant points $X_{i_1}^{j_1}$ and $X_{i_2}^{j_2}$ is defined by means of a normal and convex possibility distribution $\mu^{D_{i_1 i_2}^{j_1 j_2}}(X_{i_1}^{j_1}, X_{i_2}^{j_2}) = \pi^{D_{i_1 i_2}^{j_1 j_2}}(d)$, $d \in \mathbb{R}$, which represents the possibility of the fuzzy increase between $X_{i_1}^{j_1}$ and $X_{i_2}^{j_2}$ being d. The assignations $V_{i_1}^{j_1} = v_{i_1}^{j_1}$ and $V_{i_2}^{j_2} = v_{i_2}^{j_2}$ are possible if $\pi^{D_{i_1 i_2}^{j_1 j_2}}(v_{i_2}^{j_2} - v_{i_1}^{j_1}) > 0$. In Fig. 1 D_{12}^{II} models the description "*gentle increase*".

A constraint $M_{i_1 i_2}^{j}$ between a pair of significant points $X_{i_1}^{j}$ and $X_{i_2}^{j}$, defined over the same parameter P^j, is defined by means of a normal and convex possibility distribution $\mu^{M_{i_1 i_2}^{j}}(X_{i_1}^{j}, X_{i_2}^{j}) = \pi^{M_{i_1 i_2}^{j}}(m)$, $m \in \mathbb{R}$, which represents the possibility of the fuzzy slope between $X_{i_1}^{j}$ and $X_{i_2}^{j}$ being m. The assignations $V_{i_1}^{j} = v_{i_1}^{j}$, $V_{i_2}^{j} = v_{i_2}^{j}$, $T_{i_1}^{j} = t_{i_1}^{j}$ and $T_{i_2}^{j} = t_{i_2}^{j}$ are possible if $\pi^{M_{i_1 i_2}^{j}}((v_{i_2}^{j} - v_{i_1}^{j})/(t_{i_2}^{j} - t_{i_1}^{j})) > 0$. In Fig. 1 M_{34}^{II} models the description "*...sharp* decrease", where "*sharp*" is modelled by means of a high slope value.

Definition 3. *We define a* ***Multivariable Fuzzy Temporal Profile*** *(MFTP)* $\mathcal{M} =< \mathcal{W}^{\mathcal{M}}, \mathcal{X}^{\mathcal{M}}, \mathcal{R}^{\mathcal{M}} >$ *as a finite set of MFTPs* $\mathcal{W}^{\mathcal{M}} = \{\mathcal{M}_1^{\mathcal{M}}, ..., \mathcal{M}_s^{\mathcal{M}}\}$,

a finite set of significant points $\mathcal{X}^{\mathcal{M}} = \{X_{i_1}^{j_1}, X_{i_2}^{j_2}, ..., X_{i_g}^{j_g}\}$ *and a finite set of constraints* $\mathcal{R}^{\mathcal{M}} = \{R_1, ..., R_f\}$ *amongst the points of* $\mathcal{W}^{\mathcal{M}}$ *and* $\mathcal{X}^{\mathcal{M}}$.

The constraints $R_i \in \mathcal{R}^{\mathcal{M}}$ can be defined between significant points belonging to $\mathcal{X}^{\mathcal{M}}$, between significant points belonging to the set of subMFTPs $\mathcal{W}^{\mathcal{M}}$ or between both types of significant points. The recursive structure of the MFTP model is based in the way that humans define patterns; a complex pattern is often made up of a set of findings and a set of relations between them. Each of the findings of the pattern may also be a pattern, and may comprise a set of findings and relations between them, and so on, successively.

The pattern of the sinusal P wave is made up of three findings for which certain temporal relations between them must be satisfied: $\mathcal{M}^{Sinusal} =< \{\mathcal{M}_{P\_II}^{Sinusal}, \mathcal{M}_{QR\_II}^{Sinusal}, \mathcal{M}_{P\_aV_R}^{Sinusal}\}, \varnothing, \{L_{23}^{II}, L_{11}^{II\ aV_R}\} >$. Each of these findings is in turn an MFTP that is defined over its corresponding lead; thus $\mathcal{M}_{P\_II}^{Sinusal} =< \varnothing, \{X_1^{II}, X_2^{II}\}, \{L_{12}^{II}, D_{12}^{II}, M_{12}^{II}\} >$, $\mathcal{M}_{QR\_II}^{Sinusal} =< \varnothing, \{X_3^{II}, X_4^{II}\}, \{L_{34}^{II}, D_{34}^{II}, M_{34}^{II}\} >$ and $\mathcal{M}_{P\_aV_R}^{Sinusal} =< \varnothing, \{X_1^{aV_R}, X_2^{aV_R}\}, \{L_{12}^{aV_R}, D_{12}^{aV_R}, M_{12}^{aV_R}\} >$.

An MFTP can be represented by a graph in which nodes correspond to significant points, and arcs correspond to constraints (see Fig. 1). The MFTP model also enables us to restrict the evolution of a parameter P^j between each pair of significant points $X_{i_1}^j$ and $X_{i_2}^j$ (see Fig. 1) by means of a membership function $\mu^{\mathcal{S}_{i_1 i_2}^j}(\mathcal{A}_{i_1}^j, \mathcal{A}_{i_2}^j)$ which defines a fuzzy course within which the temporal evolution of the parameter must remain in order to satisfy the constraint [2].

5 Matching

The ultimate aim of the MFTP model is to identify a pattern $\mathcal{M}$ over a set of parameters $\mathcal{P}$, which represent the temporal evolution of a physical system S, automatically generating information organized in a hierarchy of levels of abstraction. Given that the MFTP model is based on the formalism of constraint networks, comparing an MFTP with $\mathcal{P}$ is formally equivalent to solving a CSP [1], where the domains of the variables are determined by $\mathcal{P}$.

Definition 4. *We define a* ***solution*** *of* $\mathcal{M}$ *as a set of assignations* $\mathcal{A} = \{A_1^1, ..., A_{n^1}^1, ..., A_1^s, ..., A_{n^s}^s\}$ *to all the significant points of the pattern that satisfies the set of constraints that make up* $\mathcal{M}$*, with a degree greater than zero. The degree of satisfaction of a solution* $\mathcal{A}$ *is given by:*

$$\pi^{\mathcal{M}}(\mathcal{A}) = min\{\min_{\mathcal{M}_h^{\mathcal{M}} \in \mathcal{W}^{\mathcal{M}}} \{\pi^{\mathcal{M}_h^{\mathcal{M}}}(\mathcal{A}^{\mathcal{M}_h^{\mathcal{M}}})\}, \min_{R_k \in \mathcal{R}^{\mathcal{M}}} \{\pi^{R_k}(\mathcal{A}^{R_k})\}\} \quad (1)$$

Where $\mathcal{A}^{\mathcal{M}_h^{\mathcal{M}}}$ and $\mathcal{A}^{R_k}$ are the projection of $\mathcal{A}$ over the set of significant points involved in $\mathcal{M}_h^{\mathcal{M}}$ and in R_k, respectively. π^{R_k} is the degree of satisfaction of $R_k \in \mathcal{R}^{\mathcal{M}}$ and $\pi^{\mathcal{M}_h^{\mathcal{M}}}$ is the degree of satisfaction of $\mathcal{M}_h^{\mathcal{M}} \in \mathcal{W}^{\mathcal{M}}$. $\pi^{\mathcal{M}}(\mathcal{A})$ represents the degree of similarity between a fragment of the evolution $\mathcal{P}$ with the description represented by $\mathcal{M}$.

Imitating human experts, we divide the matching into as many stages as the number of levels of abstraction given by the composition of findings. In the lowest level is the temporal evolution of the system $\mathcal{P}$, and in the highest the global pattern $\mathcal{M}$. Figure 2a shows a recursive procedure which runs through the hierarchy of the pattern until it reaches the lowest level MFTPs; it resolves them and then goes up the next level of abstraction and resolves the MFTPs there, by assembling solutions found for the MFTPs in lower levels. The procedure comes to an end when solutions for the global pattern are obtained. For example, in order to match the sinusal beat pattern, firstly we search for solutions for the three subMFTPs: $\mathcal{M}_{P\_II}^{Sinusal}, \mathcal{M}_{QR\_II}^{Sinusal}$, $\mathcal{M}_{P\_aV_R}^{Sinusal}$ and we then attempt to assemble the solutions encountered in order to obtain solutions for $\mathcal{M}^{Sinusal}$.

The solutions to the MFTPs on each level are constructed incrementally, by exploring a search tree. We suppose that the global pattern $\mathcal{M}$ has f nesting levels, where $\mathcal{M}_h^l$ is the subMFTP h in level l. The matching starts with the lowest-level MFTPs, those which are defined directly over the system's signal: $\mathcal{M}_h^o = \{\varnothing, X^{\mathcal{M}_h^o}, \mathcal{R}^{\mathcal{M}_h^o}\}$. We start from an empty tuple of assignations, and we extend it with an assignation to a new significant point, such that the degree of satisfaction of the extended tuple is greater than or equal to a limit $c_{\inf}$, below which the degree of satisfaction of a solution is considered to be unacceptable. The degree of satisfaction of the tuple $\mathcal{A}_{[r+1]}$ of assignations to $r+1$ significant points of $X^{\mathcal{M}_h^o}$, $\mathcal{A}_{[r+1]} = \{A_{i_1}^{j_1}, ..., A_{i_r}^{j_r}, A_{i_{r+1}}^{j_{r+1}}\}$, is calculated on basis of the degree of satisfaction of $\mathcal{A}_{[r]}$ by means of:

$$\pi^{\mathcal{M}_h^o}(\mathcal{A}_{[r+1]}) = min\{\pi^{\mathcal{M}_h^o}(\mathcal{A}_{[r]}), \underset{R \in \mathcal{R}^{[\mathcal{A}_{[r]}, A_{i_{r+1}}^{j_{r+1}}]}}{min}\{\pi^R(\mathcal{A}_{[r+1]})\}\} \tag{2}$$

where $\mathcal{R}^{[\mathcal{A}_{[r]}, A_{i_{r+1}}^{j_{r+1}}]}$ is the set of those constraints that are defined over points to which $\mathcal{A}_{[r+1]}$ assigns value, and which involve $X_{i_{r+1}}^{j_{r+1}}$. For example, for matching $\mathcal{M}_{P\_II}^{Sinusal}$ we start by carrying out an assignation $\mathcal{A}_1^{II} = (v_1^{II}, t_1^{II})$ to X_1^{II}; we then go on to search for an assignation $\mathcal{A}_2^{II} = (v_2^{II}, t_2^{II})$ that satisfies the set of constraints $\{L_{12}^{II}, D_{12}^{II}, M_{12}^{II}\}$. If no such assignation is found, we backtrack to the last significant point to which a value has been assigned, and we attempt to carry out another assignation. The same procedure is followed for the recognition of $\mathcal{M}_{QR\_II}^{Sinusal}$ and $\mathcal{M}_{P\_aV_R}^{Sinusal}$. We then search for solutions to those MFTPs for which all subMFTPs have been resolved; i.e. those that are in nesting level 1. For these we start from an empty tuple of assignations, and we initially extend it with solutions that have previously been found for their subMFTPs. On the basis of the degree of satisfaction of the initial tuple, $\mathcal{A}_{[r]}$, and of the solution of the lower level subMFTP that has been assembled, $\mathcal{A}^{\mathcal{M}_h^o}$, we obtain the degree of satisfaction for the extended tuple $\mathcal{A}_{[s]}$, $\mathcal{A}_{[s]} = \mathcal{A}_{[r]} \cup \mathcal{A}^{\mathcal{M}_h^o}$, by means of:

$$\pi^{\mathcal{M}_k^1}(\mathcal{A}_{[s]}) = min\{\pi^{\mathcal{M}_k^1}(\mathcal{A}_{[r]}), \pi^{\mathcal{M}_h^o}(\mathcal{A}^{\mathcal{M}_h^o}), \underset{R \in \mathcal{R}^{[\mathcal{A}_{[r]}, \mathcal{A}^{\mathcal{M}_h^o}]}}{min}\{\pi^R(\mathcal{A}_{[s]}^R)\}\} \tag{3}$$

```
procedure MATCH(M);
 if(M contains MFTPs) then
  for each (M^M ∈ M)
   MATCH(M^M);
 end if;
 BUILD_SOL(M, 0);
end procedure;
```

(a)

```
procedure BUILD_SOL(M, r);
 if(r = n) then H^M ←< A_[r], π^M(A_[r]) >;
 if(r < m) then
  L_r = A^{M_h^M} ∈ H^{M_h^M} / π^M(A_[r] ∪ A^{M_h^M}) > c_inf;
  while(L_l ≠ ∅ and c_inf < 1) do
   take and delete A^{M_h^M} from L_l;
   A^[d] = A_[r] ∪ A^{M_h^M};
   BUILD_SOL(M, r + n^{M_h^M})
  end while;
 end if;
 else
  L_r = A_i^j ∈ P/π^M(A_[r] ∪ A_i^j) > c_inf;
  while(L_r ≠ ∅ and c_inf < 1) do
   take and delete A_i^j from L_r;
   A_[r+1] = A_[r] ∪ A_i^j;
   BUILD_SOL(M, r + 1);
  end while;
 end else;
end procedure;
```

(b)

Fig. 2. (a) Procedure that makes it possible to solve a MFTP hierarchically; (b) procedure that assembles the solutions of the MFTPs

where $\mathcal{M}_k^1$ is the MFTP from nesting level 1 that we are resolving and $\mathcal{R}^{[\mathcal{A}_{[r]},\mathcal{A}^{\mathcal{M}_h^o}]}$ is the set of constraints that only involve the points to which $\mathcal{A}_{[s]}$ assigns value, and which involve at least one point to which $\mathcal{A}_{[r]}$ assigns value, and at least one other to which $\mathcal{A}^{\mathcal{M}_h^o}$ assigns value; i.e. the set of constraints that are defined between points of the assembled subMFTP and those points that already have a value assigned to them. Thus, if $\mathcal{M}_k^1$ has points that are not linked to one of the subMFTPs, we continue to extend the tuple of assignations, in a similar manner to 2. Once all level 1 MFTPs have been completed, we go up to the next level and resolve its MFTPs by assembling solutions for its subMFTPs and assignations to points by means of expressions equivalent to 3 and 2, respectively, but which are defined between the nesting levels l and $l-1$. The process finishes when the global pattern is resolved. Throughout the entire search process we prune those branches in the search space that give rise to a solution with a degree of satisfaction lower than $c_{\inf}$.Figure 2b shows a procedure that searches for solutions for an MFTP by extending a locally valid tuple of assignations. The list L_r stores the solutions of the next subMFTP, or assignations to the next significant point, whose degree of local consistency with respect to the tuple that has been assembled up to this point is higher than $c_{\inf}$; the list $\mathcal{H}^{\mathcal{M}}$ stores solutions found for the MFTP $\mathcal{M}$ and its degree of satisfaction; n is the number of significant points of $\mathcal{M}$; $n^{M_h^M}$ is the number of significant points of $\mathcal{M}_h^{\mathcal{M}}$ and $m = \Sigma_{h=1}^{m} n^{M_h^M}$. After the execution of the MATCH procedure,

the solutions found for $\mathcal{M}$ and for each $\mathcal{M}_h^{\mathcal{M}}$ remain stored in $\mathcal{H}^{\mathcal{M}}$ and $\mathcal{H}^{\mathcal{M}_h^{\mathcal{M}}}$, respectively.

In the example of the P wave, in order to assemble solutions for $\mathcal{M}^{Sinusal}$, we start from an empty tuple of solutions, and we add to it a solution $\mathcal{A}_{P\_II}^{Sinusal}$. We then search for a solution $\mathcal{A}_{QR\_II}^{Sinusal}$ that satisfies the constraint L_{23}^{II}, and with this we construct a tuple of assignations, whose degree of consistency is calculated on the basis of 3. Finally, we search for a solution $\mathcal{A}_{P\_aV_R}^{Sinusal}$ that satisfies the constraint $L_{11}^{II\ aV_R}$, having then obtained a solution for $\mathcal{M}^{Sinusal}$. If, during the search, no solution is found for the following subMFTP that is compatible with the solutions that have been assembled up to that point, we backtrack to the previous subMFTP, and we attempt to assemble a different solution.

5.1 A Heuristics for Increasing Matching Efficiency

In order to guarantee the completeness of the search, we have to find all the possible solutions for the global pattern, regardless of their degree of satisfaction. This is due to the fact that in CSPs local solutions, even the optimal ones, do not necessarily form part of a global solution. This may significantly diminish the efficiency of the hierarchical matching.

In order to avoid searching for an excessive amount of solutions we have developed a heuristics that, by making use of the continuity properties of real signals, aims to obtain a set of solutions that is as representative as possible of each occurrence of a finding. The intuitive idea behind the heuristics is to attempt to obtain a "sampling" of the occurrence. In order to do so, we search for solutions within a temporal window whose length L_W must be greater than or the same as the maximum temporal extension of the MFTP, to guarantee that no solutions are lost. Thus, for example, in order to match $\mathcal{M}_{P\_II}^{Sinusal}$ the width of the temporal window must be the same or greater than the maximum value allowed by L_{12}^{II}. Each new solution must be better than the best solution up to that point, and from the set of solutions that are found, we only store those that differ more than d_{inf} from the ones that have already been stored. We define the distance between solutions as the Euclidian distance of the vectors formed by the temporal assignations to the significant points: $d(\mathcal{A}^1, \mathcal{A}^2) = \sum_{A_i^{1j} \in \mathcal{A}^1, A_i^{2j} \in \mathcal{A}^2} (t_i^{1j} - t_i^{2j})^2$, where $\mathcal{A}^1$ and $\mathcal{A}^2$ are two solutions of $\mathcal{M}$. When the data from this window have been processed, the window is shifted by a constant interval ΔL_W, and the process is repeated. In this way for each occurrence of a finding we obtain a set of solutions that are scattered around its temporal evolution. Searching in every window for better solutions than those already found considerably shortens the search space, as it avoids spreading out all the branches that would lead to a poorer solution than the best of those already found. Storing only those solutions that are significantly different limits the search space in the following stages of the matching, at the same time as it supplies reasonable guarantees of the stored solutions being representative of the temporal evolution of the system.

ΔL_W and d_{inf} act as algorithm control parameters: if they have high values less time will be needed to perform the matching, but less solutions will be

generated and stored, and there will be a higher probability of non-optimal solutions or of losing occurrences of the pattern. Using lower values results in more time being required for the search, but is less probable that solutions will be lost or that non-optimal ones will be found. The adjustment of these parameters depends on the system's dynamics: for systems that evolve slowly we use higher values, while in rapidly-evolving systems lower values are used. The most precise matching is obtained when $\Delta L_W = \Delta t$, Δt being the sampling period and d_{inf}=0, i.e. all the solutions found are stored.

Beyond this heuristics, common constraint satisfaction problems heuristics can be exploited to speed up the matching, as searching first for uncommon events of the pattern (a high value, a sharp peak, etc.).

6 Validation

The validation that is offered here should be considered as a proof of concept and of the efficiency of the algorithms, and not as an exhaustive validation of a medical pattern. This validation consists of detecting the sinusal rhythm pattern with a reading lasting a little over 10 minutes that has a total of 1,263 beats, and to do so we use the Tool foR anAlyzing and disCovering pattErns [4] (TRACE, see Fig. 3), with which clinical research on medical signals is currently being carried out. The results of the detection are contrasted with the visual detection made by a medical specialist; there were 1,129 true positives (TP), 102 true negatives (TN), 4 false positives (FP) and 32 false negatives (FN). In our evaluation, the TPs are detected P waves that belong to a beat from the sinusal basal rhythm, while their non-detection represents a FN. Electrical activity confusion that does not correspond to any beat results in a FP. Lack of confusion in the electrical activity by auricular extrasystole followed by an early QRS complex constitutes

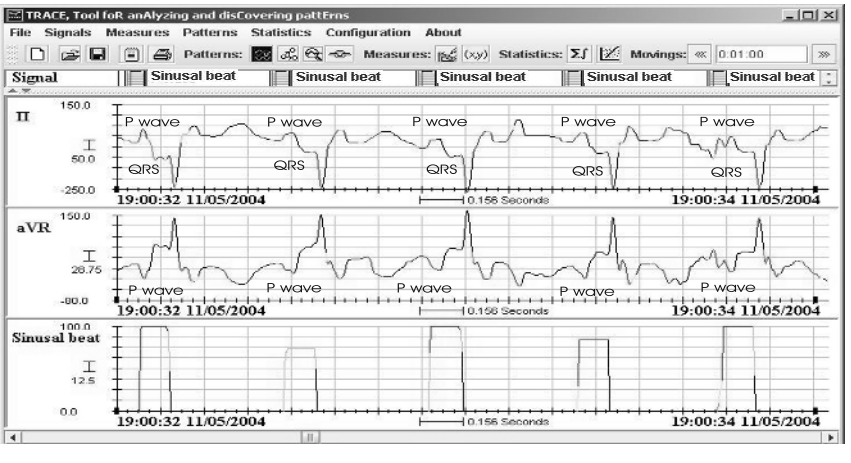


Fig. 3. Detection of the sinusal rhythm on the reading used in the validation

a TN. Thus we obtain a sensitivity of 0.917 and a specificity of 0.889, with a confidence interval of (95%,0.9–0.931) and (95%,0.747–0.956), respectively. In spite of dealing with a pattern of great physiological variability and having carried out the detection over a complex recording, without filtering, and being subjected to all kinds of artifacts, the results are highly satisfactory.

The detection, carried out on a Pentium III at 800 MHz, required 75 seconds. This proves that, in spite of the theoretically high computational complexity of the recognition algorithms, they can be used for real-time detection, even in difficult cases: the ECG was sampled at 250 Hz. This, along with the quality of the results, shows the validity of the recognition algorithms.

7 Discussion and Conclusions

We have presented a hierarchical matching procedure for signal abstraction. The use of fuzzy logic enables us to represent and handle the imprecision and uncertainty that are characteristic of human knowledge. Explicitly capturing the abstraction hierarchy that is present in the pattern simplifies the acquisition and revision of expert knowledge and allows the matching to be performed in as many levels of abstraction as the human expert employs to reason about the system, enabling the complete histories of findings of interest at each level of abstraction to be constructed. Thus, the human expert can be given detailed explanations as how the lower-level information has been combined in order to generate higher-level information. It is also more suitable for agent-based implementation, where each agent can take charge of matching each finding, using the results from the previous agents.

The heuristics that we have presented sacrifices the completeness of the matching algorithms. Nevertheless, it is highly probable that the solutions found will enable the global pattern, if it exists, to be found, since they act as a sampling of each occurrence of a finding. Using values for ΔL_W and d_{inf} that are adapted to the dynamics of the system the heuristics has never led to occurrences of the pattern being lost, and very rarely has it generated non-optimal solutions. On the other hand, the heuristics has made it possible to fully comply with real time requirements in the detection of patterns studied in the domains of patient supervision and mobile robotics.

With regard to future work, we aim to study the problem of network consistency: the description of the MFTP may be inconsistent, which means that there is no completely possible solution; hence the MFTP should be revised. Even when is consistent, it maybe possible to refine the knowledge projected onto a MFTP, which would result in a more efficient matching process.

References

1. R. Dechter. *Constraint Processing.* Morgan Kaufmann Publishers, 2003.
2. P. Félix, S. Barro, and R. Marín. Fuzzy constraint networks for signal pattern recognition. *Artificial Intelligence*, 148:103–140, 2003.

3. HJ. Marriott and MB. Conover. *Advanced Concepts in Arrhythmias*. Mosby, 1998.
4. A. Otero, S. Barro P. Félix, and F. Palacios. A tool for the analysis and synthesis of alarms in patient monitoring. In *FUSION2004*, pages 951–958, 2004.
5. A. Otero, CV. Rodríguez, J. Correa, M. Rodríguez, and P. Félix. A fuzzy constraint satisfaction approach for landmark recognition in mobile robotics. In *IPMU2004*, pages 183–191, 2004.
6. LA. Zadeh. The concept of a linguistic variable and its application to approximate reasoning. *Information Science*, 8:199–249, 1975.

A Meta-Reasoning Model for Hard Real-Time Agents*

Carlos Carrascosa, Andrés Terrasa, Ana García-Fornes, Agustín Espinosa, and Vicente Botti

Universidad Politécnica de Valencia
Departamento de Sistemas Informáticos y Computación (DSIC)
Camino de Vera s/n – 46022 Valencia – Spain
{carrasco, aterrasa, agarcia, aespinos, vbotti}@dsic.upv.es

Abstract. This paper defines a meta-reasoning model for real-time agents. The purpose of this model is to increase the adaptability in this kind of agents. This model allows to adapt the agent's behavior as well as the agent's own reasoning process. The application of this model to an specific real-time agent architecture is also presented.

1 Introducción

As Wooldridge & Parsons [1] states, "agents do not operate in isolation: they are situated in environments", considering the agent's environment as everything external to the agent.

So, a *Real-Time Agent –RTA–* is an agent situated in a real-time environment, where a *Real-Time Environment* is an environment with timing restrictions. If these timing restrictions are such that their unfulfillment may lead to catastrophic consequences, the agent is called *Hard Real-Time Agent –HRTA–*.

However, an agent is not only defined by being situated in an certain environment, but also by *how* the agent is situated within this environment. In the literature, a wide set of features have been used to describe this "how". The following list shows some of these features or capabilities (stated by authors as Franklin & Graesser [2] and Nwana [3]), contextualized in the case of RTAs:

- *Temporal Continuity*: taking into account the relevance and the criticality of some of the problems a RTA solves, the agent must work without a predetermined finalization.
- *Autonomy*: just for being an agent, a RTA must work autonomously. If a RTA possesses this capability along with the previous one, then it should never get *hanged* at any situation, nor depend on any user for taking decisions.
- *Rationality*: a RTA must be able to reason from its perceptions and to calculate the most adequate answers to its problem/environment.

* This work is partially funded by the Spanish goverment projects MCyT TIN2005-08945-C06-06 (FEDER) and TIC2003-07369-C02-01.

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 42–51, 2006.

- *Reactivity*: a RTA must react to changes in its environment, probably before some deadline. In order to be able to obtain this ability, the computational cost of the code calculating the answer must be controlled.
- *Proactivity*: a RTA must try to pursue its own goals. One of these goals is to fulfill its timing restrictions, specially if they are critical.
- *Adaptability*: a RTA must be prepared to adapt itself to changes in the environment that make the problem solving performed by the agent up to that moment difficult or even impossible.

Among all these features, adaptivity is fundamental when talking about RTAs, because they have to solve problems that may be critical, whatever happens in their environments. That is, a RTA needs to adapt itself to any possible environmental change to be able to continue solving the problem it is in charge of.

This paper introduces the option of using meta-reasoning capabilities to a real-time agent architecture in order to improve the agent adaptivity.

It is important to mention that a real-time agent may follow a reactive, deliberative or hybrid architecture, but it is only in the latter case when questions as the influence of a meta-reasoning process, as the one here presented, arise. So, the results presented in this paper are mainly applicable to hybrid real-time agent architectures.

The paper is structured as follows: section 2 briefly introduces the concept of meta-reasoning, next section describes the model of meta-reasoning for a RTA object of the paper; after that, section 4 presents the application of such meta-reasoning model to a particular RTA architecture (ARTIS Agent); last section presents the conclusions of the paper and some ongoing work.

2 Meta-Reasoning

In its widest meaning, and according to [4] definition, *meta-reasoning* is any process interested in the operation of another computational process in the same entity. This term is related to a concept distinction between object-level deliberation about external entities (e.g, to consider the benefits of the different opening movements in a chess game) and meta-level deliberation about internal entities (e.g., to decide that deliberating about which opening movement to use lacks of interest) [4]. In other words, and according to the Raja and Lesser [5] nomenclature, an agent may realize two kinds of decisions:

- Meta or macro-level decissions, managed by the meta-level controller.
- Planning or micro-level decisions, managed by the domain level controller.

The meta-level controller must be designed to take quick and cheap decisions about how many resources should be expended in domain actions and how many in control actions. The initial control decisions are also classified, according to [5], in:

- *Coordination Decisions*: They control if the agent is coordinated with other agents and how much effort should be expended in coordination.

- *Planning Decisions*: They control if the domain level planner is used and how much effort should be expended by the planner.
- *Slack Decisions*: They will prescribe how many spare time or *slack* should be included in a plan to deal with unexpected events.

It has to be taken into account that, as [5] states, an agent is not behaving rationally if by the moment it has calculated an action, it is already too late to apply this action. So, an agent should plan (and/or coordinate with other agents) just when the expected improvement outweighs the expected cost. If meaningful quantities of resources are expended in realizing this meta-decision, then the *meta-meta-decisions*[1] about if these resources must be expended should be made in the context of the global increasing of the system utility. To do this, an agent should know beforehand the effect of every combination of actions, which is intractable for any problem of a reasonable size. The problem of how to approximate this ideal of sequencing domain and control activities without consuming too much resources in the process is known as the *meta-level control problem* for limited-resource rational agents.

Taking into account that the meta-reasoning model looks for the balance between doing any atomic computation and executing a *real* action affecting the environment [6], two effects in the agent utility must be considered when evaluating each computation:

- It spends time, and so, it may incur in an opportunity cost. Specifically, this computation will cause that the agent postpone the execution of the next *real* action for, at least, the duration of a computational step.
- A computation will have some effect over the real actions chosen by the agent.

There are two possible outcomes of a computation. The simplest one is the changing of the action the agent believes to be the best one. The second and most difficult to analyze occurs when the computation does not really cause a change in the action choice, but it adds some information to the agent' state. On the other hand, this additional information would cause later computations to change the action election. This is called *indirect utility* of the computations.

Some meta-reasoning examples can be found in real-time agents literature, such as SA-CIRCA [7,8]. SA-CIRCA has the peculiarity that the real-time system solving the problem does not compete for the resources with the AI system in charge of the meta-reasoning, since they are executed in different processors.

3 Meta-Reasoning Model for RTA

The availability of a meta-reasoning process in a RTA is of great importance since it allows the agent to make the best possible from its available resources, specially considering that these resources are limited, and that the most important resource in any real-time system is the processor time itself.

[1] Decisions about meta-decisions.

The solution to the so-called *meta-level control problem*, that will prevent the meta-reasoning process to fall in useless digressions, is even more important in RTAs than in traditional AI systems, since answers to specific problems must be obtained before a deadline.

As stated above, the meta-reasoning process is responsible of obtaining the real-time agent adaptability. To do this, a *cause-effect* model is now presented. This model specifies the meta-reasoning process to be carried out by the RTA by means of *Meta-Rules* ($\mu Rules$). Such $\mu Rules$ are composed by a condition specifying the *meaningful situation* of change the RTA may face, and a set of meta-reasoning actions that will allow the agent to adapt itself to this situation.

A *situation* is defined by an environment state along with the internal state of the agent, defined by its believes. This implies that the agent will be able to adapt to changes in both the environment and its internal state.

Following to this, a *meaningful situation* of change can be defined as a situation against which the agent will not be able to answer in the same way it was doing until now. That is, it is a situation that oblige the agent to adapt itself in order to be able to face it.

This meta-reasoning model has two capabilities: the adaptation of the agent behavior and the adaptation of the agent reasoning process. Such capabilities will be specified inside the set of actions allowed in the $\mu Rules$.

In a more formal way, the meta-reasoning process will be carried out by an agent module called *Meta-Control* ($\mu Control$) that is defined as:

$$\mu\mathcal{C}ontrol = \langle \mathcal{B}_\mu, G_\mu, A_\mu, \mathcal{AFS}et \rangle$$

where:

- $\mathcal{B}_\mu$, is the set of believes of the $\mu Control$. These believes are not observable, and will reflex knowledge about the reasoning process. This is the reason why the expression $\mathcal{B} \cap \mathcal{B}_\mu = \emptyset$ holds (it is a belief set independent of the believes of the agent).
- G_μ, $\mu Control$ own objectives. These objectives are related both to the improvement of the efficiency in the RTA reasoning process, and to the adaptability of the agent to the environment changes. These objectives are in a different abstraction level than the RTA objectives, so $G \cap G_\mu = \emptyset$.
- A_μ, set of actions that will carry out the meta-reasoning. Through these actions, the *Meta-Control* changes the way that the RTA tries to solve the problem.
- $\mathcal{AFS}et$ set of *Attention Focuses*.

3.1 Adapting the Behavior

Behavior Definition. A behavior controls the way the agent solves its problems. The meta-reasoning model restricts the agent to have only one active behavior simultaneously. So, for instance, if the agent controlled an airplane, it would show a different active behavior during the take off, the flying and the landing.

Due to this, the design of the agent's has to include the specification of the different behaviors available for the agent, along with the possible transitions among them. In the model, this is represented by a non-deterministic finite automaton where the nodes will be the agent behaviors and the arcs will be the possible transitions. Each one of these transitions will be modeled as a $\mu Rule$, which condition will be the situation causing that transition and which action will be the change of the agent active behavior. It is important to note that the behavior change automaton is non-deterministic because real-time environments are also non-deterministic.

3.2 Adapting the Reasoning Process

In real-time systems, the most important resource is the processor because the system has to ensure that certain computations need to meet timing constraints. This aspect is specially problematic in Real-Time Artificial Intelligent systems, since they normally introduce algorithms with very high (sometimes unbounded) processor costs. Moreover, if the system works under hard timing restrictions, that must be met under all circumstances, the optimal management of this resource becomes a fundamental aspect of the system design.

However, it is impossible to make an optimal use of the processor when the computations can be unbounded, since it may happen that, after consuming the processor time that can be allocated for a determined reasoning process, it may not have gotten any calculation with utility for the system. In this sense, a mechanism that allows for the best possible use of the processor given the current situation becomes relevant.

The meta-reasoning model proposes to adapt the reasoning process as a way to efficiently use the processor time on line. This is achieved by introducing two new concepts: the *reactivity degree* and the *attention focus*, described in the following sections.

Changeable Reactivity: Reactivity Degree. Reactivity is an essential feature of an agent. Moreover, according to [9], a possible classification of agent architectures, divide them in reactive, deliberative and hybrid. In fact, the difference among these three kinds of architectures is not if one of them presents a reactive agent and the others do not, but in *how much* reactive each kind is versus the others.

So, reactivity will be present in different ways in a deliberative agent than in a reactive one. In a reactive agent, reactivity will be bigger and its response to an stimulus will be faster and more direct than the ones in a deliberative agent. On the other hand, in a deliberative agent, answers are more elaborated and therefore, a bigger amount of time is needed to act. In this way, each of the different agent architectures present a fixed reactivity degree that may work well in a given situation, but probably not in others. Therefore, it is intuitively perceived that reactivity is not an absolute feature, but there exists a *quantitative* difference between the reactivity of the different agent architectures; hence reactivity can be graded.

From here, it can be concluded that an agent architecture able to change its reactivity degree will be able to adapt better than traditional agent architectures, and it will also work more in a human-like way. For instance, when a person sees a burning paper that he does not want to lose, he may dedicate a moment to think about how to avoid that it will be consumed by the fire. On the other hand, if his hand is burning, he will take it out of the fire immediately, without dedicating just a little moment to think about what to do. This simple example presents a human acting according to two different reactivity degrees depending on the situation he is in. This basic idea is the starting point for the aspect presented in this section: the management of a variable reactivity degree.

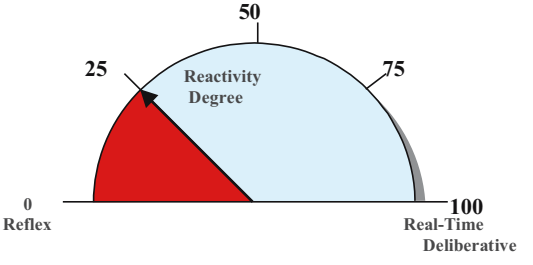


Fig. 1. RTA's Reactivity Degree

The *Reactivity Degree* of an agent is a real value between 0 and 1 indicating how much effort the agent is going to dedicate to deliberate. This degree defines two extreme situations with infinite intermediate states (figure 1). These extreme situations are:

- If the reactivity degree is 0, the agent works in **Reflex Mode**: It does not spent time in deliberating, in improving the first answer it gets. This is the reactivity degree of agents with reactive architecture.
- If the reactivity degree is 1, the agent works in **Deliberative Mode**: It dedicates all the time it has to its disposal to calculate the best answer to its problem. It is the reactivity degree of agents with deliberative architecture.

The capability of adapting the agent's reactivity is even more important if such agent works in a hard real-time environment, since in this case, deliberative processes can not be arbitrarily extended if the agent wants to meet its timing constraints. Therefore, RTAs cannot be pure deliberative agents (working in **Deliberative Mode**). If a RTA has a value of 1 in its reactivity degree, the agent will work in **Real-Time Deliberative Mode** instead of **Deliberative Mode**, dedicating not all its time, but all its *slack time*, to the deliberative process.

Variable Introspection: Introspection Degree. General psychology defines *extrospection* as [10] *"the method of psycological observation that is not based in the analysis of the consciousness contents but in the ones derived of the*

senses". This concept is opposed to *introspection* [10], that is defined as *"the self-observation method that is realized in a controlled way, under experimental conditions. The object of knowledge is the mental content of the person who is performing the observation"*.

In a similar way, when studying the deliberative process of an agent, it can be distinguished between the deliberation motivated as an answer to environment changes and the deliberation about the agent's own objectives (based on the agent's proactivity).

In this way, the meta-reasoning model defines the *Introspection Degree* as a value between 0 and 1, indicating the fraction of time that the agent is devoting to attend the environment ("extrospection") versus the time the agent devotes to deliberation ("introspection"). In particular, when the value of the *introspection degree* approaches 0, the agent is more sensitive to the environment; when the *introspection degree* value is closer to 1, the agent is more self-centered or *introspective*.

In a way similar to the reactivity degree, the *introspection degree* defines two extreme situations in the deliberation process of an agent with infinite intermediate situations. These extreme situations are:

- If the *introspection degree* is 0, the agent's deliberation is *extrovert*, all the deliberation process is motivated by the agent's environment changes.
- If the *introspection degree* is 1, the agent's deliberation is *introvert*, the agent is not interested in the environment changes, and all the deliberation process is motivated by its own interests or objectives.

Variable Concentration: Attention Focus. The *Concentration* concept is defined from the general psychology point of view as [10] *"the capability of focusing in certain stimuli, discarding the rest not related to them. The purpose of this is to focus the attention and to optimize the reflection. When attention is focused, the awareness field is narrowed"*. On the same way, general psychology considers the *Attention* concept as [10] *"the selective concentration of mental activity that implies an efficiency increase of an specific task, as well as a perceptual and cognitive inhibition of the rest of tasks. Attention is not a behavior on its own, but it qualifies specific moments in a global behavior, at which the persistence of the contact between the subject and the object predominates. Such object may be extern, and in such case it is called external action, or internal, centered in a state of the subject themselves (introspection)"*.

The objective of this capability is to apply the previous concepts to the agent architecture. So, a way to increase the efficiency of the reasoning process consists of being able of *focusing* this process; that is, to reduce the field of its reasoning, in order to use the available processor time only in the aspects that are relevant to the current situation. This can be named as changing the *concentration* of the agent.

In this way, the *attention focus* is defined as a subset of agent's believes that are meaningful in the current situation. As the *concentration* concept defined above implies, these belief subsets will be used to optimize the reasoning and

meta-reasoning processes, prioritizing the reasoning about the attention focus or focuses that are *active* in a particular moment.

As defined in the meta-reasoning model, the focusing mechanism can attend several focuses simultaneously. In this case, each one of these active attention focuses is defined to have a different *attention degree* (according to the relative importance that this focus has with respect to the other active focuses).

In a more formal way, the set of attention focuses of $\mu Control$ is defined as $\mathcal{AFS}et = \{\mathcal{AF}_i / \mathcal{AF}_i = \{\mu\mathcal{R}_j \in \mu\mathcal{R}ule\mathcal{S}et\}\}$, where each *Attention Focus* is defined by a set of *Meta-Rules*.

A *Meta-Rule* ($\mu Rule$) is defined as $\mu\mathcal{R}_j : 2^G \times 2^{B \cup \mathcal{B}_\mu} \rightarrow \{A_\mu\}^+$, that is, a function defining the answer of the *Meta-Control* to certain situations (defined by the current objectives of the RTA along with a set of believes —from the domain and the *Meta-Control*—).

It has to be emphasized that though this previous definition seems to change the concept of attention focus for defining it by a set of $\mu Rules$, it is not the case. As each one of these $\mu Rules$ is associated to a believes subset included in its situation of activation, an attention focus is implicitly also associated to a belief subset.

3.3 Meta-Meta-Reasoning

The meta-reasoning model of a RTA must be complemented / completed with a meta-meta-reasoning process in charge of adapting the meta-reasoning process. This process must be based on an on-line learning algorithm allowing to use the agent's own experience to refine, to learn and even to forget $\mu Rules$.

4 Applying the Model to a Particular Real-Time Architecture: ARTIS Agent

The meta-reasoning model presented in this paper has been realized into a real agent architecture called ARTIS Agent ($\mathcal{AA}$) [11]. This is a hybrid, vertical, real-time agent architecture which is organized in different abstraction levels or *models*. In particular, there are two models: user model and system model. The user model presents a high-level abstraction view of the architecture that facilitates the development of the $\mathcal{AA}$. The system model is the low-level, executable model of the agent in a particular real-time operating system (Real-Time Linux, in this case). The architecture also provides an automatic translation process that converts a user model specification into a runnable version of the agent in terms of the system model, without user intervention.

The inclusion of the meta-reasoning model in this architecture has involved modifications in both models. In the user model, a new language, called the *control language* [12], has been introduced in order to specify the meta-reasoning particular to each agent. On the other hand, the system model has been extended with (1) an event-based mechanism that allows for the detection of significant situations at run time, and (2) specific mechanisms which adapt the behavior

of the agent in respond to such situations. In this latter case, the mechanisms include the ability to change the run-time mode [13] and to change the reasoning process[14,12].

The incorporation of the learning mechanisms in charge of the meta-meta-reasoning process is currently in progress.

5 Conclusions

This paper has presented a meta-reasoning model for real-time agents with hard temporal restrictions, and therefore generalizable to agents with softer (or none) time requirements. The main objective of this model is to enhance the agent adaptivity, which is commonly accepted as one of the key traits of agents. In order to achieve such objective, the model proposes two adaptivity abilities: the adaptation of the agent behavior and the adaptation of the reasoning process. These two abilities are realized by introducing the concepts of *reactivity degree* and *introspection degree*, respectively, into the agent architecture.

The meta-reasoning model has not only been designed, but it has also been implemented in a hard real-time agent architecture named ARTIS Agent.

Current and future work includes the development of meta-meta-reasoning aspects inside the model (and the implementation of their corresponding mechanisms inside the ARTIS Agent architecture), as well as the extension of the model to cope with real-time multi-agent systems such as the SIMBA platform [15,16].

References

1. Wooldridge, M., Parsons, S.: Intention reconsideration reconsidered. Intelligent Agents V: Agent Theories, Architectures, and Languages (LNAI) (1998) 63–79
2. Franklin, S., Graesser, A.: Is it an agent, or just a program?: A taxonomy for autonomous agents. In: Proceedings of the Third International Workshop on Agent Theories, Architectures, and Languages, Springer-Verlag (1996)
3. Nwana, H.S.: Software agents: An overview. Technical report, Intelligent Systems Research. AAT, BT Laboratories, Ipswich, United Kingdom (1996)
4. Russell, S.: Metareasoning. In The MIT Encyclopedia of the Cognitive Sciences, MIT Press (1998)
5. Raja, A., Lesser, V.: Real-time meta-level control in multi agent systems. In: Proceedings of Multi-Agent Systems and Applications - ACAI 2001 and EASSS 2001 Student Sessions. Also Adaptability and Embodiment Using Multi-Agent Systems: AEMAS 2001 Workshop. Prague, Czech Republic. (2001)
6. Russell, S., Wefald, E.: Principles of Metareasoning. In: First International Conference on Principles of Knowledge Representation and Reasoning. Morgan Kaufmann Publishers, Inc. (1989) 400–411
7. Goldman R.P., Musliner, D.J., Krebsbach, K.D.: Managing online self-adaptation in real-time environments. In: Proc. of Second International Workshop on Self Adaptive Software, Balatonfured, Hungary (2001)
8. Musliner, D.J.: Safe learning in mission-critical domains: Time is of the essence. In: Working Notes of the AAAI Spring Symposium on Safe Learning Agents, Stanford, California (2002)

9. Wooldridge, M., Jennings, N.R.: Intelligent agents: Theory and practice. The Knowledge Engineering Review **10** (1995) 115–152
10. Larousse, B.d.C.: Diccionario de Psicología. SPES Editorial, S. L. (2003)
11. Botti, V., Carrascosa, C., Julián, V., Soler, J.: Modelling agents in hard real-time environments. In: MAAMAW'99 Proceedings. Volume 1647 of LNAI., Springer-Verlag (1999) 63–76
12. Carrascosa, C., Fabregat, J., Terrasa, A., Botti, V.: Real-time agents: Reaction vs. deliberation. In: Agents in dynamic and real-time environments Workshop. (2004) 63–70
13. Carrascosa, C., Terrasa, A., Fabregat, J., Botti, V.: Behaviour management in real-time agents. In: 5th Iberaoamerican Workshop on Multi-Agent Systems. (2004) 1–11
14. Carrascosa, C., Rebollo, M., Julián, V., Botti, V.: Deliberative server for real-time agents. In: Multi-Agent Systems and Applications III: 3rd International Central and Eastern European Conference on Multi-Agent Systems, CEEMAS 2003. Volume 2691 of LNAI., Springer (2003) 485–496
15. Soler, J.: SIMBA: Una Plataforma para el desarrollo de Sistemas Multiagente en entornos de Tiempo Real. PhD thesis, Departamento de Sistemas Informáticos y Computación. Universidad Politécnica de Valencia (2003)
16. Carrascosa, C., Rebollo, M., Soler, J., Julián, V., Botti, V.: Simba architecture for social real-time domains. In: EUMAS 2003: The First European Workshop on Multi-Agent Systems. (2003)

A Scheduling Order-Based Method to Solve Timetabling Problems

L. Ingolotti[1], F. Barber[1], P. Tormos[2], A. Lova[2], M.A. Salido[1], and M. Abril[1]

[1] DSIC, Polytechnic University of Valencia, Spain
{lingolotti, fbarber, msalido, mabril}@dsic.upv.es
[2] DEIOAC, Polytechnic University of Valencia, Spain
{ptormos, allova}@eio.upv.es

Abstract. We propose an efficient method that obtains train timetables. It solves conflicts among trains by assigning priorities for each track section of their journey. The heuristic used to determine the priority for each train, takes into account the objective function of the problem. With this method, we try to explore different regions of the same search space as soon as possible so that the final user obtains a feasible solution in a reasonable computation time.

1 Introduction

Train timetabling is a difficult and time-consuming task, with high number of very complex constraints. Cordeau et al. [1] point out the stronger competition, privatization, deregulation, and increasing computer speed as reasons for the more prevalent use of optimization techniques in the rail industry. Published methods and models indicate that the majority of authors use models that are based on the Periodic Event Scheduling Problem (PESP) introduced by Serafini and Ukovich [6]. The PESP considers the problem of scheduling a set of periodically recurring events under periodic time window constraints. The model generates disjunctive constraints that may cause the exponential growth of the computational complexity of the problem depending on its size. Schrijver and Steenbeek [4] have developed `CADANS`, a constraint programming-based algorithm to find a feasible timetable for a set of PESP constraints. The scenario considered by this tool is different from the scenario that we used; therefore, the results are not easily comparable. Nachtigall and Voget [3] also use PESP constraints to model the cyclic behavior of timetables and consider the minimization of passenger waiting times as the objective function. Their solving procedure starts with a solution that is obtained in a way similar to the one that timetable designers in railway companies use. This initial timetable is then improved using a genetic algorithm. In our problem, the waiting time for connections is not taken into account because we only consider the timetabling optimization in a single railway line. The train scheduling problem can also be modeled as a special case of the job-shop scheduling problem (Silva de Oliveira [7], Walker et al. [9]), where train trips are considered *jobs* that will be scheduled on tracks that are regarded

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 52–61, 2006.

as *resources*. The majority of these works consider the scheduling of new trains on an empty network. However, infrastructure management railway companies usually also need to optimize new trains on a line where many trains are already in circulation (that is, trains that have a fixed timetable). With this main objective, Lova et al. [2] propose a scheduling method based on *reference stations* where the priority of trains, in the case of conflict, changes from one iteration to another during the solving process.

2 Problem Specification

Given a railway line which may have both single-track sections or double-track sections, the problem consists in adding new trains to it. The railway line may be occupied by other trains whose priority is higher than the new trains, and these new trains may belong to different operator types. Their respective journeys may also be different from each other. The timetable given to each new train must be feasible, that is, it must fulfill the set of constraints defined in Section 2.2. Besides feasibility, two additional goals are pursued by this work: computational efficiency and optimality, which is measured according to the objective function defined in Section 2.3.

2.1 Notation

1. T finite set of trains considered in the problem.
2. $\mathrm{T_C}$ set of trains that are in circulation and whose timetable cannot be modified. $\mathrm{T_C} \subset \mathrm{T}$.
3. $\mathrm{T_{new}}$ set of trains that does not yet have a timetable and must be added to the railway line with a feasible one. $\mathrm{T_{new}} \subseteq \mathrm{T}$.
4. l_i location (station, junction, halt, siding).
5. N_i number of tracks in location l_i.
6. $\mathrm{N_{P}}_i$ number of tracks with platform in location l_i.
7. $\mathrm{L} = \{l_0, l_1, ..., l_\mathrm{m}\}$ ordered sequence of locations that may be visited by trains $t \in T$. The locations l_i and l_{i+1} are linked by a single or double track section, and there is not a location between them that can be visited by any train t.
8. $\mathrm{J}_t = \{l_0^t, l_1^t, ..., l_{n_t}^t\}$ journey of t. It is described by an ordered sequence of locations to be visited by a train t such that$(\forall t)_\mathrm{T}(\exists \mathrm{J}_t)(\mathrm{J}_t \subseteq \mathrm{L})$. The journey J_t shows the order that is used by t to visit a given set of locations. Thus, l_i^t and $l_{n_t}^t$ represent the i_{th} and *last* location visited by t, respectively.
9. $\mathrm{T_D}$ set of trains traveling in the *down* direction.
$t \in \mathrm{T_D} \leftrightarrow ((\forall l_i^t)_{0 \le i < n_t}(\exists l_j)_{\{\mathrm{L} \setminus \{l_m\}\}} l_i^t = l_j \wedge l_{i+1}^t = l_{j+1})$.
10. $\mathrm{T_U}$ set of trains traveling in the *up* direction.
$t \in \mathrm{T_U} \leftrightarrow ((\forall l_i^t)_{0 \le i < n_t}(\exists l_j)_{\{\mathrm{L} \setminus \{l_0\}\}} l_i^t = l_j \wedge l_{i+1}^t = l_{j-1})$.
11. C_i^t minimum required time to do commercial operations such as boarding or leaving passengers (commercial stop).
12. $\mathit{\Gamma}_t$, time that is added to the running time of one train when it stops in a station S where no commercial stop was planned. The running time of the

train is increased in the previous and next track sections to S due to the train braking. It was not planned at the time the running time in these track sections was specified.

13. φ_k minimum required time that must exist between two consecutive trains, that travel in the same direction, in the track section determined by station k and the next one.
14. $\Delta^t_{i\rightarrow(i+1)}$ running time for train t from location l^t_i to $l^t_{(i+1)}$.
15. $\mathrm{I}^t_{\mathrm{L}}/\mathrm{F}^t_{\mathrm{L}}$ lower bound for the departure/arrival time of train $t \in \mathrm{T}_{\mathrm{new}}$ from/to the initial/final station of its journey.
16. $\mathrm{I}^t_{\mathrm{U}}/\mathrm{F}^t_{\mathrm{U}}$ upper bound for the departure/arrival time of train $t \in \mathrm{T}_{\mathrm{new}}$ from/to the initial/final station of its journey.
17. `CONS` set of constraints that specify the requirements that must be satisfied by a timetable in order to be considered feasible.
18. dep^t_i departure time of train $t \in T$ from the location i, where $i \in J_t \setminus \{l^t_{n_t}\}$.
19. arr^t_i arrival time of train $t \in T$ to the location i, where $i \in J_t \setminus \{l^t_0\}$.

2.2 Feasibility of a Solution - Set of Constraints

A solution is feasible if and only if satisfies the set of constraints `CONS`. In this subsection, these constraints are described.

– *Interval for the Initial Departure*: each train $t \in \mathrm{T}_{\mathrm{new}}$ should leave its initial station l^t_0 at a time dep^t_0 such that,

$$\mathrm{I}^t_{\mathrm{L}} \leq dep^t_0 \leq \mathrm{I}^t_{\mathrm{U}} \ . \tag{1}$$

– *Interval for the Arrival Time*: each train $t \in \mathrm{T}_{\mathrm{new}}$ should arrive to its final station $l^t_{n_t}$ at a time $arr^t_{n_t}$ such that,

$$\mathrm{F}^t_{\mathrm{L}} \leq arr^t_{n_t} \leq \mathrm{F}^t_{\mathrm{U}} \ . \tag{2}$$

– *Running Time*: an order to compute the timetable in each track section of a train $t \in \mathrm{T}_{\mathrm{new}}$, is determined by the following expression

$$arr^t_{i+1} = dep^t_i + \Delta^t_{i\rightarrow(i+1)} \ . \tag{3}$$

– *Commercial Stop*: each train $t \in \mathrm{T}_{\mathrm{new}}$ is required to remain in a station l^t_i at least C^t_i time units

$$dep^t_i \geq arr^t_i + \mathrm{C}^t_i \ . \tag{4}$$

– *Maximum Delay*: for each train $t \in \mathrm{T}_{\mathrm{new}}$ a maximum delay Λ_t, and a minimum running time M_t (15) are specified; then, the upper bound for the running time of $t \in \mathrm{T}_{\mathrm{new}}$ is given by the following expression

$$\frac{(arr^t_{n_t} - dep^t_0 - \mathrm{M}_t)}{\mathrm{M}_t} \leq \Lambda_t \ . \tag{5}$$

– *Headway Time*: if two trains, $\{t_i, t_j\} \subseteq \mathrm{T}$, traveling in the same direction leave the same location k, they are required to have a difference in departure times of at least φ_k

$$|dep_k^{t_i} - dep_k^{t_j}| \geq \varphi_k \ . \tag{6}$$

– *Finite Capacity of Stations*: a train $t \in \mathrm{T_{new}}$ could arrive to a location l_i^t if and only if it has at least one available track (with platform, if $C_i^t > 0$). Thus, where $T_x = \{t/t \in \mathrm{T} \wedge t \neq x \wedge \mathrm{J}_t \cap \mathrm{J}_x \neq \varnothing\}$;

$$(\forall x)_{\mathrm{T_{new}}}(\forall l)_{\mathrm{J}_x}(\sum_{t \in \mathrm{T}_x} \mathrm{Simult}(x,t,l) + \sum_{t \in \mathrm{T}_x} \mathrm{Simult_P}(x,t,l) < \mathrm{N}_l) \wedge$$

$$(\mathrm{C}_l^x > 0 \rightarrow \sum_{t \in \mathrm{T}_x} \mathrm{Simult_P}(x,t,l) < \mathrm{N}_{\mathrm{P}_l}) \ . \tag{7}$$

`if` $[arr_l^x, dep_l^x] \cap [arr_l^t, dep_l^t] \neq \varnothing \wedge C_l^t = 0$, `then` $\mathrm{Simult}(x,t,l) = 1$, `else` $\mathrm{Simult}(x,t,l) = 0$;

`if` $[arr_l^x, dep_l^x] \cap [arr_l^t, dep_l^t] \neq \varnothing \wedge C_l^t > 0$, `then` $\mathrm{Simult_P}(x,t,l) = 1$, `else` $\mathrm{Simult_P}(x,t,l) = 0$

– *Closing Time*: Let $[\mathrm{H}_l^1, \mathrm{H}_l^2]$ be the closing time of a station l. We consider two types of closing; A, see the next expression; and B where the number of tracks in the station is decreased to one (see (7))

$$dep_l^t < H_l^1 \vee arr_l^t > H_l^2 \ . \tag{8}$$

– *Crossing*: according to the following expression, a single-track section ($i \rightarrow j$, *down* direction) cannot be occupied by two trains going in the opposite directions ($t \in \mathrm{T_D}$ and $t' \in \mathrm{T_U}$).

$$dep_j^{t'} > arr_j^t \vee dep_i^t > arr_i^{t'} \ . \tag{9}$$

– *Overtaking on the track section*: considering that the trains that travel in the same direction go for the same track into a given section, overtaking must be avoided between any two trains, $\{i, j\} \subseteq T$, going in the same direction on any common track sections, $k \rightarrow (k+1)$, of their journeys:

$$(arr_{k+1}^i > arr_{k+1}^j) \leftrightarrow (dep_k^i > dep_k^j) \ . \tag{10}$$

– *Overtaking in the station*: the expression below specifies that a train is not allowed to overtake another train in a station if: $\{i, j\} \subseteq \mathrm{T_{new}}$; both trains are traveling in the same direction; and they belong to the same type of train.

$$arr_l^i < arr_l^j \leftrightarrow dep_l^i < dep_l^j \ . \tag{11}$$

It is allowed the overtaking between two trains into a station if one of these trains is train in circulation because its timetable can not be modified. The overtaking in one station will be feasible only if it has enough available tracks.

– *Delay for an unexpected stop*: when a train t stays in a station j where no commercial stop was planned ($\mathrm{C}_j^t = 0$), the running time of t that corresponds to the previous ($l_i^t \rightarrow l_j^t$) and next ($l_j^t \rightarrow l_k^t$) track sections of j must be increased by Γ_t time units.

$$dep_j^t - arr_j^t > 0 \wedge C_j^t = 0 \leftrightarrow \Delta_{i\rightarrow j}^t = \Delta_{i\rightarrow j}^t + \Gamma_t \wedge \Delta_{j\rightarrow k}^t = \Delta_{j\rightarrow k}^t + \Gamma_t \ . \quad (12)$$

– *Reception Time*: The difference between the arrival times of any two trains $\{t', t\} \subseteq T$ in a same station l is defined by the expression below, where R_t is the reception time specified for the train that arrives to l first.

$$arr_l^{t'} \geq arr_l^t \rightarrow arr_l^{t'} - arr_l^t \geq \mathrm{R}_t \ . \quad (13)$$

– *Expedition Time*: The difference between the departure and arrival times of any two trains $\{t', t\} \subseteq T$ in the same station l is defined by the expression below, where E_t is the expedition time specified for t.

$$|dep_l^{t'} - arr_l^t| \geq \mathrm{E}_t \ . \quad (14)$$

The problem is formulated taking into account that $\forall t \in \mathrm{T_C}$ and $\forall i \in \mathrm{J}_t$, the variables arr_i^t and dep_i^t in `CONS` are instantiated according to the trains in circulation timetables. Thus, these variables are considered as constants in the constraints by the solving process.

2.3 Optimality of a Solution - Objective Function

We have defined the optimal time of a train t, Γ_{opt}^t, as the lowest time employed by t to complete its journey, satisfying all problem constraints in `CONS` but only taking into account trains in circulation. The rest of the trains belonging to $\mathrm{T_{new}}$ are ignored. Our quality criterion for this study is the minimization of the average delay, $\delta = \frac{\sum_{t\in \mathrm{T_{new}}} \delta_t}{|\mathrm{T_{new}}|}$, where $\delta_t = \frac{arr_{n_t}^t - dep_0^t - \Gamma_{\mathrm{opt}}^t}{\Gamma_{\mathrm{opt}}^t}$
Therefore, the objective function is to minimize the average deviation with respect to the optimal solution (ADOS): ADOS = MIN (δ). If there are no trains in circulation in L ($\mathrm{T} = \mathrm{T_{new}}$), the optimal time of a train t would be:

$$\Gamma_{\mathrm{opt}}^t = \sum_{i=0}^{n_t-1} \Delta_{i\rightarrow(i+1)}^t + \sum_{i=1}^{n_t-1} C_i^t \ . \quad (15)$$

3 A Scheduling Order-Based Method (SOBM)

The problem that was described in Section 2 is a search problem with exponential cost. We do not use uniform search, i.e. depth-first search, because it would be inefficient due to the size of the state space. We use an irrevocable heuristic-driven search. The intermediate states are discarded because keep them would be unfeasible in terms of spatial cost.

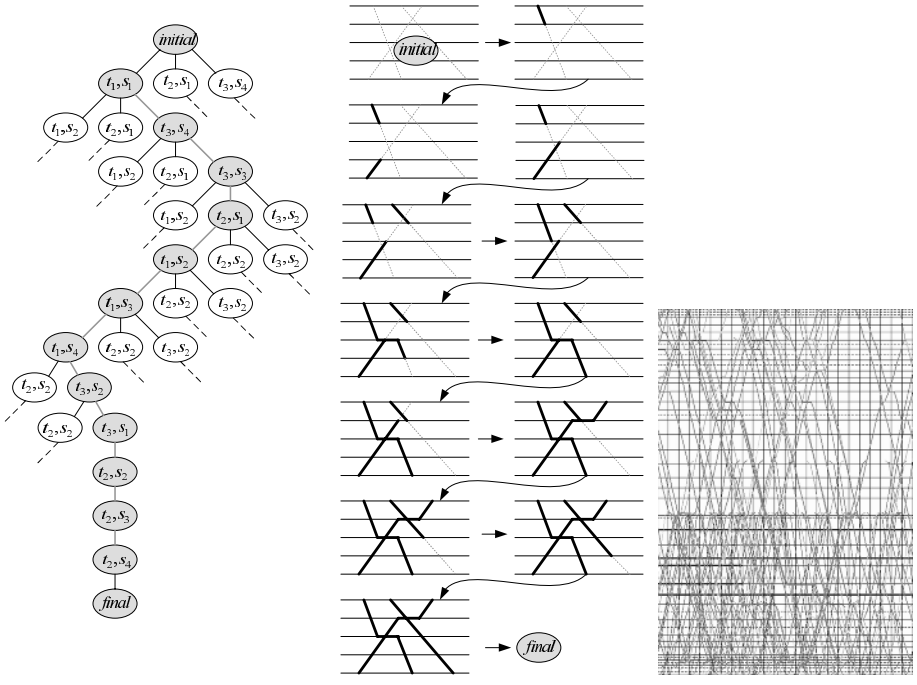
In this problem, one solution is different from others when a same conflict is solved in different way. A conflict exists when a constraint, that references the timetable of two trains, is violated. We solve each conflict by delaying one of these trains until the conflict disappears. Therefore, the problem consists in deciding which of the two trains must be delayed.

```
Function Get_Train_Timetabling() As Timetabling
begin
  Set_Occupation_Fixed_Trains (T_C)
  While Not (end_cond) do
  begin
      T_open = T_new; T_closed = ∅; prune = FALSE
      While( T_open ≠ ∅  AND prune = FALSE) do
      begin
          (t_i, s_j) = Select_Node()
          TTABLE = TTABLE + Set_Timetable(t_i, s_j)
          δ_est = Estimate_Cost(TTABLE)
          if(δ_est ≥ δ_best)
          then prune = TRUE
          elseif(Last_St(t_i, s_j) then T_open = T_open \ {t_i}
      end
      if(prune = FALSE) then
      begin
          δ = Evaluate_Timetabling(TTABLE)
          if δ < δ_best then TTABLE_best = TTABLE; δ_best = δ
      end
  end
  return TTABLE_best
end
```

Fig. 1. An outline of SOBM

3.1 Using a Search Tree to Model the Train Timetabling Problem

We consider the problem as a search tree whose root node represents the empty timetable (*initial* in Figure 2a). For each node where no successor is possible, there is an artificial terminal node (*final* in Figure 2a). Each intermediate node is composed by a pair (t_i, s_j), which indicates that a feasible timetable must be found for the train $t_i \in \mathrm{T_{open}}$ in its track section s_j. When the timetable of a train t_i is completed, i.e. the procedure `Last_Station(`t_i, s_j`)` is `TRUE` in Figure 1, this train is eliminated from the set $\mathrm{T_{open}}$. Each level of the search tree indicates which part of the timetable of each train may be generated (see Figure 2a). Given a node $n = (t_i, s_j)$ in a given level l; its brothers in l and the node (t_i, s_{j+1}) (if s_j is not the last track section of t_i) are the successors of the node n in the next level $l + 1$. The problem consists of finding a path in the search tree, from the *initial* node to a *final* node, so that the order of priorities established by this path produces the minimum average delay. Figure 1 shows



(a) An example of the Solving Method.

(b) A Solution Given by SOBM.

Fig. 2. A Model for the Problem and a Solution Obtained

the algorithm followed by the method to produce feasible solutions. If we take Figure 2a into account, the procedure `Select_Node()` decides which node of each level will form part of the final path. Each iteration of the inner loop in Figure 1 corresponds to one level in the search tree of Figure 2a. The heuristic that is used by the procedure `Select_Node()` is given in the next subsection.

Each time a new node (t_i, s_j) is chosen, the procedure `Set_Timetable(`t_i`, `s_j`)` assigns a feasible timetable to train t_i in the track section s_j. Then, this pair (t_i, s_j) with its corresponding timetable is added to the set *TTABLE*. Once time *TTABLE* has been updated with the new pair, an estimated cost is computed (δ_{est}) for the current path. If δ_{est} is greater or equal to the best solution found at the time (δ_{best} in Figure 1), then the current path is pruned (*prune*=`TRUE` in Figure 1), and a new path is started from the *initial* node. Given that a feasible timetable was assigned to t from its initial station l_0^t until the station $l_{i_t}^t$, and $M_{i \to n_t}^t$ is the minimum running time possible for the train t to travel from $l_{i_t}^t$ until $l_{n_t}^t$, the procedure `Estimate_Cost(`*TTABLE*`)` in Figure 1 computes the value for δ_{est} according to the following expression:

$$\delta_{\text{est}} = \frac{\sum_{t \in \mathrm{T_{new}}} \delta_{\text{est}}^t}{|\mathrm{T_{new}}|} \text{ where } \delta_{\text{est}}^t = \frac{arr_{i_t}^t - dep_0^t + M_{i_t \to n_t}^t - \Gamma_{\text{opt}}^t}{\Gamma_{\text{opt}}^t}$$

3.2 Heuristic to Choose a Node for Each Level of the Tree

In this subsection, we detail how the method determines the node that must be chosen at each level of the tree. Given that a feasible timetable was assigned to t from its initial station l_0^t until the station $l_{i_t}^t$, for each level, we measure the partial delay of each train $t \in \mathrm{T_{open}}$ according to the following expression:

$$\delta_{\mathrm{partial}}^t = \frac{arr_{i_t}^t - dep_0^t - \Gamma_{\mathrm{opt}}^t}{\Gamma_{\mathrm{opt}}^t} \quad . \tag{16}$$

Given that the minimum partial delay is $\delta_{\min} = \min_{t \in \mathrm{T_{open}}}(\delta_{\mathrm{partial}}^t)$, the probability ρ_t, of the train t of being selected, is computed according to the following expression:

$$\rho_t = \frac{(\delta_{\mathrm{partial}}^t - \delta_{\min} + \varepsilon)^\alpha}{\sum_{t \in \mathrm{T_{open}}}(\delta_{\mathrm{partial}}^t - \delta_{\min} + \varepsilon)^\alpha} \quad . \tag{17}$$

A train is chosen according to the parameterized Regret-Based Biased Random Sampling (RBRS) [5] and [8] in such a way that the train with higher priority is not necessarily the train chosen, due to the random component of the RBRS method.

4 Results

We describe an example problem that was solved using the SOBM approach.We use the railway line between Madrid and Jaen which covers 54 locations (stations, halts, siding, etc); it is 369,4 kilometers long and has 30/23 two/one-way track sections. In this example, the number of trains in circulation, between 12:00 and 24:00 hours, before adding the new trains is 103. We have to add 59 new trains to the line in the hourly interval [12:00, 24:00], satisfy the constraints given in 2.2 taking into account the trains already in circulation, and minimize the average delay of the new trains. The planner usually works with the help of only a graphical tool (one train after another). The solution obtained is feasible but not necessarily a good solution due to the combinations that must be considered simultaneously. In the example considered, the solution obtained by the SOBM approach has a delay of 7,2% with respect to Γ_{opt}. Figure 2b shows the timetable obtained for each train in a graphical way.

The Spanish Manager of Railway Infrastructure (ADIF) provides us with real instances to obtain a realistic evaluation of the proposed heuristic. We describe ten problem instances in Table 1 (columns 2 to 9) by means of: the length of the railway line, number of single/double track sections, number of stations, number of trains and track sections (T-ts) corresponding to all these trains, for trains already in circulation (Traffic) and for new trains (Tnew), respectively.

The results are shown in Table 2. This Table presents the best value of the objective function and the number of feasible solutions that were obtained for each problem (columns 2 and 3 for the RANDOM approach; columns 4 and 5 for the SOBM approach). The algorithm is implemented using C++ running on

Table 1. Real railway problem instances provided by ADIF

Problems	Infrastructure Description				Traffic		Tnew	
	Km	1-Way	2-Way	Stat	Trains	T-ts	Trains	T-ts
1	209,1	25	11	22	40	472	53	543
2	129,4	21	0	15	27	302	30	296
3	177,8	37	4	25	11	103	11	146
4	225,8	33	0	23	113	1083	11	152
5	256,1	38	0	28	80	1049	15	235
6	256,1	38	0	28	81	1169	16	159
7	96,7	16	0	13	47	1397	16	180
8	96,7	16	0	13	22	661	40	462
9	298,2	46	0	24	26	330	11	173
10	401,4	37	1	24	0	0	35	499

Table 2. Results of the SOBM and RANDOM approach

Problems	RANDOM		SOBM	
	# of Solutions	ADOS%*	# of Solutions	ADOS%*
1	169	8,6	168	5,9
2	611	10,1	608	10,0
3	2185	21,1	3101	16,0
4	311	13,2	445	5,5
5	396	19,3	452	17,5
6	424	14,7	521	14,1
7	267	18	263	15,4
8	67	50,9	85	45,5
9	1112	11,5	1129	8,7
10	405	19,2	397	17,9
Average	**594,7**	**18,6**	**716,9**	**15,6**

a Pentium IV 3,6 Ghz. The running time was of 300" for all the problems, and the parameters of the RBRS set to α=1 and ε=0,5.

The difference between the RANDOM approach and the SOBM approach is the way that the trains are chosen at each level of the search tree. In the first approach the trains are chosen randomly, in the second approach, the trains are chosen according to the method explained in Section 3. The results show that a guided heuristic such as SOBM explores regions in a more promising way in less time than the RANDOM approach does and this leads to better solutions.

5 Conclusions

In this paper, we present a Scheduling Order-Based Method (SOBM) that proposes a model for the Train Timetabling Problem and a procedure to solve it.

We present a procedure and a heuristic to decide between one or another alternative when a disjunctive constraint must be solved. This decision is based on the knowledge about the problem under consideration and a given objective function. Several realistic instances of the problem have been verified as well as different traffic conditions and train configurations. The method can be applied to any railway line and does not require a specific configuration in the railway infrastructure. The set of constraints can be modified without affecting the solving process used during the optimization. SOBM is being used currently by ADIF .

Acknowledgments

This work has been supported by the Research projects TIN2004-06354-C02-01 (MEC, Spain - FEDER) and GV04B/516 (Generalitat Valenciana, Spain). We are grateful to the Spanish Manager of Railway Infrastructure (ADIF) and to Jose Estrada Guijarro for give us the opportunity to interact with a real environment, for provide us with the instances presented in this paper, and for his effort and time to define the problem and evaluate the resulting schedules.

References

1. Cordeau, J., Toth, P. and Vigo, D. A survey of optimization models for train routing and scheduling. Transportation Science 32, 380-446 (1998)
2. Lova, A., Tormos, P., Barber, F., Ingolotti, L., Salido, M.A. and Abril, M. Intelligent Train Scheduling on a High-Loaded Railway Network. ATMOS 2004: Algorithmic Methods and Models for Optimization of Railways. Lecture Notes in Computer Science, LNCS (Springer Verlag). To appear. 2005
3. Nachtigall, K. and Voget, S. Minimizing waiting times in integrated fixed interval timetables by upgrading railway tracks 103, 610–627, (1997)
4. Schrijver, A., and Steenbeek, A. Timetable construction for railned. Technical Report, CWI, Amsterdam, The Netherlands (in Dutch), (1994)
5. Schirmer, A. and Riesenberg, S. Parameterized heuristics for project scheduling-Biased random sampling methods. Technical Report 456. Institute fr Betriebswirtschaftslehre der UNIVERSITT KIEL (1997)
6. Serafini, P. Ukovich, W. A mathematical model for periodic scheduling problems. SIAM Journal on Discrete Mathematics 2(4), 550–581, (1989)
7. Silva de Oliveira, E. Solving Single-Track Railway Scheduling Problem Using Constraint Programming. Phd Thesis. Univ. of Leeds, School of Computing (2001)
8. Tormos, P. and Lova, A. A Competitive Heuristic Solution Technique For Resource-Constrained Project Scheduling. Annals Of Operations Research 102, 65–81, (2001)
9. Walker, C., Snowdon, J. and Ryan, D. Simultaneous disruption recovery of a train timetable and crew roster in real time. Comput. Oper. Res. 32 (8), 2077–2094, (2005)

A Topological-Based Method for Allocating Sensors by Using CSP Techniques

R. Ceballos, V. Cejudo, R. M. Gasca, and C. Del Valle

Departamento de Lenguajes y Sistemas Informáticos,
Universidad de Sevilla (Spain)
{ceballos, cejudo, gasca, carmelo}@lsi.us.es

Abstract. Model-based diagnosis enables isolation of faults of a system. The diagnosis process uses a set of sensors (observations) and a model of the system in order to explain a wrong behaviour. In this work, a new approach is proposed with the aim of improving the computational complexity for isolating faults in a system. The key idea is the addition of a set of new sensors which allows the improvement of the diagnosability of the system. The methodology is based on constraint programming and a greedy method for improving the computational complexity of the CSP resolution. Our approach maintains the requirements of the user (detectability, diagnosability,...).

1 Introduction

Model-based diagnosis (MBD)[1][2] allows to determine why a correctly designed system does not work as expected. In MBD, the behaviour of components is simulated by using constraints. Inputs and outputs of components are represented as variables of constraints. These variables can be observable and non-observable depending on the sensors allocation. The objective of the diagnosis process is to detect and identify the reason for any unexpected behaviour, and to isolate the parts which fail in a system.

The diagnosability of systems is a very active research area in the diagnosis community. A toolbox integrating model-based diagnosability analysis and automated generation of diagnostics is proposed in [3]. The proposed toolbox supports the automated selection of sensors based on the analysis of detectability and discriminability of faults. In this line, a methodology to obtain the diagnosability analysis using the analytical redundancy relations (ARR) was proposed in [4]. This approach is based on an exhaustive analysis of the structural information. The objective is the addition of new sensors to increase the diagnosability. In a previous work [5] a methodology to analyze the diagnosability of a system based in a process algebras was proposed. A framework for testing the diagnosability of a system is defined by using the available sensors, a model abstraction, and some snapshots of sensor readings.

In this work, a new approach is proposed in order to improve the computational complexity for isolating faults in a system. Our approach is based on the addition of new sensors. A constraint satisfaction problem (CSP) is obtained in

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 62–68, 2006.

order to select the necessary sensors to guarantee the problem specification. We propose an algorithm for determining the bottleneck sensors of the system in order to improve the computational complexity of the CSP. A CSP is a framework for modeling and solving real problems as a set of constraints among variables. A CSP is defined by a set of variables $X=\{X_1, X_2,..., X_n\}$ associated with a set of discrete-valued, $\mathcal{D}=\{\mathcal{D}_1, \mathcal{D}_2,..., \mathcal{D}_n\}$ (where every element of $\mathcal{D}_i$ is represented by set of v_i), and a set of constraints $C=\{C_1, C_2,..., C_m\}$. Each constraint C_i is a pair (W_i, R_i), where R_i is a relation $R_i \subseteq D_{i1}\cdot...\cdot D_{ik}$ defined in a subset of variables $W_i \subseteq X$.

The remainder of the paper is organized as follows. Section 2 provides the definitions and notation in order to clarify MBD concepts. Section 3 introduces the basis of our approach. Section 4 describes the CSP generation. Sections 5 shows the greedy method for improving the CSP resolution. Finally, conclusions are drawn and future work is outlined.

2 Notation and Definitions

In order to explain our methodology, it is necessary to establish some concepts and definitions from the model-based diagnosis theories.

Definition 1. A *System Model* is a finite set of equality constraints which determine the system behaviour. This is done by means of the relations between the non-observable and observable variables (sensors) of the system.

Definition 2. A *Diagnosis* is a particular hypothesis that shows the system differs from its model. Any component could be working or faulty, thus the diagnosis space for the system initially consists of 2^{nComp} - 1 diagnoses [2], where *nComp* is the number of components of the system. The goal of diagnosis is to identify and refine the set of diagnoses.

Definition 3. The *Discriminability Analysis* [3] determines whether and under which circumstances the considered (classes of) faults can be distinguished.

Definition 4. The *Diagnosability level* is the quotient of the number of the (classes of) faults which can be distinguished each other, and the number of all the possible faults. The size of the possible faults is initially 2^{comp} - 1.

Definition 5. A set of components T is a *Cluster of components* [6], (i) if it does not exist a common non-observable variable of any component of the cluster with any component outside the cluster, and (ii) if for all $Q \subset T$ then Q is not a cluster of components.

All common non-observable variables between components of the same cluster belong to the cluster, therefore, all the connections with components which are outside the cluster are monitored. A cluster of components is completely monitored, and for this reason the detection of faults inside the cluster is possible without any

information from other components which do not belong to the cluster. A more detailed explanation and the cluster detection algorithm appears in [6].

3 The Basis of the Algorithm

Our approach is based on the generation of new clusters of components by allocating sensors in some of the non-observable variables. These new clusters reduce the computational complexity of the diagnosis process since it enables the generation of the diagnosis of the whole system based on the diagnosis of the subsystems. Let C be a set of n components of a system, and C_1 and C_2 be clusters of n - m and m components such as $C_1 \cup C_2 = C$; then the computational complexity for detecting conflicts in C_1 and C_2 separately is lower than in the whole system C, since the number of possible diagnoses of the two clusters is $(2^{n\text{-}m}) + (2^m)$ - $2 \leq 2^{n\text{-}m} \cdot 2^m$ - 2 which is less than 2^n-1.

The clustering process enables isolating the faults of the original system, since the multiple faults which include components of different clusters are eliminated. These kind of faults are transformed into single or multiple faults which belong to only one cluster. The computational complexity for detecting conflicts and discriminating faults in a system is always higher than for an equivalent system divided into clusters.

4 The CSP Problem Specification

The objective is to obtain the best allocation of sensors in order to generate new clusters. The allocation of the sensors will be formulated as a Constraint Satisfaction Problem (CSP). A CSP is a way of modeling and solving real problems as a set of constraints among variables.

The methodology was applied to the 74181 4-Bit ALU. It is one of the ISCAS-85 benchmarks [7]. It includes 61 components, 14 inputs and 8 outputs. Table 1 shows the set of variables and constraints for determining the number and location of sensors for this example. The following variables are included:

– *nNonObsVar*: This constant-variable holds the number of non-observable variables.

Table 1. CSP for the 74181 ALU sensors allocation

Variable (= initial value)	**Domain**
(1) nSensors = {free}	$\mathcal{D}$={1, . . . , nNonObsVar}
(2) clusterOfComp$_i$ = {free}	$\mathcal{D}$={1, . . . , nComp}
(3) clusterDist$_t$ = {free}	$\mathcal{D}$={1, . . . , nComp}
(4) sensor$_k$ = {free}	$\mathcal{D}$={true, false}
Constraints	
(5) if (sensor$_{E01}$ = false) ⇒ clusterOfComp$_{M11}$ = clusterOfComp$_{M32}$	
(6) if (sensor$_{E02}$ = false) ⇒ clusterOfComp$_{M19}$ = clusterOfComp$_{M32}$	
...	

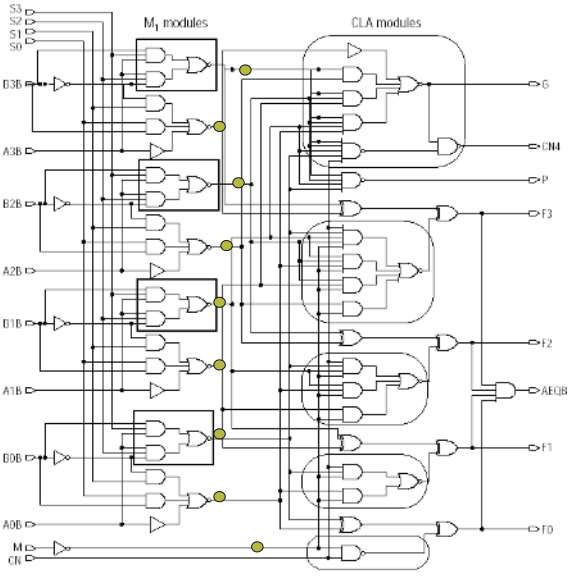


Fig. 1. 74181 ALU

- *nSensors*: This variable holds the number of new sensors. It must be smaller than the number of non-observable variables.
- $sensor_k$: This set of variables represents the possible new sensors of the system. They hold a boolean value in the interval {true, false}, where *true* implies that there must be a sensor, and *false* the opposite.
- $clusterOfComp_i$: This set of variables represents the cluster associated to each component i.
- $clusterDist_t$: This set of variables holds the number of components included in each cluster t.

For each common non-observable variable between two components a constraint is generated which guaranties that if there is not a sensor, the two components must belong to the same cluster. Table 1 shows the constraints (5),(6),... that hold this kind of information, and it is based on Figure 1. The final sensors allocation is stored in $sensor_k$, and the distribution of the clusters is stored in $clusterDist_i$. The optimization problem can have different objectives, depending on the user and the problem requirements. Two typical goals can be:

- To minimize the number of sensors (if the number of clusters is fixed).
- To minimize the maximal number of components in each cluster (if the maximal number of sensors is fixed).

It is possible to add other constraints in order to guarantee some properties of the solution. For example, in order to guarantee prices, to respect requirements of the customers, to store incompatibilities, to specify problems, ... We

have applied the limited discrepancy search (LDS) [8] algorithm in order to search the solution. This algorithm is based on the limitation of the number of discrepancies.

5 Improving the Algorithm: A Greedy Method

The computational complexity of a CSP is exponential in general. We propose a method to obtain the most important allocation of the new sensors in order to generate more clusters; that is, the bottlenecks of the system. Our method has two phases:

1. The calculation of the minimal paths: A graph where the nodes represent the components of the system, and the edges represent the connections between each two components (non-observable variables). Each edge has a weight calculated as the number of common non-observable variables between two components. By applying the Floyd's algorithm (dynamic programming), all the shortest paths between all pairs of nodes is stored.
2. In order to determine which are bottlenecks of the system, each minimal path will vote which sensors are more important. Figure 2 shows this algorithm.

```
sensorsOrder(P)
  componentVotes[nComp][nNonObsVar]
  sensorVotes[nNonObsVar]
  // All the components (1..nComp) votes the variables (sensors)
  // associated to the minimal paths
  forEach j between 1 to nComp
    forEach P_k from component i to component j
      forEach q between 1 to length(P_k)
        △ = (voteValue / (length(P_{k,i,j}) · length(P_k))
        forEach v includes in path[q]
          componentVotes[i][P_{k,i,j,q}] += △
        endForEach
      endForEach
    endForEach
  endForEach
  // Recounting of votes for each sensor
  forEach sensor_j between 1 to nSensors
    sensorVotes[j] = 0
    forEach i between 1 to nComp
      sensorVotes[j] += componentVotes[i][j] / ( △ · nComp )
    endForEach
  endForEach
  return sort(sensorVotes)
```

Fig. 2. Algorithm for obtaining the bottleneck sensors of the system ($O(n^2 \cdot m^2)$, where n is the number of components and m is the number of non observable variables)

Each minimal path will vote for the included non-observable variables of the minimal path. The number of votes are scaled in order to guarantee that each component generates the same total number of votes. These votes allow to generate a sorted list of non-observable variables. This list is composed of the most relevant sensors with the aim of generating new clusters.

The bottlenecks of the system represent the best sensors in order to isolate components and faults. The sorted list of sensors enables creating a CSP with less variables to find the solution of the problem in a limited time. Only the solutions included in the combinations of the m bottleneck sensors will be tested, and therefore, the number of possible solutions will be lower than 2^m. The optimal solution is not guaranteed, but the reduction of computational complexity enables finding a solution in a limited time.

Example: In the *Alu74181* example the most important sensors are (based on the number of votes): $E_{02}(930)$, $E_{03}(878)$, $X_{28}(773)$, $E_{01}(737)$, $E_{00}(583)$, $D_{00}(514)$, $D_{01}(463)$, $D_{02}(326)$, $D_{03}(301)$,... The other sensors have less than 166 votes. The first 9 sensors are represented by shaded circles in Figure 1. The possible diagnoses in the system are 2^{61} - 1. By using the first 9 selected sensors, the number of clusters is 17 (all with less than 6 components) and the computational complexity is reduced because of the reduction of possible diagnosis to less than 2^9.

6 Conclusions and Future Work

The objective of our approach is the allocation of a set of new sensors in order to improve the computational complexity and diagnosability of a system. The methodology was applied to an standard example, and the results are very promising. It is based only on topological properties. This enables applying this approach to different kinds of systems. As a future work, we are working on new greedy methods to improve the votes counting.

Acknowledgements

This work has been funded by the Spanish Ministry of Science and Technology (DPI2003-07146-C02-01) and the European Regional Development Fund.

References

1. Reiter, R.: A theory of diagnosis from first principles. Artificial Intelligence 32 **1** (1987) 57–96
2. de Kleer, J., Mackworth, A., Reiter, R.: Characterizing diagnoses and systems. Artificial Intelligence **2-3**(56) (1992) 197–222
3. Dressler, O., Struss, P.: A toolbox integrating model-based diagnosability analysis and automated generation of diagnostics. In: DX03, 14th International Workshop on Principles of Diagnosis, Washington, D.C., USA (2003) 99–104

4. Travé-Massuyés, L., Escobet, T., Spanache, S.: Diagnosability analysis based on component supported analytical redundancy relations. In: 5th IFAC Symposium on Fault Detection, EEUU (2003)
5. Console, L., Picardi, C., Ribaudo, M.: Diagnosis and diagnosability analysis using PEPA. In: ECAI 2000, Proceedings of the 14th European Conference on Artificial Intelligence, Berlin, Germany, IOS Press (2000) 20–25
6. Ceballos, R., Gómez-López, M.T., Gasca, R., Pozo, S.: Determination of Possible Minimal Conflict Sets using Components Clusters and Grobner Bases. In: DX04, 15th International Workshop on Principles of Diagnosis, Carcassonne, France (2004) 21–26
7. Hansen, M.C., Yalcin, H., Hayes, J.P.: Unveiling the ISCAS-85 Benchmarks: A Case Study in Reverse Engineering. IEEE Design and Test of Computers **16**(3) (1999) 72–80
8. Harvey, W.D., Ginsberg, M.L.: Limited discrepancy search. In: Fourteenth IJCAI, Montreal, Canada, IOS Press (1995)

Adapting the Point of View for Behavior-Based Navigation

M. Ardaiz, A. Astigarraga, E. Lazkano, B. Sierra, J. M. Martínez-Otzeta, and E. Jauregi

Department of Computer Science and Artificial Intelligence,
University of the Basque Country,
Manuel Lardizabal 1, 20018 Donostia
http://www.ehu.es/ccwrobot

Abstract. Perception triggered response behavior together with the selection of the appropriate environmental cues can properly guide the robot towards its goal. But often landmarks are difficult to be recognised by the robot; these difficulties are increased due to the morphology of the robot and its movements. Dynamic orientation of the camera's parameters, fixing the attention of the visual system on relevant environmental features, can greatly improve vision based landmark identification. This paper presents an approach to sonar and compass based dynamic selection of pan, tilt and zoom values.

Keywords: Behaviour-based robotics, visual landmarks, purposive vision, recognition-triggered response.

1 Introduction

Recognition-triggered response is a biological navigation strategy [22] [13] that connects different locations by means of local navigational algorithms. Sequencing appropriately different recognition-triggered responses, the robot can show the ability to follow a predefined sequence of landmarks drawing a concrete route. The recognition of a certain location triggers the activation of the proper local navigation method, guiding the robot towards the goal. In this context, landmarks in the environment can be used as cues by the robot to identify different locations, either to self-localise in the environment or to trigger different behaviours. Although landmarks can be different in nature [18][20][14] [5], vision systems provide a rich source of information for landmark identification, a priori less sensitive to aliasing. If the robot is provided with a vision system, then the problem is to determine what scene information should be used to best implement the task the robot must accomplish.

Emergency exit panels are natural landmarks in office-like indoor environments. Although human specified landmarks, emergency exit panels are international natural landmarks mandatory in every public building. According to the European "Council Directive 92/58/EEC of 24 June 1992 on the minimum requirements for the provision of safety and/or health signs at work" (Official Journal L 245 , 26/08/1992 P. 0023 - 0042), emergency exit panels that must follow some shape, colour and location standards. But

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 69–78, 2006.

visual processes are directly related with the task to be performed by the observer. Animate or purposive vision mandates that perception is linked to action and that vision must be based on expectations [4][1]. The knowledge about the environment may somehow condition the interpretation of what is being seen. Both, active vision and, more recently, zooming capabilities are used in the field of mobile robotics mainly to focus the attention of the visual system on some feature of the environment [10][21] [16].

Our current effort is aimed at improving emergency exit panel identification behaviour with a dynamic orientation of the camera and zooming capabilities, thus improving the navigational skills of the robot once integrated into the overall navigational architecture. In [3] a mechanism for tilt and zoom parameter adjustment was presented that fitted the panel identification subsystems needs. This mechanism is now extended to modify also the camera's horizontal angle using the magnetic orientation as reference, in order to allow the robot to properly position the view to look for the panels. Therefore, in this paper we present the design and implementation of an active camera positioning behaviour that exploits robot's sensors capabilities to change its pan, tilt and zoom parameters to reliably identify emergency exit panels from the scene in its goal-based navigation task, and describe how it is integrated in the overall behaviour-based [6] [15] control architecture.

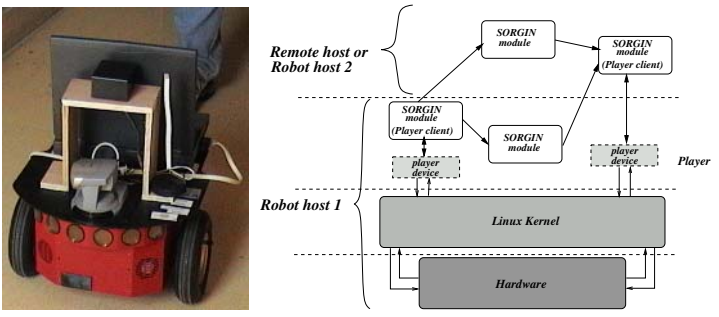
The rest of the paper is organised as follows: the next section reviews the system, i.e the environment, the robot and software used, and the basic modules or single behaviours that compose the control architecture for the navigation task. Section 3 is devoted to the dynamic adjustment of the camera parameters, reviewing the panel identification system, briefly describing how tilt and zoom are adapted and emphasising on the compass based pan angle selection. The experiments performed to evaluate the behaviour of the new system and the results obtained are presented in section 4. Finally, section 5 includes conclusions and suggests further research.

2 Robot-Environment System

The robot platform is a small Pioneer 3 model from Activmedia Robots named Galtxagorri (see figure 1(a)). The Cannon VCC5 vision system mounted on top of it possesses pan, tilt and zoom capabilities. Therefore, the camera's parameters can be adapted at run time to adjust the point of view according to the task in hand. The robot is also provided with a front ring of eight sonar sensors and a TCM2 compass device mounted on a home-made wooden platform to avoid magnetic disturbances from the motors.

The environment the robot moves in is a typical office-like semistructured environment, with rectangular halls connected by corridors, full of geometrical properties that can be exploited.

The Player-Stage [8] client/server library allows us to communicate with the different devices connected to the robot, all but the compass that it is plugged to a serial port and accessed through an independent function set. This software, appropriately combined with SORGIN, a software framework specifically developed to define and implement behaviour-based control architectures [2] provides us with the adequate programming environment for robot control. Figure 1(b) shows how those two tools reside on the robot.



(a) Galtxagorri (b) Software architecture

Fig. 1. Robotic platform

The behaviour-based control architecture consists of the following modules:

1. Behaviours necessary for wandering around. In terms of biologically inspired navigation, these behaviours produce a search strategy in a fixed direction by means of two modules: left and right free space balancing, and magnetic orientation following. Each module outputs translational and rotational velocity values and the final output motor command is the weighted sum of both modules. The behaviour produced by the combination of these two modules is more than a simple wandering because, at the same time as it avoids collisions, it directs the robot to the goal defined by the orientation. The desired compass orientation acts as an attractor when there is enough free space for the robot to move safely, thus adding persistence to the global behaviour.
2. Behaviours for triggering orientation changes. These landmark identification behaviours do not produce motor responses, but send and receive signals and data. Again, two processes are identified: corridor identification and emergency exit panel identification modules.

Figure 2 shows these basic modules and the communication among them. The module filled in grey represents the behaviour for actively controlling camera head, which is the main topic of the work described here and which is needed to robustly perceive the visual landmarks and trigger the corresponding action.

3 Dynamic Pan, Tilt and Zoom Adjustments for Visual Landmark Detection

One of the main problems the robot faces when recognising visual landmarks, is the big sensitivity to the distance they are located at. The robot can miss the panel when it is out of its nominal trajectory. If the robot is too close or too far from the panel, or if the camera is not appropriately oriented to see it, the panel gets out of the camera's view or it is viewed too small for the robot to reliably identify it. Obviously, the dynamic adaptation of the camera's pan, tilt and zoom values should improve the recognition

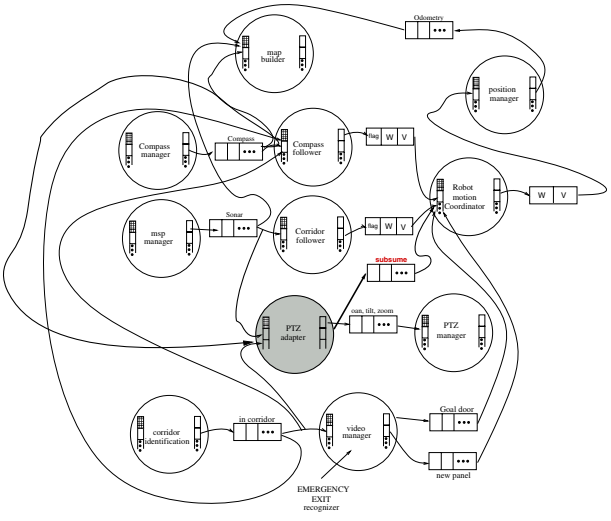


Fig. 2. Basic modules of the control architecture

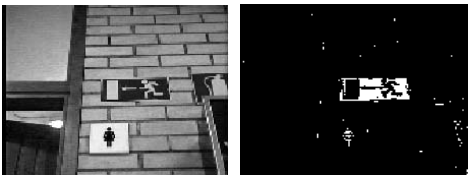
process. Looking at the inner working of active vision systems, we found that most of them look for geometrical relations between the location of features with respect to robot's position [7] [16]. Instead of imposing geometrical constraints to the robot's perception system we claim that it is better to adapt those geometrical relations to fit to robot's perception of the environment. The panel identification subsystem is composed of a multi layer perceptron (MLP). Because it is the MLP itself which must identify the panels, we rely on the MLP identification process and let it indicate us the appropriate values for each parameter. In other words, we accommodate the pan, tilt and zoom values according to the MLP needs.

3.1 Emergency Exit Panel Recognition

To extract the panel from the background in the image, we first apply the colour enhancing algorithm (offered by The Gimp GNU image processing tool [11]) to the image in order to obtain a better segmentation. After the image has been enhanced a simple thresholding is enough to appropriately segment the green areas of the image (see figure 3).

To classify a binary image as containing or not a panel, we use a multi layer perceptron (MLP) [17] trained using as input the quadratic weighted sums of 20×20 sized image blocks [12]. These training images were taken from within robot's nominal trajectory and using pan angle perpendicular to robot motion and fixed tilt and zoom values. The MLP is improved by looking for appropriate input weights using a *Genetic algorithm* (GA) [9] and associating the LOO performance of the MLP itself to the individuals as fitness function, raising the classification accuracy of the net up to 97.5%.

In order to add robustness to the panel recognition process, we make use of a confidence level (cl) proportional to the number of consecutive images positively identified



(a) (b)

Fig. 3. Original and segmented images

that affects the global translational velocity according to $v' = v(1 - cl)$. The aim of this velocity reduction is to slow down the robot when a panel is being recognised so that the robot does not miss it.

This processing allows the robot to detect the visual landmarks within the nominal trajectory; however, the robot misses the panel if it is too close of too far from the wall due to the lack of capabilities to direct the viewpoint to the correct area.

Three parameters must be adapted:

- Elevation or tilt: it is dependent on the distance to the wall.
- Zoom: it determines the panel size in the image and it is again dependent on the distance.
- Camera's direction or pan: opposite to the previous parameters, it cannot be adjusted according to the distance. Robot heading angle with respect to the wall must be known.

3.2 Sonar Based Tilt and Zoom Adjust

To establish the relationship between the camera's tilt angle and the distance to the wall, measured by means of the robot's sonar sensors – more specifically the two sensors perpendicular to the wall– we just placed the robot at different positions and considered a valid tilt angle if it was a zoom range for which the MLP was able to robustly identify the panel. Although the collected data did not show exactly a linear relationship among the parameters, we approximated the curve using a linear function with bounded minimum value.

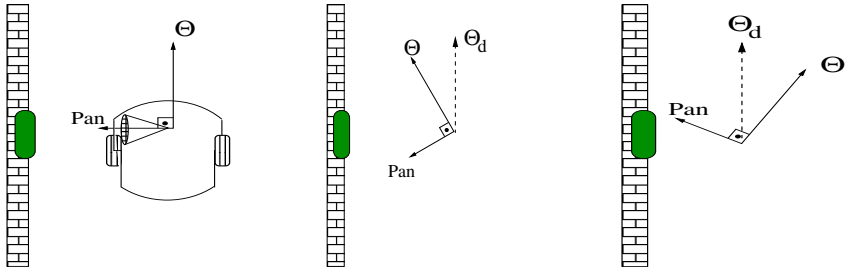
On the other hand, emergency exit panels are small in size and the robot can miss them even if the camera is adequately focused to the area where the panel is expected to be. Therefore, to reliably identify emergency panels from the scene, the MLP system needs exit panels of certain size. When the robot is too far from the wall, the camera's view of the exit signal is too small to be properly identified by the MLP system; and if the robot gets very close to the wall, the emergency image takes up the whole screen and the MLP system fails again. The right size of the panel can be obtained by dynamically adapting the camera's zoom parameter according to the MLP needs.

With the dynamic tilt adjustment behaviour running on the robot, we collected data for the zoom values in a similar manner. Letting the robot select the tilt angle and changing the zoom value using uniform intervals (ranging from 0 to 1300, 20 units rate), we captured the MLP results for each zoom value and the sonar measured distance to the

wall. Using as centroid the mean value of the acceptable zoom range for each measured distance, we interpolated the curve using a B-spline [19]. The reader is refereed to [3] for more details.

3.3 Compass Based Pan Adjustment

The local navigation strategies guide the robot through a trajectory parallel to walls, but non-symmetric environments and dynamic obstacles make the robot vary the orientation (θ) with respect to the reference (θ_d). Figure 4 reflects the effect of this displacement in the pan angle. The panel identification subsystem suffers from the view angle misplacement and so does the global robot behaviour.



(a) Correct view (b) Incorrect (c) Incorrect

Fig. 4. Relation between pan angle and robot orientation

In order to adequate the pan angle, the angle of the robot heading with respect to the wall must somehow be known. Sonar sensors could be used to approximate the wall angle with respect to the reference trajectory, for example by means of linear regression. However, sonar readings heavily depend on the reflection properties of surfaces and are sensible to specular reflections and lost echos. Noisy sonar measurements would produce unreliable angle approximations and, therefore, failures in the pan adjustments. Instead, taking into account that the reference heading θ_d is known for every panel that needs to be identified, we chose to measure the displacement with respect to the reference heading using the electronic compass reading. The displacement is then the difference between the given reference θ_d and the magnetic orientation θ given by the compass: $\theta - \theta_d$.

Three different cases are distinguished as shown in figure 4. In figure 4(a) the difference is within a range in which the (previously used) fixed pan angle is valid and no correction is needed. In order to measure this range, we made some experiments in which the robot turned on the spot 360° and the output of the panel identification subsystem was registered, while the zoom and tilt adapting mechanism was running. Figure 5 shows the nature of the data collected to identify this range.

Out of this range, when the robot is oriented towards the wall as in figure 4(b), the pan angle can be adapted according to the difference among the desired (θ_d) and the

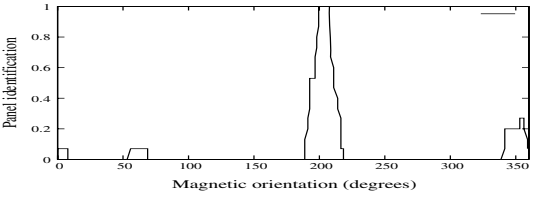


Fig. 5. Panel orientation: EW. Range: 205±10

actual compass orientation (θ) as shown in equation 1 (the $\pm$ sign depends on the side the camera must point to).

$$pan = \pm 90 \pm (\theta - \theta_d) \tag{1}$$

Finally, the pan axe limits cannot be exceeded physically. Valid pan angle values are within $(-98°, \cdots, 98°)$ range. If the robot is moving away from the wall as in figure 4(c), it is not possible to vary the pan angle to correct the camera's viewpoint. Nevertheless, it is possible to re-direct the robot to a more comfortable position. The modularity of the developed control architecture facilitates the addition of control signals. It is enough for the PTZ adjusting module to send a subsume control signal to the motor fusion module that will let the compass following module act alone on the rotation velocity. In this manner, the robot corrects its heading when the panel is being recognised and the pan can be adjusted within the allowable range.

The basic idea is then to use the robot heading direction to position the camera. It was mentioned before that sonar sensors were used to measure the distance to the wall when the robot is parallel to it in order to calculate the appropriate tilt and zoom values. If the robot is not aligned with the wall, then the distance measurement may be incorrect, due to the deviation from the reference angle. To cope with this problem we make use of the pan angle selected using the compass value and choose, at each time step, the sonar sensors closest to the perpendicular angle to the wall. This distance is the one used afterwards to adjust the zoom and the tilt. Figure 6(a) shows the sonars used when the robot is directed within the acceptable orientation range and figure 6(b) shows the ones used in the extreme case in which robot heading points directly to the wall.

Pseudo-code in figure 7 summarises the dynamic PTZ adjustment behaviour.

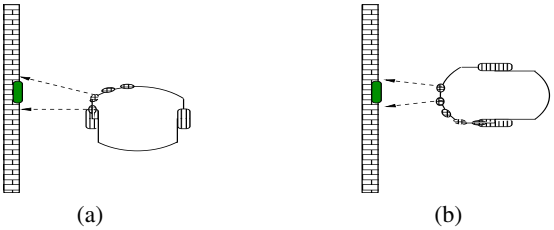


Fig. 6. Sonar selection for distance measurement

```
inputs: θ_d;
α_c = read_compass();
sonars = read_sonars();
pan = ±90°;
if θ ∉ θ_d ± range;
    pan = ±90° ± (θ − θ_d);
if pan ∉ (−90°, ··· , 90°);
    subsume_control();
else
{
    d = distance(sonars, PAN);
    t = dtilt(d, f_tilt);
    a = dzoom(d, Bspline);
}
set_ptz_values(pan, t, z);
```

(a) PTZ module

```
inputs: w_θ, v_θ, w_cf, v_cf, cl, s_pan
v = α_θ * v_θ + α_cf * v_cl
w = α_θ * w_θ + α_cf * w_cl
v = (1 − cl) * v;
if s_pan
     w = w_θ;
set_robot_velocity(v, w);
```

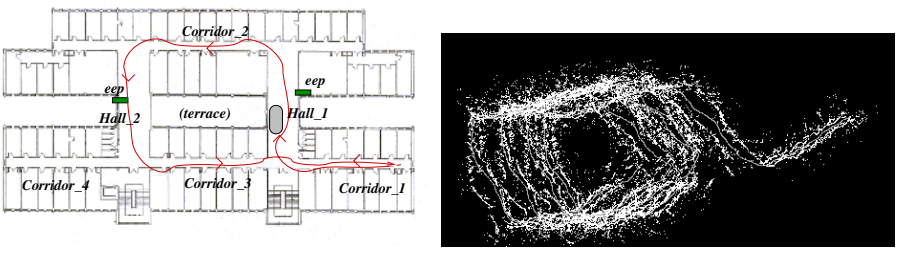
(b) Cooperative control module

Fig. 7. Pseudo-code for PTZ adjust

4 Experimental Results

In order to evaluate the performance and robustness of the new module different experiments were conducted. A set of static experiments was designed for verifying the correctness of the PTZ selection functions. Locating the robot at different distances from the wall (using floor tiles as references) and at 4 different orientations $(0°, 30°, 60°, 90°)$ forcing the robot at difficult positions with respect to the panel, we kept track of the panel recognition output. A success was considered if the output measured 1 (15 consecutive positive identifications), otherwise a fail was registered. Thereby, only the PTZ adjusting module and the panel identification module were run, and the robot was not driven by the local navigation strategies. For a series of 384 experiments, performed at two different environments we obtained an accuracy of $\%92.7$.

Besides, the new behaviour module was added to the global control architecture and a new set of experiments was defined. Locating again the robot at different initial positions, some obstacles were placed to force the robot to move out of its nominal trajectory, and to change its orientation to avoid the obstacle. Here, the robot was driven by the local navigation strategies and it was supposed to turn $90°$ when the panel was positively identified. For 30 runs in two different environments it again only missed the panel 5 times. It must be said that the experiments were performed with various illumination conditions that seriously affected the identification process. The new module is now running as part of the global control architecture and allows the robot to fullfill more complex trajectories. Figure 8(a) represents the environment in which the robot moves and the proposed circular trajectory. To complete the task the robot must go around the central area 4 times and in each round two emergency panels have to be identified triggering the correct change in the compass orientation. Figure 8(b) shows the map built on line, while the robot is performing the task.



(a) (b)

Fig. 8. Proposed trajectory and recorded path

5 Conclusions and Further Work

This paper present an improvement in the exit panel recognition system, showing how robot's perception of the environment by means of a MLP recognition system can be used to implement a sonar and compass based active camera head control. The addition of the compass based pan angle selection also helped to select the more adequate sonar sensors for measuring the distance and hence, improving the tilt and zoom adjusting mechanism. With such active process it is possible to detect emergency exit panels even if the robot is pushed out of its nominal trajectory. This emphasises the principle that vision does not have to function in isolation, but as part of a system that interacts with its environment in order to carry out its task.

It is mandatory to improve the panel recognition subsystem making it less sensible to various illumination conditions. An adaptive thresholding could be used over the enhanced image for this purpose or the images could also be filtered to remove undesirable variations. With a more robust identification, the number of consecutive images needed to positively recognise a panel could be relaxed, improving the overall performance specially when the robot shows snake-like movements due to obstacles.

Acknowledgements. This work was supported by the Gipuzkoako Foru Aldundia under grant OF838/2004 and by the MCYT under grant TSI2005-00390.

References

1. J. Aloimonos. *Active perception*. Lawrence Erlbaum Assoc Inc, 1993.
2. A. Astigarraga, E. Lazkano, I. Rañó, B. Sierra, and I. Zarautz. SORGIN: a software framework for behavior control implementation. In *CSCS14*, volume 1, pages 243–248, 2003.
3. A. Astigarraga, E. Lazkano, B. Sierra, and I. Ra. Active landmark perception. In *Proceedings of the 10th IEEE International Conference on Methods and Models in Automation and Robotics (MMAR)*, volume 2, pages 955–960, 2004.
4. D. H. Ballard. Animate vision. *Artificial Intelligence*, 48:57–86, 1991.
5. O. Bengtsson and A.J. Baerveldt. Robot localization based on scan-matching – estimating the covariance matrix for the IDC algorithm. *International Journal Robotics and Autonomous Systems*, 1–2(44):131–150, 2003.

6. R. A. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of robotics and automation*, RA–26:14–23, 1986.
7. A. J. Davison. *Mobile robot navigation using active vision*. PhD thesis, University of Oxford, 1999.
8. B. P. Gerkey, R. T. Vaughan, and A. Howard. The player/stage project: tools for multi-robot and distributed sensor systems. In *Proc. of the International Conference on Advanced Robotics (ICAR)*, pages 317–323, 2003.
9. D. E. Goldberg. *Genetic algorithms in search, optimization and machine learning*. Addison–Wesley, 1989.
10. E. Hayman, T. Thórhallsson, and D. Murray. Tracking while zooming using affine transfer and multifocal tensors. *International Journal of Computer Vision*, 51(1):37–62, 2003.
11. S. Kimball and P. Mattis. The GIMP: GNU image manipulation program, 1997. http://www.xcf.berkeley.edu/~gimp/gimp.html.
12. E. Lazkano, A. Astigarraga, B. Sierra, and I. Rañó. On the adequateness of emergency exit panel and corridor identification as pilot scheme for a mobile robot. In *Intelligent Autonomous Systems 8*, volume 1, pages 915–924, 2004.
13. H. A. Mallot and M. A. Franz. Biomimetic robot navigation. *Robotics and Autonomous System*, 30:133–153, 2000.
14. M. Mata, J.M. Armingol, A. de la Escalera, and M.A. Salichs. Learning visual landmarks for mobile robot navigation. In *Proceedings of the 15th World congress of the International Federation of Automatic Control*, 2002.
15. M. Matarić. Behavior–based control: examples from navigation, learning and group behavior. *Journal of Experimental and Theoretical Artificial Intelligence*, 9:323–336, 1997.
16. F. Michaud. Teaching a robot how to read symbols. In J. P. Müller, E. Andre, S. Sen, and C. Frasson, editors, *Proceedings of the Fifth International Conference on Autonomous Agents*, pages 184–185, Montreal, Canada, 2001. ACM Press.
17. T. Mitchell. *Machine learning*. McGraw-Hill, 1997.
18. U. Nehmzow and C. Owen. Experiments with manchester's fourty two in unmodified large environments. *Robotics and Autonomous Systems*, 33:223–242, 2000.
19. L. A. Piegl and W. Tiller. *The NURBS Book*. Springer Verlag, 1997.
20. N. Popescu. Robust self-localization of a robot by intelligent fuzzy system. In *14th Conference on Control Systems and Computer Science*, pages 175–179, 2003.
21. B. Tordoff. *Active control of zoom for computer vision*. PhD thesis, University of Oxford, 2002.
22. O. Trullier, S. I. Wiener, A. Berthoz, and J. A. Meyer. Biologically-based artificial navigation systems: review and prospects. *Progress in Neurobiology*, 51:483–544, 1997.

Agent Based Simulation for Social Systems: From Modeling to Implementation

Candelaria Sansores and Juan Pavón *

Universidad Complutense de Madrid, Dep. Sistemas Informáticos y Programación
28040 Madrid, Spain
csansores@fdi.ucm.es, jpavon@sip.ucm.es

Abstract. This paper presents experimental results of our model-driven approach to agent based simulation. According to this, the development process for agent based simulation should focus on modeling rather than implementation (i.e., programming on some concrete agent simulation platform). This requires the use of transformation tools from simulation models to implementation code. Describing social phenomena with a visual, high-level modeling language and implementing the simulation in this way should facilitate the use of toolkits by experts in social sciences without a deep knowledge of programming concerns. This simulation approach is supported here by the INGENIAS Development Kit (IDK), which provides a model editor for multi-agent systems, and code generation support. To validate the transformation mechanism, we have modeled a concrete social system with the INGENIAS agent-oriented modeling language, and generated two independent implementations for two different platforms (Repast and Mason simulation toolkits). This experimentation shows the feasibility of the model driven implementation approach and has enabled the study of facilities provided by simulation toolkits that can have impact on the transformation process, in particular, the scheduling techniques. Also, comparing the simulation results of the case study with the original work and between implementations has led us to discover biases introduced by simulation mechanisms that can be found in most simulation platforms.

1 Motivation

Computer simulation has an important role in the development of theories of non linear and dynamic systems, and its use is at the origin of the interdisciplinary scientific area of *complex systems* [8]. Complex systems are dynamical systems composed of a high number of components and multiple interactions among them. Also, [5] requires that the components be themselves complex, which is certainly true for the actor of social systems (like the individual), but in the case of some biology systems (like *swarm systems*) the actors use to be simple. Even social systems models usually contain relatively simple actors, and simplicity vs. complexity of actors is still

* This work has been developed with support of the Consejo Nacional de Ciencia y Tecnología (CONACYT) from México and the project TIC2002-04516-C03-03, funded by the Spanish Council for Science and Technology.

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 79 – 88, 2006.

an open debate (reviewed in [12]). Furthermore, [6] defines complex systems as complex adaptive systems that consist of networks of interacting agents and exhibit dynamic aggregate behaviors that emerge from individual activities. An aggregate behavior is emergent if it requires new categories to describe it, which are not used when describing the behavior of the underlying components.

Our work focuses on modeling and implementing non-linear complex systems in social sciences that exhibit these characteristics. As it is extensively discussed in [8], mathematical modeling has had little success for studying this kind of systems, mainly because expressing laws that regulate these phenomena is rather difficult if not impossible. An alternative is to make use of computer simulation where systems are modeled as societies of agents. Agents, in this context, are autonomous software systems that are intended to describe the behavior of observed social entities (e.g. individuals, organizations). An advantage of the agent-based modeling approach is the ability to estimate the plausibility of the behavior of agents, the way in which agents interact, and the consequences of that behavior and interaction [1]. Our aim is to build *artificial societies* [4], that is, multi-agent based models of social entities whose social structure and group behavior emerge from the interaction of individuals, under relative simple rules. Also, in this category we can include swarm systems, like many social insects groups, which show impressive collective problem-solving capabilities.

In this context, we have addressed several open issues found in conceptual modeling and implementation of social simulation systems. For conceptual modeling we provide an easily customizable tool for describing phenomena in different social domains with a visual language, which is based on the agent-oriented methods and tools of INGENIAS [9]. This agent-oriented visual language is implementation language-independent so it hides computational complexity to modelers. The technical description and an example of the use of this language is presented in [12]. The use of an agent-oriented modeling language facilitates the task of building conceptual models with respect to the direct use of simulation toolkits, where scientists do not build conceptual models but directly program their systems by using simulation support libraries. This is a problem for users or modelers without a strong programming background. Therefore, we propose to use an agent-oriented high-level language (in our case, INGENIAS), and for implementation issues we provide code generation tools that support the transformation of conceptual models into code for different simulation toolkits. This has an additional advantage, as it facilitates model replication (in the sense that a simulation model can be executed on different simulation toolkits, to see whether the results do not deviate from one platform to another).

This paper complements [12] and [11] by presenting experimentation results that validate the feasibility of our approach for modeling and simulation of artificial societies as multi-agent systems (MAS). The validation has been made by replicating the same model on two different toolkits, in this case, Repast [10] and Mason [7]. In this way, we show that the proposed agent-based modeling language can be independent of particularities of specific toolkits. This work also shows interesting issues related with simulation facilities provided by the toolkits we used, which can facilitate or complicate the translation process.

Section 2 briefly presents a case study and describes the model replication results obtained during the validation process, which considers whether the transformed

simulation models implement the specification correctly. Section 3 discusses simulation facilities of the two simulation toolkits that have appeared to be determinant during simulation implementation. Section 4 summarizes our contributions and concludes with the identification of further work.

2 Validation Process of Model Translation

In agent-based social simulations, scientists, in general, are presented with already emerged collective phenomena and must theorize about micro-rules that can generate them. In this way, they model behaviors at the individual level, whose combination may result in a complex system behavior. To this extent, we are considering examples of systems where emergent properties of local interaction among adaptive agents influence one another in response to the influence they receive. Here we show the results of the work on one such model, which has been replicated in two different simulation toolkits, RePast and Mason. This case study is based on a system originally described by [2].

This model examines which type of mental constructs corresponds to the norm of reciprocity within the simulation-based study of altruism: a simple or sub-cognitive altruistic algorithm, or smart, or quasi-cognitive. In the former, agents apply routines under given conditions. In the latter, agents execute actions to achieve their goals. Thus, the system studies altruism among simple and smart entities in a society. In this case study the society consists of a population of vampire bats (the agents) that live in roosts, where they get back to after hunting and perform social activities like grooming and sharing food. The entities explicitly modeled as agents are the bats. Roosts are modeled as aggregates of bats. In roosts, bats are allowed to share food and to groom one another.

Each simulation cycle includes one daily and one nightly stage. During the daily stage, the simulated bats perform the social activities, in the night they hunt. Each night 93% of the population will find food to survive until the next hunt. The remaining 7% will starve, unless they receive help from some fellow (under the form of regurgitation). A starving bat will turn to grooming partners for help, and will avoid death if any of them is found to be full. Help-giving allows animals to achieve credits, which will be extinguished if and when help is returned. Animals are endowed with social knowledge: consisting of a memory of past grooming and food-sharing interactions, and of consequent credits. After each simulation the number of death bats, the number of altruistic acts performed and the number of credits turned on or off are recorded.

The experiment in this case study is the dynamic variant of smart agents in [2]. The smart algorithm implemented a small number of rules modifying the values of agents' motivational force for pursuing goals, in particular of the normative goal [12]. This was supposed to increase or decrease as an effect of others' impact on one's own conditions. If one receives help, the force of the altruistic motivation increases, whereas it decreases if one is denied help. This effect is even stronger, if one is not reciprocated and its credits are therefore not extinguished.

The main questions are: What happens under these circumstances? What are the effects of goal-dynamics on altruism, and more specifically on the fitness of the smart population?

2.1 Simulation Results from Reimplementation

The results given by [2] showed that different strategies (Cheater, Prudent, Fair, Generous, Martyr) emerge in the experimental condition, and their difference increases when strategies are inherited by the agents' offspring. These strategies correspond to different patterns of relationships among agents' goals.

The implementations to compare results were generated for Repast [10] and Mason [7], both generated from different visual models. The mechanisms used for code generation are explained in [11]. Here we show the execution of both applications with the same parameters: 10000 days, a population of 150 bats, 10 roost, reproduction mechanism and inheritance activated.

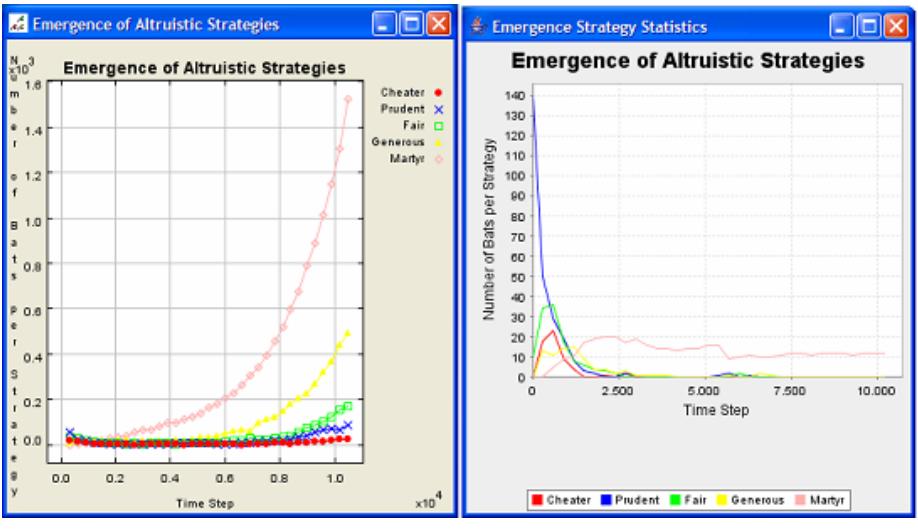


Fig. 1. Emergence of Altruistic Strategies with inheritance. On the left side Repast statistic graph, on the right Mason graph. Values are number of agents per strategy (y), and simulation steps (x). Population is composed of 10 roosts of 15 agents each. Every agent is Prudent at the beginning.

First test results are shown in Fig. 1. On the left side we can see the statistic graph generated with Repast simulation, and on the right the one obtained from Mason.

Based on the results presented in the original model, it is clear that Repast simulation results aligned very well with the original ones. In both conditions, the dominant strategy appears to be Martyrdom. Since this strategy is self-reinforcing one would expect this result: as soon as Cheaters start to decline, donations will increase and reciprocity will emerge. Consequently, Martyr bats will grow spontaneously. Mason statistic results did not align with the original ones as we can see in Fig. 1. Note the difference in scale between both y axes, in Mason chart, population decreases even thought reproduction is activated. This could be due to an inadequate

specification, like missing parameters in the specification, or to an erroneous implementation, in this case an erroneous transformation of the visual model.

We cross checked both independent simulations and even though they were implementing the same underlying process, Mason implementation did not produce results that were sufficiently close neither with the original model nor with Repast results. We had high degree of confidence in the similarity of both implementations, but one of them show significant differences from the original published results and the other one aligned very well, so we speculated that there was a possible source of inconsistency introduced by random number generators or the simulation engine itself, so we debugged the second implementation to find possible errors.

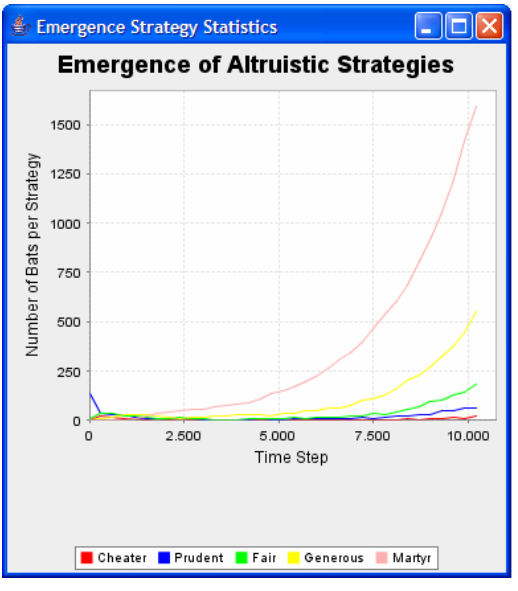


Fig. 2. Mason Statistic Graph without biases. Emergence of Altruistic Strategies with inheritance. Values are number of agents per strategy (y), and simulation steps (x). Population is composed of 10 roosts of 15 agents each. Every agent is Prudent at the beginning.

This task showed some issues (see section 3) on Mason toolkit that were introducing some biases on the results, and having a second implementation to compare with can help to find them and discard translation mechanism errors. On Fig. 2 we can see Mason statistic chart once we handled the bias issues and generate adequate code.

The second test we performed to validate our framework' translation tool was concerned with the reproduction system of the original model. In the first test, we implemented reproduction with inheritance of strategy and credits. This means that offspring gain credits transmitted by their parents. In the second test, inheritance was turned off. The results are shown in Fig. 3. Without inheritance, strategies will not differentiate, and the population gradually and slowly decreases.

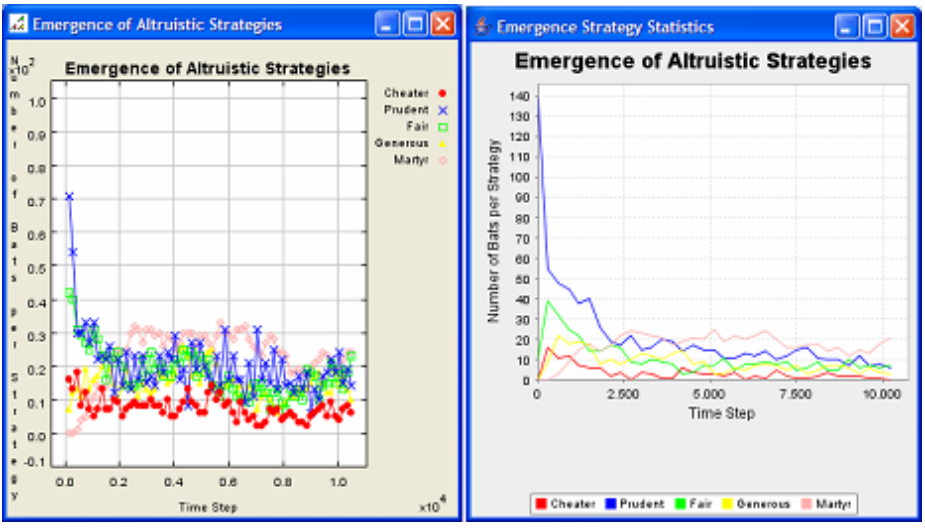


Fig. 3. Emergence of Altruistic Strategies without inheritance. On the left side Repast statistic graph, on the right Mason graph. Values are number of agents per strategy (y), and simulation steps (x). Population is composed of 10 roosts of 15 agents each. Every agent is Prudent at the beginning.

Having two independent simulations re-implementations made it possible to discover shortcomings of the translation mechanisms and greatly aided us in finding possible errors introduced by external sources, like the simulation toolkit itself.

3 Multi-agent Based Simulation Toolkit Facilities

Most multi-agent based simulation environments are *time-driven* discrete event simulation tools where the simulated time is advanced in constant time steps for simulating the perception-reaction cycle of agents who act by the passage of time. The simulated entities are modeled and implemented in terms of agents that interact with each other and with their environment, that is, in each time step agents are scheduled to execute their autonomous actions.

An agent in the context of these rapid development environments use to be an ordinary object, ranging from having no communication at all between entities to communication via the environment, although some of them do provide libraries to implement an agent communication mechanism. Also, this development tools provide the notion of space, letting each entity be assigned a location in the simulated physical geometrical space. More sophisticated mechanisms, like learning abilities for agents, have to be implemented by the developer; besides, no modeling concepts such as beliefs, desires, and intentions, which are common to express agent's behavior, are supported, Table 1 summarizes the main characteristics provided by existing simulation environments for the social domain.

Table 1. Main characteristics provided by existing simulation environments for the social domain

Scheduling	Environment	Communication	Agents Model
Time-driven	Represented with the notion of *Spaces*	Indirect communication through the environment	An ordinary object

Due to this simplicity on the agents concepts that are considered by existing simulation environments, some authors [3] question whether these are really computational agents as those found on multi-agent systems theory or on distributed artificial intelligence. Although this is a logical question, we also think that this kind of agents can be develop, some open Java libraries toolkits like Repast or Mason provide means to develop more sophisticated MAS concepts, so we developed our own abstract layer [11] on top of toolkits facilities to consider concepts from our agent-oriented modeling language, like agents' goals, beliefs, organizations, roles, etc.

The main facilities provided by these toolkits are oriented to *scheduling*, *space* and indirect *communication* mechanisms. Following, we review the most relevant for our code generation approach.

3.1 Scheduling

All multi-agent simulations need some way of ordering events, but often this scheduling is concealed in the design. Repast and Mason make *scheduling* explicit. This could seem inappropriate because MAS have no central control and having a scheduling mechanism could be confused as having a central planner, however, this is not the intention of the *scheduler* in simulation toolkits. The definition of a model period is related with the timing of interactions, and not with a "social" planner that decides who performs what actions. Besides, the order of agents' activation is systematically randomized from period to period (each step) in order to avoid the reproduction of biases.

If global behavior of the system is said to emerge from agents and their interactions, why the model is not just instantiated, letting the agents interact and monitoring what happens without the scheduler functions? Doing so certainly provides an emergent behavior, but social scientists could not be able to understand under what circumstances or what local interaction among agents gives rise to the global emergent structure. Besides, to discover the consequences of their theories in the artificial society modelers need to have control over the simulation to observe what is happening in every moment and what parameters influence the results.

The *scheduling* mechanism in Repast includes a fully concurrent discrete event scheduler. This scheduler supports both sequential and parallel discrete event operations and is responsible for all the state changes within a Repast simulation. Consequently, any changes we want to occur in a simulation must be scheduled with a schedule object. Scheduling offers a way of integrating the actions of many different agents in many different levels of the simulation. In essence, scheduling consists of setting up method calls on objects to occur at certain times.

Repast scheduling mechanism although flexible enough is contrary to some of the ideas that motivated a multi-agent based approach since creating a list of actions to be taken at each time step looks more like a central coordinator approach. However, we are not committed to do it. Therefore, in our case we were able to schedule the agents themselves for execution.

The reproduction mechanism in our model required of dynamic scheduling support. Although rescheduling worked fine, the *ordering* issue of agents' executing in the same time step lacked some ideal characteristics, like the possibility of scheduling agents at different ordering numbers. Now, it only supports *last* ordering parameter, which means that agents scheduled with this parameter on will be executed last. Subdividing a time step in different orders is an ideal characteristic, in our case we required to represent a time step like a day, with two stages, the day and night stage.

Mason *scheduling* mechanism is a single-process discrete-event simulation. It does not schedule events on the scheduler to send to an agent; rather it schedules the agent itself. Thus, it is more in alignment with MAS fundamentals, and so with our MAS framework for modeling and implementing agent-based simulation systems, facilitating the translation mechanism. In this way, an agent is seen like a computational entity that may be scheduled to perform some action, and that can manipulate the environment.

Mason provides many facilities for scheduling, including different *ordering* numbers of scheduled agents in the same time step, as well as dynamic scheduling. However, during the test of our code generation system, we found an issue that introduced biases in our simulation results and that were related to dynamic scheduling and ordering.

When we dynamically scheduled agents' offspring during the 20th month (in order to model the juvenile phase) to be executed every time step after their birth in the same order as parents (in this way, parents and offspring will be randomly executed together, all of them were bats and our intention was they to be treated indistinctly), the ordering mechanism made a distinction between the same *order* number defined at different time steps, so the parents will be executed randomly between them, and offspring would do the same, but never like a whole list of agents scheduled in the same order, it seemed like ordering in different time steps for the same order number created a sub-ordering. This is not a bug of the system, this behavior is already pursued by the toolkit developers since the *Sequence* data structure we used to schedule parent agents is not a dynamic data structure, consequently when we added bats' offspring they were placed in a different internal data structure. This fact is not stated in Mason documentation, and let to misunderstandings, which introduced errors difficult to debug in our system.

3.2 Space

The *Space* describes the spatial arrangements of agents in the model. It is like a controlled environment where agents interact. Note that we do not explicitly state that agents are physically in the environment, though they could be.

In Repast toolkit, the space class controls the environment in which the action takes place. It is theoretically possible to omit the explicit 'Space' object, but this is rarely done. Spaces in Repast function as a collection of agents that define the spatial relationship of agents relative to each other.

Mason Spaces relate arbitrary objects or values with locations in some notional space, although its use is entirely optional. Agents located in spaces are referred to as *embodied agents*, in this way agents are brains, and do not need to be bodies.

In this sense, we completely agree with Mason philosophy of spaces, providing the possibility of having agents not necessarily situated in a space. In this case, agents' communication mechanisms have to be implemented like direct interaction mechanism between them and not through the space like embodied agents.

Both simulation toolkits provide enough facilities to handle spaces. However, for our purpose of translating design artifacts from a visual agent-oriented model to source code [11], Mason facilitates much more the task, specially its support for social networks formation and management.

We did not pretend to provide an exhaustive survey of simulation toolkits in this work, but to evaluate some of the core features more commonly found in most of the simulation toolkits that support a MAS simulation paradigm and which should be well understood to accomplish the transformation from agent-oriented modeling to simulation platform code.

4 Conclusions

In this paper we presented experimental results of the validation process of our approach for agent-based social simulation. The whole development process is driven by modeling artifacts that are represented with visual elements of a modeling language and that are then transformed into different implementations for particular simulation toolkits. Describing social phenomena and implementing the simulation in this way facilitate the use of toolkits by users who are not necessarily experts in computer programming, but in social sciences. Also, model replication issues are easily addressed with this approach.

This is illustrated with a social system case study, for which two implementations in two different simulation platforms have been derived. Initially, we chose Repast and Mason toolkits because both tools are of widespread use for agent-based simulation, and because their way of specifying agents can be supported by the INGENIAS code generation approach without too much difficulty.

These experiments showed the feasibility of the model driven approach presented in [11] and enabled the study of particular aspects of each simulation toolkit relevant to multi-agent based simulations. More specifically, we evaluated facilities provided by simulation toolkits that can have impact on the transformation process from models to implementation in different target platforms, like scheduling techniques.

Also, we compared the simulation results of our implementations with the original work to discover possible misunderstandings in models specifications or interpretations. Aligning the models proved to be a very rewarding task since it led us to discover biases introduced by simulation mechanism found in most simulation platforms.

There is still work to do to reach full automation of the code generation step. We plan to include also deployment support and monitoring of the transformation process. Deployment information is independent of the process defined in the model and should be defined through a customizable tool.

References

1. Axtell, R., *Why Agents? On the Varied Motivations for Agent Computing in the Social Sciences*, in *Working Paper No. 17*. 2000, center on Social and Economic Dynamics, The Brookings Institution: Washington, DC.
2. Di Tosto, G., R. Conte, M. Paolucci. *Altruism Among Simple and Smart Vampires*. In: *Proceedings of the 1st Conference of the European Social Simulation Association (ESSA)*. Groningen, NL (2003),
3. Drogoul, A., D. Vanbergue, T. Meurisse. *Multi-agent Based Simulation: Where Are the Agents?* In: J.S. Sichman, et al. (Eds.): *Multi-Agent-Based Simulation II: Third International Workshop, MABS 2002*. Lecture Notes in Computer Science, Vol. 2581. Springer. Bologna, Italy (2002), 1-15.
4. Epstein, J.M., R. Axtell. *Growing artificial societies: social science from the bottom up*. Complex adaptive systems. 1996, Washington, D.C.: Brookings Institution Press.
5. Gilbert, N., K.G. Troitzsch. *Simulation for the Social Scientist*. 1996, Buckingham, U.K.: Open University Press.
6. Holland, J.H. *Emergence: from chaos to order*. 1998, Reading, Massachusetts: Addison-Wesley.
7. Mason. 2004. Mason: Multiagent Simulation Toolkit. Available from: http://cs.gmu.edu/~eclab/projects/mason/
8. Moretti, S. *Computer Simulation in Sociology: What Contribution?* Social Science Computer Review, 2002. Vol. 20(1): p. 43-57.
9. Pavón, J., J. Gómez-Sanz. *Agent Oriented Software Engineering with INGENIAS*. In: V. Maríık, et al. (Eds.): *Multi-Agent Systems and Applications III, 3rd International Central and Eastern European Conference on Multi-Agent Systems, CEEMAS 2003*. Lecture Notes in Computer Science, Vol. 2691. Springer. Prague, Czech Republic (2003), 394-403.
10. Repast. 2004. Repast: Recursive Porus Agent Simulation Toolkit. Available from: http://repast.sourceforge.net
11. Sansores, C., J. Pavón. *Agent-Based Simulation Replication: a Model Driven Architecture Approach*. In: A. Gelbukh, et al. (Eds.): *Fourth Mexican International Conference on Artificial Intelligence, MICAI-2005*. Lecture Notes in Artificial Intelligence, Vol. 3789. Springer-Verlag. Monterrey, México (2005), 244-256.
12. Sansores, C., J. Pavón, J. Gómez-Sanz. *Visual Modeling for Complex Agent-Based Simulation Systems*. In: J. Sichman, et al. (Eds.): *Multi-Agent-Based Simulation, Sixth International Workshop on Multi-Agent-Based Simulation, MABS 2005*. Lecture Notes in Computer Science, Vol. 3891. Springer-Verlag. Utrech, Netherlands (2005), 174-189.

Agent Behavior Representation in INGENIAS*

Jorge J. Gómez Sanz, Rubén Fuentes, and Juan Pavón

Facultad de Informática,
Universidad Complutense de Madrid
{jjgomez, ruben, jpavon}@sip.ucm.es
http://grasia.fdi.ucm.es

Abstract. Nowadays, we have different agent oriented methodologies that enable developers to produce agent oriented designs. One of the recurrent problems of these methodologies is how to describe the behaviour of agents within a system. A developer needs primitives to express autonomy, proactivity, and social concerns of his agents, but there are problems in understanding what does these elements mean, beyond any natural language explanation. There is a clear need of semantic models understandable by average engineers. These models could help in foreseeing the impact of autonomy with respect system goals, or determining if, in an agent specification, a task will ever be scheduled for execution. This paper presents a proposal of semantic model for the visual modelling language used in INGENIAS, a project started in 2002 and considered the inheritor of MESSAGE/UML.

1 Introduction

When describing a Multi-Agent System, MAS from now on, with software engineering approaches, a first concern consists in providing primitives to represent a static view of the system, such as the system structure or initial knowledge of the agents, and dynamic aspects, as the behaviour of the agents. Known formal techniques, such as Z, Petri Nets, and modal logic, have been applied to agents but their success has been limited. Petri Nets, and Finite State Machines in general, are applied mainly to model protocols, but, as Singh [12] remarks «*Such protocols are verifiable, but are devoid of content. They only state how the tokens are ordered*». Z has been used to model agents, like in [3], however, understanding Z requires some training and recognising agent concepts in a Z notation is not trivial. There is a stronger tradition of using modal logic to describe semantic models, concretely temporal modal logic as remarks [13]. Nevertheless, understanding and mastering a modal logic description requires even more training than Z, and there are less support tools.

Our hypothesis in this paper is that a state based description of the behaviour of an agent can be better understood than a modal logic or Z based one. Also, to ensure the success of such description, it should be used agent oriented understandable

* This work has been funded in part by the Spanish Council for Science and Technology undergrant TIN2005-08501-C03-01.

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 89 – 98, 2006.

metaphors. In the elaboration of these metaphors, we propose applying two formalisms: one to present agent specific constructs to the developer and a different one, more precise, to describe their semantics. In this proposal, the first formalism is meta-modelling and the second PROLOG, a logic programming language widespread and with a good tool support. This combination should enable a greater number of developers work with agent concepts.

Meta-modelling is currently applied in several agent oriented methodologies and their use increases with time. Two reasons for its acceptance are the existing support for any meta-model (the use of stereotypes in UML tools allows to reuse UML notation for agent concepts, for instance) and its similitude with existing notations for object oriented systems (it decreases the learning curve of agent concepts). Though meta-models are semi-formal, this has not been an obstacle for agent developers. However, at some points in the development, especially when evaluating the correctness and consistence of a specification, formal semantics are needed.

The primitives we will use for presenting agent behaviour to developers will be those defined in INGENIAS [6]. It provides modelling primitives that allow to apply the model from [11] as well as integrating some key constructs from the BDI paradigm, i.e. goal satisfaction by means of tasks and selection of tasks according to the goals they connect to. Besides, INGENIAS incorporates interactions, workflows, organizations, and other meaningful concepts extracted from the agent research literature, and shows how these concepts relate to the BDI paradigm.

As semantic model for these concepts, initially, we intended to follow the pure BDI model [10], which describes agents in terms of Believes, Desires, and Intentions. When studying BDI, we found problems in identifying the state of the agent in this approach, mainly because the original model is described in modal logic with possible worlds semantics. In these semantics, the state concept is rather fuzzy. Though it is not the same research line, the work from [11] gives us a better starting point for obtaining a set of abstract components as a base for a state based description of the agent behaviour. It provides a state based description of an agent by using new metaphors to represent the behaviour of an agent. This notion is easier to implement and would admit the integration of BDI concepts we are interested into. Hence, we assume it will be more easily understood by developers.

Once decided what the model would be about, we proceeded with a study of some representative agent oriented methodologies to verify to what extent state based representations where considered and how original our proposal was. Using [11] as a reference, the study gathers data about how existing solutions for agent modelling deal with agent mental state representation, its evolution, and control, i.e. how the agent takes decisions. The results are presented in Fig. 1. From this study, we draw some conclusions: (a) the concept of mental state, even internal state, is not considered in depth, despite control structures found require, at least, some notion of state; (b) agent control in general bases on the definition of rules or tasks; and (c) there are no explicit primitives to describe or contain descriptions of the evolution of agents along the time. INGENIAS provides constructs to deal with these aspects, however, it lacks of an appropriate semantic model, though INGENIAS specifications have been implemented in several frameworks and languages.

To introduce the proposal in detail, we present in the second section a synthesis of the concepts applied in INGENIAS to describe agents. Following, section three describes the evolution of an agent mental state and what it has to do with mental state management. Fourth section introduces a formal definition of the semantics of agent behaviour related primitives in INGENIAS. Fifth section contains a review of related work. Finally, sixth section contains the conclusions.

	Mental State	Evolution	Control
Agent0	Symbol based representation by means of reified temporal logic. Mental entities are: believes, compromise, and capability.	The are operations for adding knowledge (inform) and to knowledge management, though the later is not specified. It also talks about removing compromises (*refrain*)	Rules (*IF* actions) associated to the agent mental state. Rules execute actions taking into account the satisfaction of a logic formulae whose terms are mental state entities
Vowel Engineering	Not considered explicitly. They talk about different agent architectures, from cognitive ones to finite state machines, but they are not described in detail	Not considered explicitly	No control appears explicitly. It is associated to how interactions are conceived, with Petri Nets.
MESSAGE	There are ontologies to capture believes, goals, compromises, and tasks. UML is the notation to represent the mental state	It mentions the use tasks to satisfy goals.	Task-goal associations are defined. Control mechanisms consists in satisfying a set of goals which are structured in a is-subgoal-of hierarchy.
MAS-CommonKADS	There are goals, services, and capabilities. It is not explained how these concepts are implemented or their semantics All descriptions are in natural language	There are no explicit mechanisms to manage goals. There are vague references to them in the *experience model*.	A *experience model* to describe reasoning capabilities.
BDI	Believes, intentions, Goals, and Plans.	Though it is not explained how, it is assumed the capability of creating, modifying, and removing believes, goals, and intentions.	Only those tasks that make the agent achieve a desire are executed. Tasks modify the mental state of the agent. Tasks can be grouped into plans. Task scheduling mechanism is not detailed.

Fig. 1. Survey of the support of mental state, mental state evolution, and agent control in a representative set of agent oriented methodologies. The full description is available at http://grasia.fdi.ucm.es/ingenias.

2 Describing an Agent in INGENIAS

An agent is an autonomous entity which has a purpose and means to achieve it. This definition bases on the work of Newell [9], which defines an agent as an special entity living in the knowledge level and that behaves according to the *rationality principle*.

Agents do have a mental state, as in [11], which is defined as an aggregate of mental entities. A mental entity in INGENIAS is the root of a sub-hierarchy of

INGENIAS entities. Mental entities are refined into: elements that allow the agent to describe the state of the world (*belives* or *facts*); the agent internal state or other agent's internal state (*believes* or *facts*); control information needed by the agent itself, like *goals* representing a state of the world that the agent wants to achieve; and *compromises*, i.e. an obligation to other agent.

To describe the evolution of the mental state and interpreting it, which is not described in detail in [11], we use two components: the *Mental State Processor* and the *Mental State Manager*. The first is responsible of deciding what to do next taking into account available information, i.e., the mental state. A *decision* here is the act of choosing which task or tasks, from those known by the agent, has to be executed in the current situation to satisfy the goals of the agent.

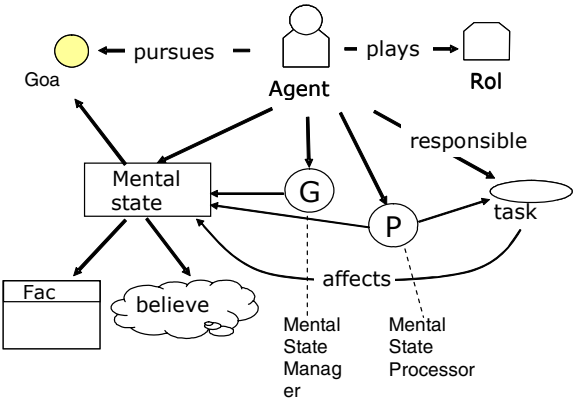


Fig. 2. Elements needed to describe an agent in INGENIAS. The tools used in INGENIAS uses this notation as well as its equivalent with UML classes and stereotypes. A more formal description of these elements, following the Meta-Object Facilities standard for metamodeling, appears in http://grasia.fdi.ucm.es/ingenias/metamodel.

The second component is responsible of defining means to add, remove, or modify entities from the mental state and ensuring its internal consistency, if needed. This entity would execute these operations in a consistent way, being the developer the responsible of defining what is consistent in each situation. A trivial example of consistency maintenance would be not allowing two instances of different mental entities with the same id.

A developer would have freedom of choice on defining more precise meaning of these components. This could be done in different ways in INGENIAS: by associating these components with others whose behaviour is known or by describing its behaviour with other INGENIAS diagrams. An example of the first situation would be assuming that the behaviour of the mental state processor is equivalent to a AI planner, perhaps the STRIPS [4] planner. In this case, it would make sense to associate to the Mental State Manager proper operations of this specific planner: adding and removing facts.

An example of the second way would be describing these components in terms of tasks that consume and produce different mental entities, especially goals. In

INGENIAS, it is not possible to describe meta-tasks, i.e. tasks that produce/manage other tasks. Allowing this would be equivalent to let an agent create new atomic behaviours by itself, which is hard to specify with or without methodologies, since it deals with machine learning techniques. On the other hand, it is possible to describe new goals and mental entities, influencing, perhaps, which tasks will be selected next. This way we achieve with this model certain degree of capability of sketching learning behaviours. The description of these tasks, which we call control tasks, can be done by means of existing diagrams of tasks/goals which the methodology already incorporates.

3 The Evolution of the Agent

So far, it has been stated that an agent has a mental state, that there are entities controlling its evolution, the *Mental State Manager*, entities changing it, the *tasks*, and entities deciding what to do next, the *Mental State Processor*. Nevertheless, from the point of view of a developer, it is necessary to mention, as well, which states the agent is going to get through. Being an agent a situated entity, i.e. an entity embedded in an environment, a developer should expect that his agent perceives new information not considered in design time. This new information may be the result of actions of other agents, actions of the agent itself, or just be a spontaneous change in the environment.

This problem cannot be avoided, since it is not possible to know in advance the unexpected. In our approach, we describe which mental states our agent should achieve along its live. If we ensure these states, actions performed by the agent should proceed as expected with a reasonable guarantee. Mental state descriptions can be as simple as the enumeration of the mental entities that has exist in runtime, or a complex boolean expression to be satisfied with information extracted from the existing mental entities in a certain moment.

This new need, thinking about agents in runtime, requires concepts to represent in design time agents which will exist in runtime, just as in object oriented notations you can talk about class instances, i.e. objects, in design time, though these only exist in runtime. INGENIAS assumes that any agent can be instantiate as many times as the developer requires. Each one of these instances can evolve towards behaviours different from the original one if an only if either each agent is deployed in a different environment, or there are definitions of mental state manager or processor that incorporate certain degree of learning.

Autonomous Entity Query is the first class citizen concept INGENIAS use to refer a *concrete agent*, i.e. an existing agent runtime instance with a specific identifier, or a *set of agents*, described with a query intended for a yellow or white pages service and to be answered in runtime. Both can be associated with mental states. Such associations are translated as *agent X or the set of agents Y, in runtime, must satisfy this mental state*. Obviously, there are also links of these runtime instances with their design time counter pads.

3.1 INGENIAS Diagrams for Representing Agent Mental States

Using elements from the introduction of this section and section 2 permits to describe agent behaviour to a reasonable extent. They are used in special diagrams which can be built with the INGENIAS Development Kit (IDK), a set of tools that allows modelling a MAS and mapping specification to code:

- **Description of initial mental state.** This initial mental state is build up with a set of believes, facts, goals, and compromises. The description of this diagram is made with an INGENIAS agent diagram.
- **Description of intermediate mental states.** This is done by means of the same kind of diagrams, INGENIAS agent diagram, but replacing the *agent* concept with the *autonomous entity query*, concept.
- **Evolution description.** It is about ensuring that the tasks incorporated in the specification can produce the evidences needed to reach each described mental state and satisfy declared goals. There are situations in which reaching a goal depends on the occurrence of a concrete event. Therefore, it is not possible to ensure to in advance that described mental states will be achieved, but at least, it can be said that if all goes well, it will behave as expected.

We omit here references to other diagrams that can be built with current IDK version. In this version of the semantic model, we centre into the tasks that an agent executes and how they affect its internal state.

4 Semantics

The concrete semantics of the elements from section 2 and 3 is hard to express because INGENIAS bases on a semi-formal specification technique: meta-modelling. This technique ensures a proper construction of diagrams but says nothing about the concrete semantics of its constituents. Concrete semantics of an *agent* concept, for instance, are expressed in INGENIAS by means of natural language descriptors. As a consequence, any INGENIAS specification can be considered ambiguous to certain extent. In this section, we overcome this ambiguity by associating the specification to a PROLOG equivalent, which is more valid as specification language and it is even executable. Other alternative specification methods have been already discussed in the introduction. With respect alternative already built semantic models, they will be referred in the related work section, section 5.

The basic lifecycle of the agent is shown in Fig. 3, something very similar to the one shown in [7]. It is a possibly infinite sequence of transitions from one mental state to another. Environment appears by means of the *perceives* predicate. This predicate would be responsible of adding new information to the agent, if there is any. If a after a fixed amount of time nothing happens, a timeout triggers and the predicate returns an empty list.

The evolution of the mental state is described into Fig. 4. It would consist of several calls to the predicate *evolves* with the initial mental state of the agent as

parameter. The resulting mental state would be the second argument of the predicate. Of course, the resulting mental state could be the same as the initial. This would be interpreted as if the agent were waiting for some event from the environment.

```
generalLyfecycle(mentalState(InitialMentalState), mentalState(NewMentalState)):-
        perceives(NewInformation),
        append(NewInformation, InitialMentalState,UpdatedMentalState).
        evolve(mentalState(UpdatedMentalState), mentalState(NewMentalState)).
```

Fig. 3. General lifecycle of an agent. Perception is an I/O operation that adds new information to the agent mental state.

```
evolve(InitalMentalState,FinalMentalState):-
        mentalStateProcessor(InitialMentalState,Task,Goal,Condition),
        execute(InitalMentalState,Task, FinalMentalState),
        manageMentalState(add,
                [believe([executing(Task),satisfy(Goal),when(Condition)])],
                FinalStateExecute,
                MentalStateTaskExecution),
        mentalStateManager(MentalStateTaskExecution, FinalMentalState).

evolve(EM,EM).
```

Fig. 4. Description of the evolution of the agent. Mental state processor selects a task which is executed next. The result of the task is added to the mental state. It could be possible as well that nothing could be done at this moment. This is handled with the second predicate which leaves the mental state as it was before.

Fig. 4 also clarifies the role of mental state processor, tasks, and the mental state manager. There is a queue of tasks represented as some believes of the agent. The agent believes that it has started a task *Task.* He also believes that this task will make the agent reach the goal *Goal* when the agent mental state satisfies the condition expressed in *Condition*. Such conditions would be extracted from intermediate agent mental states. These believes are important towards deciding which tasks will be executed following (Fig. 6) or deciding if a goal has been reached (Fig. 5).

As it was said before, the description of the mental state manager and processor could be made in terms of control tasks, see section 2. Due to the paper length constraints, we have simplified the number and type of these tasks. Fig. 5 presents and explains basic management tasks: add/remove mental entities, goal management, and consistence management (duplicate removal).

The Mental State Processor responsibility would be choosing a single task with which the agent can satisfy some goal. Task management is simple (Fig. 6). It consists in, first, checking that required conditions for executing a task hold and, second, adding the output of the task, i.e. mental entities, to the mental state. If there were not an executable task, according to PROLOG semantics, there would be a backtracking until finding another task or failing. The backtracking, after the failure, would lead to the second predicate *evolve* in Fig. 4. We interpret this situation as a skip action in which the agent decides to do nothing.

```
manageMentalState(add,
      Concept,
      mentalState(Ls),
      mentalState (Os)):-append(Concept,Ls,Os).

manageMentalState(remove,
      Concepto,
      mentalState(Ls),
      mentalState (Os)):-delete(Ls,Concept,Os).

manageMentalState(noDuplicates,
      mentalState(Ls),
      mentalState (Os)):-sort(Ls,Os).

manageMentalState(manageGoals,
      mentalState (Ls),
      mentalState (Os)):-
      member(goal(G),Ls),
      member(believe([ejecutando(T),satisfy(goal(G)),when(Condition)]),Ls),
      isSubList(Condition,Ls),
      B=believe([executing(T),satisfy(goal(G)),when(Condition)]),
      manageMentalState (remove,B, mentalState(Ls), mentalState(O1s)),
      manageMentalState (remove,goal(G), mentalState(O1s), mentalState(Os)).

mentalStateManager(InitialState,FinalState):-
      manageMentalState (manageGoals,
            InitialState,
            FinalStateTemp),
      mentalStateManager (FinalStateTemp, FinalState).

mentalStateManager(InitialState,FinalState):-
 manageMentalState(noDuplicates, InitialState, FinalState).
```

Fig. 5. Mental state management defines the meaning of basic operations to be applied to the mental state. In this case, it declares how to add and remove entities, what to do with goals, and how to maintain consistency (no duplicates). A *goal* is reached when the mental state of the agent is that described in a believe inside the term *Condition*. Duplicate removal is implemented with a sort operation whose side effect is removing duplicates. Adding and removing entities is implemented as adding/removing operations of a list data structure.

```
execute(Input,Task,Output):-
      produce(Input, Task, Produced),
      manageMentalState (add,
                        Produced,
                        Input, Output).

mentalStateProcessorl(mentalState(Es),T,goal(G),Condition):-
      member(goal(G),Es),
      task(T),
      satisfies(T,goal(G),Condition),
      not(member(believe([executing(T),satisfy(G),_]),Es)).
```

Fig. 6. Description of the mental state processor. It selects a goal and a task able to satisfy the selected goal. Selected task is returned as result. By now, we consider only tasks that produce concrete outputs, though INGENIAS also provides definitions for tasks modifying existing mental entities.

5 Related Work

Since the semantics are BDI based, we could have used one of the declarative implementations of the BDI model, like those cited in [8], and translating INGENIAS diagrams to that BDI implementation. Though valid, this approach has an important limitation. In INGENIAS, the developer is responsible of defining what kind of mental state management is more adequate to the problem domain. By choosing a language, such as *Jason* [1], then the mental state management and how decisions are taken is constrained by the built-in mechanisms of the language. INGENIAS intends to decouple design decisions from the semantic model. We enable this way decisions like the one exemplified in section 2, making the mental state processor a STRIPS planner with its own semantics. With the current approach, the developer can adapt this semantic model to the design requirements of the problem.

There is a strong similitude between the task-mental state approach shown in this paper and the blackboard-data source approach from [2], but they are different. The main differences between this approach and blackboard architectures are the interpretation of what a mental state is and what drives the modification of this mental state. In the mental state of an agent there can be only certain types of entities defined by the developer in design time. Besides, each entity has specific semantics that constrain how they can be used. For instance, a *believe* is not the same as a *compromise*. The first contains information about how the agent sees the world or about its internal state. The second implies the intention of satisfying an acquired obligation towards another agent. With respect the modification of the mental state, it is performed by means of an entity that resembles a process, the task. In a blackboard architecture, processes are triggered by changes in the black board. Here we require something more, since tasks are triggered and chosen because they intend to satisfy a concrete goal.

6 Conclusions

This paper has presented the mechanisms that INGENIAS provides to represent agent behaviours. The paper has introduced a semantic model described in PROLOG to declare the exact meaning of INGENIAS entities in the specification. This semantic model bases on the description of a mental state manager and a mental state processor, and allows developers to customize the representation to their concrete needs.

The PROLOG code presented in the paper is not complete, due to the limit of pages of the paper. However, the most relevant parts have been included and they are enough to capture the semantics of an INGENIAS specification of an agent. With this model we expect developers to better understand what their specification stand for. Besides, it is a first step towards the application of formal methods to verify, for instance, properties to be satisfied by the system.

The diagrams mentioned along the paper, are implemented in the INGENIAS Development Kit also known as IDK. This environment is a development distributed under the GNU Public License. INGENIAS methodology and some development examples can be consulted in [5] as well as in the *grasia!* Research group website (http://grasia.fdi.ucm.es/ingenias).

References

1. Bordini, R. and J. F. Hübner (2004). *Java-based AgentSpeak interpreter used with saci for multi-agent distribution over the net*.http://jason.sourceforge.net/Jason.pdf
2. Carver, N. and V. R. Lesser (1992). *The Evolution of Blackboard Control Architectures*. UM-CS-1992-071. Department of Computer Science, University Massachusetts
3. d'Inverno, M., D. Kinny, et al. (1998). *A formal specification of dMARS*. Intelligent Agents IV, Springer-Verlag Berlin. **LNAI 1365:** 155-176.
4. Fikes, R. and J. Nilsson (1971). *STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving*. Artificial Intelligence. **2**.
5. Gomez-Sanz, J. J. (2002). *Modelado de Sistemas Multi-Agente*. Dpto. Sistemas Informáticos y Programación. Madrid, Universidad Complutense de Madrid.
6. Grasia (2004). *GRASIA site and INGENIAS Development Kit site*.http://grasia.fdi.ucm.es && http://ingenias.sourceforge.net
7. Kinny, D., M. Georgeff, et al. (1996). *A methodology and modelling technique for systems of BDI agents*. Seventh European Workshop on Modelling Autonomous Agents in a Multi-Agent World (MAAMAW-96), Springer-Verlag. **LNAI 1038:** 56--71.
8. Mascardi, V. (2005). *A Survey on Languages for Programming BDI-style Agents*. PROMAS Technical Forum, Ljubljana, Slovenia.
9. Newell, A. (1982). *The knowledge level*. Artificial Intelligence **18**: 87-127.
10. Rao, A. S. and M. P. Georgeff (1991). *Modeling Rational Agents within a BDI-Architecture*. 2nd International Conference on Principles of Knowledge Representation and Reasoning (KR'91), Morgan Kaufmann publishers Inc.: San Mateo, CA, USA.
11. Shoham, Y. (1993). *Agent Oriented Programming*. Artificial Intelligence **60**(1): 51-92.
12. Singh, M. P. (2000). *A social semantics for agent communication languages*. Issues in Agent Communication **LNAI 1916**: 31-45.
13. Wooldridge, M. (1997). *Agent-based Software Engineering*. IEEE Proceedings on Software Engineering **144**(1): 26--37.

Agent-Based Modeling of Social Complex Systems

Candelaria Sansores and Juan Pavón*

Universidad Complutense de Madrid, Dep. Sistemas Informáticos y Programación
28040 Madrid, Spain
csansores@fdi.ucm.es, jpavon@sip.ucm.es

Abstract. This thesis proposal aims to provide a new approach to the study of complex adaptive systems in social sciences through a methodological framework for modeling and simulating these systems like artificial societies. Agent based modeling (ABM) is well fitted for the study of social systems as it focuses on how local interactions among agents generate emergent larger and global social structures and patterns of behavior. The issues addressed by our framework are presented as well as its most important components.

1 Motivation

One of the most difficult challenges for understanding social phenomena is their intractably complex nature. The emergence of societies and the establishment of co-operative relationships among their individual components have facilitated the generation of an extremely rich repertoire of behavioral possibilities not present in solitary organisms. The collective action of social individuals produces hierarchical phenomena that extend in time and space well beyond the local environments of the participants. Large-scale processes generated by local-scale interactions without central control, so-called self-organization, may in turn affect the specification of how individuals behave, interact and respond in their most immediate spatial domains. This relationship between individual behavior and macroscopic properties is hard to understand and experiment with.

Social scientists have attempted to understand this complexity with different methodologies based on mathematical equations. Although these methodologies have enhanced social science research, they have failed of capturing emergent behavior and self-organization, and had little success in expressing laws that regulate these phenomena. However, new approaches to the study of complex adaptive systems have emerged. One of the emerging methodologies is the use of *Agent-Based Modeling* (ABM) and computer simulation as a tool for analysis and explanations that are impossible or too difficult to formulate using mathematical functions only. In a computer simulation, the social system is represented with symbols of a programming language and the laws that regulate their behavior are formulated with the algorithms implemented by the simulation program. The system to be simulated is modeled with

* This work has been developed with support of the Consejo Nacional de Ciencia y Tecnología (CONACYT) from México and the project TIC2002-04516-C03-03, funded by the Spanish Council for Science and Technology.

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 99 – 102, 2006.

Agents. In this context, agents are autonomous software systems that are intended to describe the behavior of observed heterogeneous social entities (e.g. individuals and organizations).

An advantage of ABM is the ability to estimate the plausibility of the behavior of agents, the way in which agents interact, and the consequences of that behavior and interaction [2]. ABM is well fitted for the study of social systems as it focuses on how local interactions among agents serve to create larger and global social structures and patterns of behavior. Thus, agent based modeling and simulation allow creating new social worlds, or *artificial societies* [5], by modifying various conditions and parameters, as new needs arise. Thus, in agent-based social simulations (ABSS) the model is a multi-agent system (MAS) which allows the exploration of the micro-to-macro relation [3].

2 Agent Based Modeling Open Issues

As agent-based simulation has gained in popularity, software tools for modelers are emerging. There are various toolkits for developing agent-based simulation systems, like REPAST, ASCAPE and MASON. The design philosophy of these toolkits is to provide a model library to which an experienced programmer can easily add features for simple simulations. These libraries have great advantages for modelers over developing their own, but also have some limitations. They require modelers to have a good working knowledge of the programming language that they are aimed at. Thus, for inexperienced programmers specifying a simulation model in a high-level declarative language instead of a low-level programming language would be desirable. This was the purpose of SDML, which has been built on Smalltalk. Unlike ASCAPE and REPAST, it does not demand users to be fluent in the underlying programming language, but they have to learn a complex interface that can be as difficult to master as a full programming language, which finally limits its usability.

Another kind of tools that have emerged for developing simulations is rapid development environments. These allow the building of simple models using visual programming, for instance, STARLOGO, NETLOGO, CORMAS and AGENTSHEETS. Although they are relatively easy to use, agents in this kind of systems are quite simple, usually with a poor or inexistent agents' cognitive model, and without support to model direct interactions between agents.

Simulations developed with both kinds of tools, either toolkits or rapid-development libraries, are proprietary model descriptions. This makes impracticable to compare two different implementations of the same specification. Therefore, replication of models turns out complicated due to the diversity of specification languages and the lack of a common one that addresses conceptual modeling. One approach that seems to consider this requirement more formally is SESAM. This framework provides an environment for modeling, which is based on UML-like activity diagrams. Although it does a step forward in the modeling issue, it still lacks of certain functionalities which are prerequisites for complex negotiation abilities of complex goal-oriented agents.

In fact, an open debate stays on the need for complex goal-oriented agents. From a computer science view, a complex adaptive system is a form of *complex* multi-agent

system (MAS) with adaptive agents. Complex systems are dynamical systems composed of a high number of agents and complex interactions among them. Also, [6] requires that agents be themselves complex, which is certainly true for the actor of social systems, but in the case of some biology systems (like *swarm systems*) the actors use to be simple. Even social systems models usually contain relatively simple actors. Due to this *simplicity* in agents' implementations, some authors (for instance [4]) question whether these are really computational agents, as those found on MAS or DAI, and arguments sustaining agents are not used to implement agent-based simulations, but only to design them.

Simplicity is not a problem if we consider modeling is a term for simplification. However, regarding MAS discipline we question ourselves if MAS potential is not being leveraged, and the plausibility issue with regard to social simulation also arises, which, in principle, involves modeling human cognition a little more realistically.

3 Proposal

The proposal of this thesis is to address the open issues just reviewed with the following contributions:

1. An easily customizable tool for describing social models with a visual language. This visual language is toolkit-independent for specifying MAS and should support agents that range from simple to complex ones, and consider organizational issues to manage other dimension of complexity (the society of agents and the system's architecture). Simple agents like those already supported by existing agent-based simulation toolkits and complex agents that mimic real agents as much as necessary.
2. An agile methodology for developing simulations, starting on the conceptual modeling, supported by a visual model editor, and ending with the transformation of the specification model in an implementation for any target simulation toolkit, hiding computational complexity to modelers. Because each agent-based simulation toolkit has unique strengths and weaknesses, modelers should be given the opportunity to choose the one that best addresses each particular kind of problem.
3. Replication support, the need for a toolkit-independent visual modeling language is mainly supported by model replication requirements. For instance, [1] pointed out the relevance of model simplicity for replication issues, but we also consider fundamental, besides simplicity, to establish a common language to describe agent-based models applied to the social sciences, independently of the implementation platform. This common language could be also a mean for communication of models among users or tools. Presently, the way social scientists communicate their models for replication is by passing source code to each other, which demands high effort to understand and interpret the models, sometimes leading to misunderstandings.
4. Tools to adapt the visual language proposed to model phenomena in different social domains. We are aware of the fact that having a universal modeling and simulation language for the social domain is not feasible. Therefore, our aim is to provide an agent oriented language that can be customized to particular social

domains, by specialization or addition of new elements, which can be defined by the modelers of such domains.

The basis for achieving these goals is the application of the INGENIAS agent-oriented methodology [7], as it provides methods and tools for multi-agent systems specification and code generation support. INGENIAS is based on the specification and management of meta-models that describe the agent-oriented modeling language and support the transformation towards implementation for multiple target platforms. Here we plan to adapt these meta-models for the simulation problem, and the methods and tools accordingly. The approach has been presented at ESSA 2004 and EUMAS 2004 conferences and the preliminary results have been presented to MABS 2005 [9] and MICAI 2005 [8] conferences.

References

1. Axelrod, R. *Advancing the Art of Simulation in the Social Sciences*. In: R. Conte, et al. (Eds.): *Simulating Social Phenomena*. Lecture Notes in Economics and Mathematical Systems, Vol. 456. Berlin Springer. (1997), 21-40.
2. Axtell, R., *Why Agents? On the Varied Motivations for Agent Computing in the Social Sciences*, in *Working Paper No. 17*. 2000, center on Social and Economic Dynamics, The Brookings Institution: Washington, DC.
3. Coleman, J.C. *Foundations of Social Theory*. ed. M.H.U.P. Cambridge. 1990, Cambridge, MA: Harvard University Press.
4. Drogoul, A., D. Vanbergue, T. Meurisse. *Multi-agent Based Simulation: Where Are the Agents?* In: J.S. Sichman, et al. (Eds.): *Multi-Agent-Based Simulation II: Third International Workshop, MABS 2002*. Lecture Notes in Computer Science, Vol. 2581. Springer. Bologna, Italy (2002), 1-15.
5. Epstein, J.M., R. Axtell. *Growing artificial societies: social science from the bottom up*. Complex adaptive systems. 1996, Washington, D.C.: Brookings Institution Press.
6. Gilbert, N., K.G. Troitzsch. *Simulation for the Social Scientist*. 1996, Buckingham, U.K.: Open University Press.
7. Pavón, J., J. Gómez-Sanz. *Agent Oriented Software Engineering with INGENIAS*. In: V. Marík, et al. (Eds.): *Multi-Agent Systems and Applications III, 3rd International Central and Eastern European Conference on Multi-Agent Systems, CEEMAS 2003*. Lecture Notes in Computer Science, Vol. 2691. Springer. Prague, Czech Republic (2003), 394-403.
8. Sansores, C., J. Pavón. *Agent-Based Simulation Replication: a Model Driven Architecture Approach*. In: A. Gelbukh, et al. (Eds.): *Fourth Mexican International Conference on Artificial Intelligence, MICAI-2005*. Lecture Notes in Artificial Intelligence, Vol. 3789. Springer-Verlag. Monterrey, México (2005), 244-256.
9. Sansores, C., J. Pavón, J. Gómez-Sanz. *Visual Modeling for Complex Agent-Based Simulation Systems*. In: J. Sichman, et al. (Eds.): *Multi-Agent-Based Simulation, Sixth International Workshop on Multi-Agent-Based Simulation, MABS 2005*. Lecture Notes in Computer Science, Vol. 3891. Springer-Verlag. Utrech, Netherlands (2005), 174-189.

Agent-Based Solutions for Natural Language Generation Tasks

Raquel Hervás and Pablo Gervás

Departamento de Sistemas Informáticos y Programación
Universidad Complutense de Madrid, Spain
raquelhb@fdi.ucm.es, pgervas@sip.ucm.es

Abstract. When building natural language generation applications it is desireable to have the possibility of assembling modules that use different techniques for each one of the specific generation tasks. This paper presents an agent-based module for referring expression generation and aggregation, implemented within the framework of a generic architecture for implementing multi-agent systems: Open Agent Architecture.

1 Introduction

Natural language generation (NLG) is subdivided into several specific subtasks [1], and each one of them operates at a different level of linguistic representation - discourse, semantics, lexicon, syntax... Natural language generation can be applied to domains where the communication goals and the characteristics of the texts to be generated can be very different, from the transcription into natural language of numerical contents [2] to the generation of literary texts [3].

Each type of NLG application may need a different way of organising the system into modules [4]. Even given a specific organization (or *architecture*) of the system, different types of applications may require different solutions for each of the specific tasks involved in the generation process. In this context, a Multi-Agent System (MAS) implementation [5] may be highly benefitial. Each agent may be assigned a specific task to solve, and agents may negotiate between to reach a final solution, with no need to define an explicit architecture beyond the society of agents. Defining the architecture of the system would be limited to establishing the communication interfaces between the agents.

To put this idea to the test, two subtasks of the process of automatic generation of language texts have been selected: referring expresion generation and aggregation.

2 Referring Expression Generation and Aggregation

The appropriate use of referring expresions to compete with human-generated texts involves a certain difficulty. Simple algorithms to decide when to use a pronoun or a full noun produce very poor results. According to Reiter and Dale [6], a referring expresion must communicate enough information to identify univocally the intendent referent within the context of the current discourse, but

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 103–112, 2006.

always avoiding unnecessary or redundant modifiers. Reiter and Dale propose an algorithm for generating noun phrases to identify objects in the sphere of attention of the reader. Kibble and Power [7] propose a system for planning coherent texts and choosing the referring expresions. They claim that textual and syntactic planning mut be partially directed by the goal of maintaining referential continuity, increasing the opportunities for the unambiguous use of pronouns. Palomar et al. [8] also present an algorithm to identify noun phrases which are intended referents for personal, demostrative and reflexive pronouns or which have been ommitted in Spanish texts. They define a list of restrictions and preferences for the different types of pronominal expresions and they stress the importance of each type of knowledge - lexical, morphological, syntactic, and statistical - in the resolutoin fo anaphoric references.

Aggregation is the task of deciding how to compact the representation of the information in a given text. However, there is no accepted definition in the literature of what it is [9]. This task operates at several different linguistic levels, but in this paper we are only considering its application to concepts and their attributes. For instance, the system must decide whether to generate *"The princess is blonde. She sleeps."* or to generate *"The blonde princess sleeps."*. One should take care not to produce texts with too many adjectives when the information to be processed is dense in terms of attributes, as in *"The pretty blonde princess lived in a strong fancy castle with her stern rich parents."* Aggregation is generally desireable, but a balance must be found between conciseness and an acceptable style.

3 Open Agent Architecture

The Open Agent Architecture (OAA) [10] is a framework for developing multi-agent systems intended to enable more flexible interactions among a dynamic community of heterogeneous software agents. The operation of the architecture is based on the idea of delegation: agents do not hard-code their interactions (method calls or messages) in a way that fixed how and whom they will interact with, instead the interactions between OAA agents are expressed in terms of needs delegated to a Facilitator agent. This Facilitator agent coordinates the agent community so that it can achieve its task. It does this by providing services such as parallelism, failure handling, and conflict detection, which relieves each client agent from having to worry about these issues itself. OAA's Distributed Agents are simply programs - or wrappers around programs - that share several common functionalities, and which are possibly distributed across various machines.

In OAA, control of how interaction and communication occurs among agents arises from cooperation between 4 distinct knowledge sources:

- the requester which specifies a goal to the Facilitator,
- providers who know what services they can provide and register their capabilities with the Facilitator,

– the Facilitator which receives requests from requesters, maintains a list of available provider agents, and has a set of general strategies for meeting goals, and
– meta-agents that contain domain- or goal-specific knowledge and strategies which are used as an aid by the Facilitator.

These knowledge sourcers interact in the following way to make cooperation possible among a set of OAA agents. The Facilitator matches a request to an agent or agents providing that service, delegates the task to them, coordinates their efforts, and delivers the results to the requester. This manner of cooperation among agents is suitable both for straightforward and compound, multi-step tasks. In addition to delegation, OAA also provides the ability to make direct calls to a specific agent, and to broadcast requests.

The OAA's Interagent Communication Language (ICL) is the interface and communication language shared by all agents, no matter what machine they are running on or what computer language they are programmed in. The ICL has been designed as an extension of the Prolog programming language, to take advantage of unification and other features of Prolog. A number of key declarations and other program elements are specified using ICL expressions. These include declarations of capabilities, events (communications between agents), requests for services, responses to requests, trigger specifications, and shared data elements.

4 Multiagent Module for Referring Expression Generation and Aggregation

The work presented here is a multi-agent module for the tasks of Referring Expression Generation and Aggregation, implemented within the OAA architecture. Each agent is responsible for a specific subtask, and the conjunction of all the subtasks gives rise to a full implementation of the required tasks. This module differs from a blackboard architecture in the ability to negotiate that OAA agents have. When an agent requests an action from the rest of the system, the system may gather the solutions offered by different agents and use the one that best fits its purpose, or a combination of various solutions. The agents in OAA negotiate between them, rather than simply sharing data.

The multiagent system is made up of five specific agents and one auxiliary agent, coordinated by the Facilitator provided by the OAA architecture. A brief description of these agents and their functionality follows:

– **FileAgent.** Auxiliary agent capable of reading and writing text files.
– **ReferringAgent.** Main agent for the the two tasks addressed by the module. It deals with interpreting the input data and ensuring they are available for the rest of the agents.
– **PronounAgent.** Agent in charge of deciding if the reference to a given concept is made using its full name or only a pronoun.
– **KnowledgeAgent.** Agent that enriches some of the concepts with the attributes associated to them in the knowledge base.

- **AggregateAgent.** Agent that aggregates some of the descriptions of concepts in the text - in term of attributes - with the occurrences of the concepts in the text.
- **DisaggregateAgent.** Agent that performs the opposite task: it separates a description of a concept in terms of its attributes from a given occurrence of the concept in the text.

4.1 Data Representation

The notation described here corresponds to the internal representation within the multiagent system. Messages to be communicated in the final text are organised into paragraphs, and they are represented in the ICL language of the OAA architecture.

Characters, locations, and attributes are represented as simple facts that contain an identifier - which distinguishes each character or location from the others - and its name.

```
character(ch26,princess)
location(l14,castle)
attribute(ch26,blonde)
```

The identifier occurring in attribute facts corresponds to the identifier of the concept that has that attribute, and the name corresponds to the attribute itself.

The current prototype operates over simple linguistic constructions: the description of a concept using an attribute or the relation between two concepts.

```
[character(ch26,princess),
    isa(),
    attribute(ch26,pretty)]

[character(ch26,princess),
    relation(ch26,l14,live),
    location(l14,castle)]
```

Pronominal references are indicated by adding a 'pron' element to the corresponding fact, as in

```
character(ch26,princess,pron)
```

Finally, concepts may be accompanied by attributes that will precede the name of the concept, as in "the pretty blonde princess". This list of attributes is added to the corresponding concept.

```
character(ch26,princess,
          [attribute(ch26,pretty),
          attribute(ch26,blonde)])
```

4.2 Specific Funcionality of the Agents

Each one of the agents that make up the module is responsible for a very specific task. Thanks to the flexibility of multi-agent systems, not all agents will be required every time to act on a text draft. Each agent carries out different modifications over the text draft, and only some of them may be desireable at a given stage.

ReferringAgent. The ReferringAgent interprets the input data obtained from a file and makes them available for the rest of the agents to work on them. It sends a request to the agent system for reading from a text file the input to the referring expresion generation module. The agent reads this string, transforms it into the internal representation of the agent system, and makes it public to the rest of the agents by specifying to the Facilitator that it has those data and that any other agent may read them or modify them. Another task available from this agent is that of saving the processed information to file. An example of input text is:

[character(ch26,princess);relation(ch26,l14,live);location(l14,castle)]
[character(ch26,princess);isa();attribute(ch26,pretty)]

For this text, the following facts are generated as messages:

```
message(0,0,[character(ch26,princess),
             relation(ch26,l14,live),
             location(l14,castle)])
message(0,1,[character(ch26,princess),
             isa(),
             attribute(ch26,pretty)])
```

Messages are numbered indicating to which paragraph they belong, and their position with respect to other messages within that paragraph. The rest of the agents will eliminate or add messages, and it is necessary to retain the order between them for the generated text to keep the desired structure.

PronounAgent. The Pronoun Agent decides whether the reference to a given concept is carried out using its full name, for instance *"the princess"*, or the corresponding pronoun, in this case, *"she"*. The algorithm employed by the agent is based on two ideas. When writing a text one cannot use a pronoun to refer to something that has not been mentioned before, or readers will be confused. An example might be:

She lived in a castle. A princess was the daughter of parents.

Also, if the full name reference and the pronominal reference are too far apart, the reader will be confused and he will be unable to relate the two occurrences of the same concept. An example is given in the following text:

A princess lived in a castle. She was the daughter of parents. She loved a knight. She was pretty. She was blonde. They lived in it.

The heuristic used by the agent relies on using a pronoun if the concept in question is one of the last two concepts mentioned by their full name. A possible result would be:

```
message(0,0,[character(ch26,princess),
             relation(ch26,l14,live),
             location(l14,castle)])
message(0,1,[character(ch26,princess,pron),
             relation(ch26,ch25,love),
             character(ch25,knight)])
message(0,2,[character(ch26,princess),
             isa(),
             attribute(ch26,pretty)])
```

KnowledgeAgent. The Knowledge Agent enriches some of the concepts using the attributes that they have associated in the knowledge base. This agent has an associated probability value, and it decides whether to enrich a concept or not based on that probability. This value must be chosen with care. If it is too small, the text will have very few adjectives, and if it is too high the text will have redundant mentions of certain attributes.

An example of the operation of this agent is given below. In this case, attributes have been added to the reference to the castle, but not to the reference to the princess.

```
message(0,0,[character(ch26,princess),
             relation(ch26,l14,live),
             location(l14,castle,[attribute(strong)])])
```

AggregateAgent. The Aggregate Agent aggregates some of the descriptions in the text to other occurrences of the concepts that they correpond to. In order to achieve this, it searches in each paragraph for messages of the type "X is Y", and it aggregates the attribute appearing in the message with an earlier occurrence of the given concept in the same paragraph. This is done in some cases and not in others according to a probability associated with the agent. If the aggregation is performed, the message containing the description is eliminated from the paragraph.

An example of the results that this agent may generate from the initial messages read is given below. The probability used in this case is 0.5.

```
message(0,0,[character(ch26,princess,[attribute(pretty)]),
             relation(ch26,l14,live),
             location(l14,castle)])
```

DisaggregateAgent The Disaggregate Agent carries out a task complementary to that of the `AggregateAgent`. It disaggregates some of the attributes of a concept and it inserts into the text a separate message describing them of the type "X is Y". In each paragraph it checks all the concepts that have an associated list of attributes, and - again according to a predetermined probability - it decides to disaggregate some of them. If it decides to disaggregate a given attribute, it eliminates it from the list of attributes associated with the concept and it adds a descriptive message following the message where the concept occurred.

An example of the operation of this agent - with a probability of 0.5 - and starting from the example presented in the description on the `KnowledgeAgent`, would be:

```
message(0,0,[character(ch26,princess),
             relation(ch26,l14,live),
             location(l14,castle)])
message(0,1,[location(l14,castle),
             isa(),
             attribute(l14,strong)])
```

4.3 Experiments and Results

The OAA architecture provides mechanisms for monitoring and refining the implemented agents. Each agent can be manually switched on or off. These mechanisms have been used to perform some experiments over the implemented multi-agent system.

Although the OAA architecture is intended to make the agents independent from one another, there are some restrictions that must be taken into account. The first agent that should be called is the `ReferringAgent`, for without the input data the rest of the agents can do nothing. However, as this agent sends a request to the system for reading or writing a file, the `FileAgent` must have been called beforehand, so that it can reply to the requests presented by the `ReferringAgent`.

The other four agents of the system can be called in any order, and even called repeatedly. Given the probabilistic factors involved, and the fact that messages are added or eliminated in some cases, the result of system operation will vary depending on the order in which the agents are called. A clear example is the possible interaction between the `DisaggregateAgent` and the `KnowledgeAgent`. If the first one is called before the second one, very few disgregations will be performed on the text, since the concepts initially have no associated list of attributes. On the other hand, it the `KnowledgeAgent` has already been called, there will be more concepts with associated attributes, and the action of the `DisaggregateAgent` will be more visible.

Another issue to be taken into account is that of pronouns. Each of the agents operates over the concepts without considering whether they will be finally realised as pronominal references, so in theory messages such as *"Pretty she loved a*

knight" are possible. Since this sort of sentence is ungrammatical, in such situations the system will ignore any attributes associated with the concept described by a pronoun. This gives rise to correct sentences such as *"She loved a knight"*.

5 Discussion

The results obtained from the operation of the multi-agent system have been compared with those of an existing application, ProtoPropp [11], and it evolutionary version EvoProtoPropp [12]. ProtoPropp is a system for the automatic generation of stories that reuses a case base of previous stories to produce a new one that matches the user's requests. The system operates over a knowledge base organised as a taxonomy. This knowledge base or ontology includes the characteristics of the concepts and the relations that exist between them. The data structure that the system outputs is plot plan, in which a skeleton of the plot is described in terms of the elements of the ontology that the system handles - characters of the story, locations for the action, events that take place... From this, the textual representation of the story is obtained.

In the generation module of ProtoPropp the referring expresions to be used for each concept are determined using a very simple heuristic: the first time that a concept appears in a paragraph, the generator uses its full name, and for all subsequent occurrences in that paragraph it uses a pronoun. The problem with this method is that two appearances of the same concept may be quite far apart within the same paragraph, so the reader is confused when reading the text. The generation module of ProtoPropp does not aggregate concepts and attributes. A fragment of text generated by ProtoPropp is the following:

> *A princess lived in a castle. She loved a knight. She was pretty. She was blonde. It had towers. It was strong.*

EvoProtoPropp is an implementation of ProtoPropp in which the tasks of referring expresion generation and aggregation are carried out using Evolutionary Algorithms. The same fragment given above, if generated by EvoProtoPropp would come out as:

> *A pretty princess lived in a strong castle. She loved a brave knight. The princess was blonde. The castle had towers.*

Using the multi-agent system described in this paper, that same fragment of text is generated as:

> *A princess lived in a strong castle. She loved a brave knight. The princess was pretty. She was blonde. The castle had towers.*

Both EvoProtoPropp and the multi-agent system improve the results of ProtoPropp, resulting in a better use of adjectives and more referential coherence of the final text.

A possible further improvement would be to introduce an additional agent responsible for checking that the text fulfills a set of restrictions before accepting it as valid. The current implementation may result in redundant uses of attributes, like in *"The pretty princess was pretty"*. Problems of coherence may also arise in cases where two occurrences of the same concept appear associated with different attributes, as in *"The pretty princess lived in a castle. The blonde princess loved a knight"*. This situation might erroneously suggest that there are two different princesses in the generated story, even if the underlying data imply a pretty blonde princess.

Another issues to be tested is the concurrent operation of the agents. The current implementation relies on the agents being called in any order, but no two at the same time. The OAA architecture allows the definition of triggers over the data, so that an agent may learn when data over which it has already worked are modified by another agent. This could lead that agent to revise the data in order to apply its functionality again.

6 Conclusions and Future Work

The first experiments concerning the optimization of Referring Expresion Generation and Aggregation tasks have given promising results. Each agent is responsible for very specific issues within these tasks. This results in a very flexible solution, so that users may call all or any of these tasks separately with great ease.

The positive results of this experiment opens the doors to considering the application of similar solutions to other aspects of NLG, such as the organisation of a generation system in modules, or the definition of its control flow. There are many ways of organising a natural language generation system, and their advantages and disadvantages are still subject to discusion [4,13]. One can find singnificantly different architectures according to their division into modules or the topology of the connections between them. In [4] several of these architectures are discussed in detail, and all of them show advantages and disadvantages. The developer of an NLG application must consider a wide range of architectural solutions, for each one of them may be relevant for some particular aspect of his problem.

The cFROGS architecture [14] aims to provide a possible NLG application developer with the necessary infrastructure to facilitate his task as much as possible. This is achieved by providing generic architectural configurations for the most commonly used configurations used in this type of system. cFROGS identifies three basic design decisions when building a generation system: the set of modules to use, the flow of control that handles those modules, and the data structures that pass from one module to another.

The idea of a multi-agent system may be adapted to an architecture such as cFROGS from two different points of view. On one hand, the agents may be used as wrappers for modules of an NLG system implemented in cFROGS, so that the flow of control that governs them is the OAA architecture itself. The NLG systems implemented according to this structure would have similarities

with a blackboard architecture. On the other hand, a single module within a cFROGS architecture might be a wrapper for an OAA architecture, responsible for starting both the Facilitator and the rest of the agents as necessary.

References

1. Reiter, E., Dale, R.: Building Natural Language Generation Systems. Cambridge University Press (2000)
2. Goldberg, E., Driedger, N., Kittredge, R.: Using natural-language processing to produce weather forecasts. IEEE Expert: Intelligent Systems and Their Applications **9** (1994) 45–53
3. Callaway, C., Lester, J.: Narrative prose generation. In: Proceedings of the 17th IJCAI, Seattle, WA (2001) 1241–1248
4. DeSmedt, K., Horacek, H., Zock, M.: Architectures for natural language generation: Problems and perspectives. In Ardoni, G., Zock, M., eds.: Trends in natural language generation: an artificial intelligence perspective. LNAI 1036. Springer Verlag (1995) 17–46
5. Ferber, J.: Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA (1999)
6. Reiter, E., Dale, R.: A fast algorithm for the generation of referring expressions. In: Proceedings of the 14th conference on Computational linguistics, Nantes, France (1992)
7. Kibble, R., Power, R.: An integrated framework for text planning and pronominalization. In: Proc. of the International Conference on Natural Language Generation (INLG), Israel (2000)
8. Palomar, M., Ferrández, A., Moreno, L., Martínez-Barco, P., Peral, J., Saiz-Noeda, M., Muñoz, R.: An algorithm for anaphora resolution in spanish text. Computational Linguistics **27** (2001) 545–567
9. Reape, M., Mellish, C.: Just what is aggregation anyway? In: Proceedings of the 7th EWNLG, Toulouse, France (1999)
10. Martin, D.L., Cheyer, A.J., Moran, D.B.: The open agent architecture: A framework for building distributed software systems. Applied Artificial Intelligence **13** (1999) 91–128 OAA.
11. Gervás, P., Díaz-Agudo, B., Peinado, F., Hervás, R.: Story plot generation based on CBR. In Macintosh, A., Ellis, R., Allen, T., eds.: 12th Conference on Applications and Innovations in Intelligent Systems, Cambridge, UK, Springer, WICS series (2004)
12. Hervás, R., Gervás, P.: Uso flexible de soluciones evolutivas para tareas de generación de lenguaje natural. Procesamiento de Lenguaje Natural **35** (2005) 187–194
13. Reiter, E.: Has a consensus NL generation architecture appeared, and is it psychologically plausible? In McDonald, D., Meteer, M., eds.: Proceedings of the 7th. IWNLG '94, Kennebunkport, Maine (1994) 163–170
14. García, C., Hervás, R., Gervás, P.: Una arquitectura software para el desarrollo de aplicaciones de generación de lenguaje natural. Procesamiento de Lenguaje Natural **33** (2004) 111–118

An Autonomous and User-Independent Hand Posture Recognition System for Vision-Based Interface Tasks

Elena Sánchez-Nielsen[1], Luis Antón-Canalís[2], and Cayetano Guerra-Artal[2]

[1] Dpto. E.I.O. y Computación. Universidad de La Laguna, Spain
enielsen@ull.es
[2] Instituto de Sistemas Inteligentes (IUSIANI). Las Palmas de G.C., Spain
{lanton, cguerra}@iusiani.ulpgc.es

Abstract. This paper presents a system for hand posture recognition that works with colour video streams under varying light conditions for human-machine interaction in vision-based interface tasks. No initialization of the system is required and no user dependence is involved. With this aim, we first model on-line each user's skin colour from the skin cue imaging of his/her face detected by means of Viola and Jones detector. Afterwards, a second order isomorphism approach performs tracking on skin colour blob based detected hand. Also, we propose this approach as a mechanism to estimate hand transition states. Finally, evidences about hand postures are recognized by shape matching, which is carried out through a holistic similarity measure focused on the Hausdorff distance. The paper includes experimental evaluations of the recognition system for 16 different hand postures in different video streams. The results show that the system can be suitable for real-time interfaces using general purpose hardware.

1 Introduction

Vision-based Interaction or Vision-based Interface (VBI) paradigm [1] aims to produce precise and real-time analysis that will be useful in a wide range of applications, from perceptual intelligence products [2] to virtual and augmented reality systems, or from the "Looking and People" domain [3] where computer vision technology is used to sense and perceive the user in an HCI context. In these circumstances, recognition of gestures is an important topic to solve in order to support visual aspects of interaction.

Hand gestures can be roughly classified in static hand postures and temporal hand gestures. Hand postures express concepts by hand configurations and hand shapes, while temporal hand gestures represent some actions by hand movement. Also, hand postures can act as special transition states in temporal gestures, and supply a cue to segment and recognize temporal hand gestures. Different approaches have been proposed for hand posture recognition due to the different posed difficulties such as autonomous initialization, user independency, varying light conditions, changeability of hand shape and real-time performance.

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 113–122, 2006.

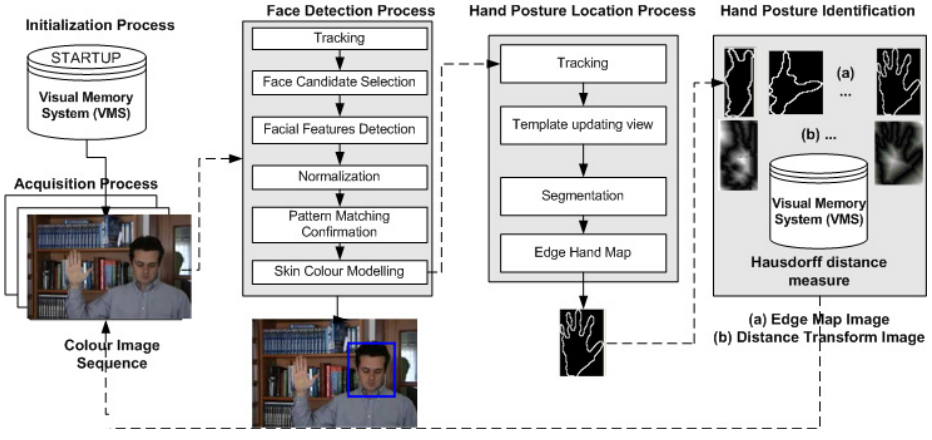


Fig. 1. System Processes Overview

Approaches proposed in [4] and [5] cover a no-real time elastic graph matching method for classifying hand postures against complex backgrounds with a 86.2% recognition rate and a particle filtering with hierarchical layered sampling for simultaneous tracking and recognition of hand states, which works with probabilistic prior knowledge on skin colour and requires initialization of the tracker process and manually segmentation of hands with the purpose of including skin colour information in the form of a probabilistic prior. Other approaches take advantage of appearance models of human hands using supervised and unsupervised learning paradigms [6] or probabilistic frameworks for view-based arm recognition using a Hidden Markov Model (HMM) [7].

In this paper, we propose a system that offers an integrated solution to: autonomous initialization, user independency, varying light conditions, changeability of hand shape and real-time computation for hand posture recognition. The proposed approach is focused on achieving continuous operation of three processes: (1) detection, (2) tracking and (3) matching of hand poses. Figure 1 illustrates the architecture of the automated system. In order to achieve an autonomous initialization of the system and user-independency, we propose detect the user face with the fastest and most accurate pattern detection method for faces in monocular images [8]. Once the first detected face is computed, an individual skin colour model is achieved on-line for detecting skin blob based hands using the skin cue imaging computed from the face detected. Next, we propose a tracking method based on a second order isomorphism that tracks hand blobs and, at the same time, we use this method in order to estimate hand transition states. As a result, hand posture is recognized only if a pose change is detected. This situation leads to computation of real-time performance using general purpose hardware. In order to classify the hand posture, we propose a shape comparison method that makes use of a holistic similarity measurement and a visual memory that contains all the different postures to be recognized.

The paper is organized as follows: Section 2 illustrates the hand posture detection process. In Section 3, the tracking process is described. Identification of hand posture is detailed in Section 4. Experimental results with diverse people under different environments and illumination conditions are presented in Section 5 and Section 6 concludes the paper.

2 Hand Posture Detection

Detection of hand poses is a complex problem because human hands are deformable articulated objects with many degrees of freedom. Therefore, we propose a hand segmentation approach based on human skin colour modeling. However, the use of skin colour modeling involves different issues: (1) it is no user-independent and (2) it is no environment-independent. Therefore, we propose an autonomous and independent skin colour model for each user that interacts with the system. With this aim, we model each user's skin colour cue using a red and green normalized colour space [9] from the skin imaging of his/her face detected by means of interesting results achieved by recent face detectors [8], [10]. To be precise, the face detector framework used, which is illustrated in Figure 1, is focused on the idea of a boosted cascade classifier. It is capable of processing images extremely rapidly and achieving high detection rates and low false positive rates. Once the first face detected is computed and the individual skin colour is modeled, all the pixels that best fit in the modeled colour space (see Figure 2.c) are located through a saliency map that shows the different skin areas. This image is combined with the difference image (see Figure 2.b), that is computed between the original current image (see Figure 2.a) and the average background image. Applying the restrictions of a single user with a single hand up, the resulting image contains two major blobs (see Figure 2.d). One of those blobs matches the detected face and the second one defines the hand pose. The first hand detected is taken as the template to track in the next frames.

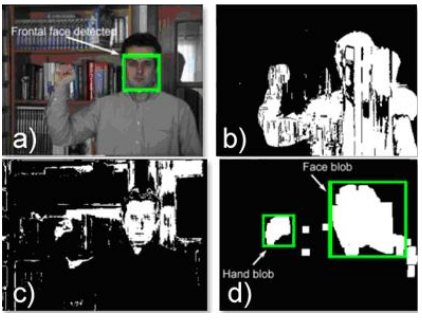


Fig. 2. Skin detection: a) input image; b) background subtraction; c) skin areas; d) skin blobs

3 Hand Posture Tracking

A template matching approach with updating process is introduced in order to perform robust hand shape tracking and to detect hand states transition. It is focused on the framework of representation spaces based on second order isomorphism [11] between physical objects from the real world (distal space) that are denoted as visual objects and their internal representations in a proximal space I. In a variety of the proximal space are located all the real views of an object that evolves before a camera [12]. The following principle [11] is established by the use of a second order isomorphism: *if the similarity between two visual objects A and B is greater than between visual objects B and C, the distance between their respective representations (A', B' and C') should verify that* $d(A', B') < d(B', C')$. This establishment leads to the definition of the continuity principle [12]: *small transformations in the visual object perception (deformations, rotations, warpings, etc.) correspond to small displacements of the localization coordinates of that object in the proximal representation space* I. As a result, the visual evolution of the perceptual object describes a warped visual transformation curve in the proximal space I (see Figure 3). The different points of the visual curve in the proximal space I correspond to the distinct object views in the distal space. Like in other geometric spaces, a distance measure can be established in I. We use the L_2 norm as the distance measure between two different points.

In this context, *the visual object* (hand posture) is represented by a template U of $m \times n$ pixels and the representation space I is conformed by a subspace of $R^{m \times n}$, where U is an element (point) of I that corresponds to the visual object. To be precise, the space I includes all the possible views that can be formed in an image of $m \times n$ pixels. The values of every pixel v_i (being i between 1 and $m \times n$) correspond to the projection of U in every one of the $m \times n$ coordinates. Diverse templates that belong to the different comparison areas between the pattern U to match of size $m \times n$ and the current image of size $M' \times N'$ ($m < M'$ and $n < N'$) are projected for each frame that composes an image sequence which includes a hand shape to be located. The number of the N points generated

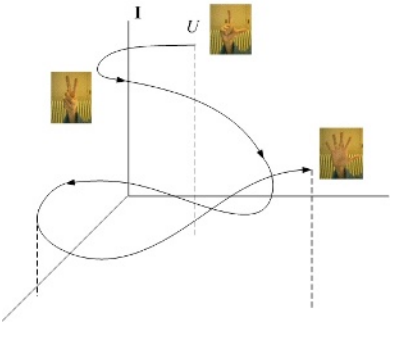


Fig. 3. Object warped visual transformation curve: it describes the visual evolution of the tracked object (hand) in the proximal space I. Each arrow corresponds with a different view of the object during their evolution in the distal space.

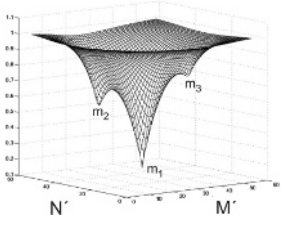


Fig. 4. Similarity values computed between the current template pattern and the current image patch: (a) m_1 (absolute minimum) corresponds to the patch image most similar to the current view of the hand posture; (b) m_2 and m_3 (local minima) correspond to similar objects related to the hand pose

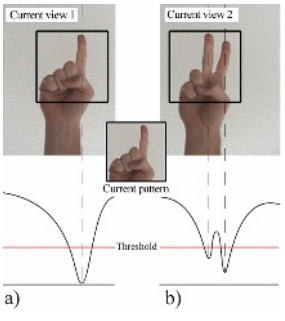


Fig. 5. Detecting hand state transition: (a) Similarity value between current view 1 and current pattern that computes an only absolute minimum; (b) Similarity value between current view 2 and current pattern, a new local minimum is computed due to the transition between current view 1 and current view 2

from the space I is equivalent to $(M' - m + 1) \times (N' - n + 1)$. Among them is the proximal representation point that corresponds to the m x n area of the image most similar to the current view of the object of interest. This process can be represented by a bidimensional discreet function such is illustrated in Figure 4. The absolute minimum (represented by m_1) corresponds to the image most similar to the current view of the hand posture and the other local minima (represented by m_2 and m_3) correspond to similar objects related to the hand pose. We call them *context objects* [12].

The updating template process is based on the closeness condition of the *context objects*. That is, the template must be updated before the appearance of any *context object* is more similar to the reference template than to the current view of the target object. With this purpose, an updating threshold must be set up taking into account the closeness of the objects of context. Therefore, when a new view of the object of interest (hand pose) is taken as the current template, a new updating threshold is also computed. The value assigned to this threshold is obtained dividing by two the distance between the new template and the closest context object. This way, the view of the target object is taken as the

current template when the absolute minimum value computed is smaller than the established threshold. At the same time, this threshold leads to establish when a new state transition of hand pose has taken place focused on the notion that the different hand shapes generate local minima due to its auto-similarity. As a result, we can determinate that a new hand pose has taken place when the second minimum value is smaller than the established threshold. Figure 5 illustrates the process: (i) Figure 5.a shows how the hand pose generates a minimum value when the current view (view 1) is compared with a very similar current pattern and (ii) Figure 5.b shows that the second finger of the next current view (view 2) introduces a second minimum that lies below the established threshold. This situation leads to a threshold and template updating and consequently, a new hand pose U_j to recognize.

4 Hand Pose Identification

Once a new user hand pose U_j has been detected, a shape matching approach is computed using the Hausdorff distance similarity measure. Each hand pose is represented by its edge map image, which is computed using the Canny edge approach. For two sets of points A and B, the Hausdorff distance [13] is defined as:

$$H(A,B) = \max(h(A,B), h(B,A)) \tag{1}$$

Where

$$h(A,B) = \max_{a \in A} \min_{b \in B} \|a - b\| \tag{2}$$

The function $h(A,B)$ is called the *directed Hausdorff* distance from set A to B, where $\|.\|$ is any norm. It ranks each point of A based on its distance to the nearest point in B and uses the largest ranked point as the measure of distance. In order to avoid erroneous results due to occlusions or noise conditions, the directed distance can be extended to find the *partial directed distance* between sets A and B by replacing the maximum value by a k^{th} quartile value. It is defined as

$$h_k(A,B) = \underset{a \in A}{K^{\,th}} \min_{b \in B} \|a - b\| \tag{3}$$

The *Partial Bidirectional Hausdorff* distance is then formulated as:

$$H_{kl}(A, t(B)) = \max(h_k(A, t(B)), h_l(t(B), A)) \tag{4}$$

where $t(.)$ denotes a transformation of a set T.

We use a visual memory system that is called *VMS* with the purpose of storing the different postures that compose the hand gesture vocabulary communication for human computer interaction. This memory is created in a start-up stage. New gestures can be added to the visual system whenever the user wants to. In order to address diverse visual aspects for each stored hand pattern M and non-local

distortions, pattern postures are represented by q different samples in the visual memory of the system (VMS):

$$VMS = \{P_{mq} : m = 1, 2, \cdots, M; q = 1, 2, \cdots, Q\} \tag{5}$$

Where each q^{th} sample of each mth pattern hand posture is defined by its ith binary edge map and its ith distance transform [14], which generates for each edge map posture image an image where background pixels have been labelled with their Euclidean distance to the closest object pixel.

In the same way, every new J segmented user hand posture U_j to recognize is represented by its binary edge map and its distance transform.

Focused on the Hausdorff similarity measure and the visual memory notion, the identification of a specific hand gesture leads to the computing of the minimum *partial bidirectional Hausdorff* distance (4) between U_j and P_{Mq} using the set of transformations T, which is based on generating representations of different size in order to take into account the different changes in appearance of user hand postures. P_{Mq} denotes each one of the M stored patterns edge map in VMS. Two rescale operations are computed for each posture to recognize. Firstly, U_j is transformed to the size of the q^{th} sample, which represents a stored pattern P_{mq} in VMS and secondly, the specific pattern in VMS, P_{mq} is transformed to the size of U_j.

Figure 6 shows the block diagram for recognizing one hand posture, U_j. The algorithm is repeated for every stored pattern that composes the VMS. Finally, the posture is identified as the pattern for which the minimum *bidirectional partial Hausdorff* distance is given. In order to address the issue of rejecting gestures which are not in VMS, a decision rule is used:

$$output = \left\{ \begin{matrix} P_{m*q}, \text{if } H_{kl}(h_k(t(U_j), P_{m*q}), h_l(P_{m*q}, t(U_j)) < \delta) \\ otherwise, reject \end{matrix} \right\} \tag{6}$$

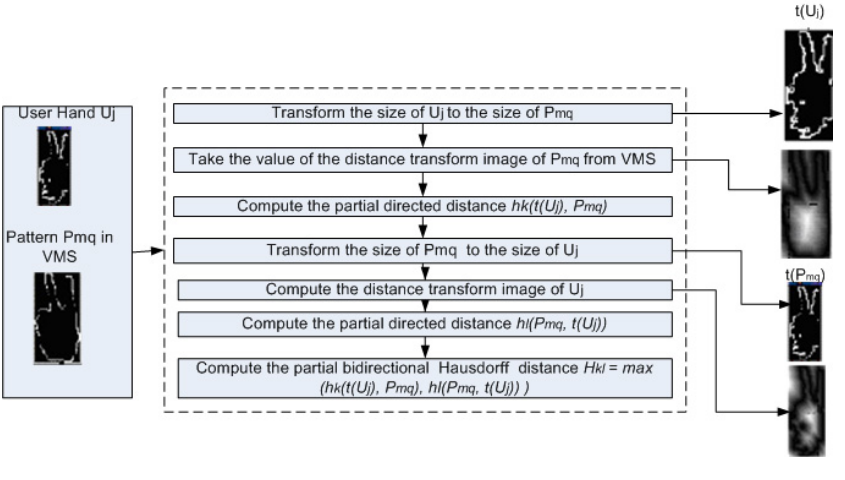


Fig. 6. Hand posture recognition algorithm

Where P_{m*q} is the selected pattern. This way, the pattern P_{m*q} is identified if the distance measure H_{kl} is below a user-supplied threshold δ, which is set empirically at 2.

5 Experimental Results

In order to carry out empirical evaluations of the system, different video streams which are labelled S1 - S5 were acquired at 25 fps on different days without special illumination restrictions and for diverse people. Each sequence contains frames of 320 × 240 pixels. Figure 7 illustrates a frame example from sequence S1 to S4. The gesture vocabulary used, shown in Figure 8, is composed by 16 postures, where some of them are really similar. Right now, only right hand postures have been used in our experiments; nevertheless, postures with the left hand can also be recognized. For each hand posture, four different samples are acquired with 36×53 average sizes. The template size used for the tracking process is established to 24×24 pixels. Table 1 illustrates the experimental results of face detection, recognition and tracking of hand postures from S1 to S5, using a P-IV 2.2 GHz. It must be taken into account that no person had received prior training of the postures to be performed, with the exception of the first person that characterizes sequence 1. Also, the sequences of different postures were carried out in a consecutive and temporal way in order to study the performance of tracking approach for detecting transition states of hand poses.

The results reported in Table 1 show the **total number of frames** of each sequence, the **Face detection rate**, the **Hand recognition rate**, the number of frames that the tracking process considers a new hand posture to process (named **Correct frames** in Table 1), where **Frames to process** (in Table 1) is the correct number of frames that the system must process (this value has been *a priori* computed, evaluating manually each sequence), and the number

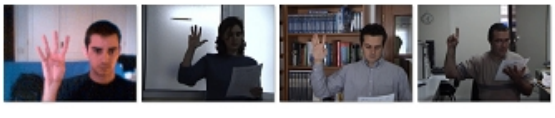


Fig. 7. Individual sample frames from sequence S1 to S4 with different light conditions

Fig. 8. Gesture vocabulary

Table 1. Performance results from sequence S1 to S5

Sequence	Total Frames	Face rate detection	Hand rate recognition	Frames to process	Correct frames	Identifiable digits
S1	2052	99.9%	94%	70	66	16
S2	1121	76.7%	85%	24	14	10
S3	1062	98.1%	87.5%	34	27	15
S4	1643	98.8%	87%	29	20	11
S5	1891	96.7%	87.5%	20	17	14

of different digits (**Identifiable digits**) that can be recognized during the sequence, where the maximum value is 16, corresponding to the different gestures to recognize. The achieved processing rate is 18 fps (this result includes the recognition process in each frame of the sequence), where the average time to process face, tracking and recognition processes are: 14.6ms, 0.7 ms and 40 ms respectively.

Two major conclusions have been obtained from the experimental evaluations: (i) if gestures are acceptably performed such as sequence S1, each one of the new hand poses will be correctly supplied to the recognition process and the hand gesture will be properly classified. (ii) If gestures are incorrectly performed, the recognition process can be confused by other similar gestures presented in the pose vocabulary, as for example gesture pairs 0-15, 8-9, 10-11, 12-13, and 13-14. It must be pointed out that in these experiments the hand gesture recognition has been performed for each frame of the sequence and this restriction is not necessary to be established if hand gestures are acceptably carried out. The hand gesture recognition needs 40 msecs. per frame, but for example for sequence S1, where the postures were acceptably performed, the tracker identifies as interesting patterns to recognize just 70 from the original 2052. Therefore the system load can notably be reduced, becoming clearly real-time.

6 Conclusions

An autonomous and user-independent system for temporal hand posture recognition has been presented. This system makes use of face detection to provide an autonomous mechanism of adjustment and personalization of the skin colour model in order to have an individualized cue for detecting the hand template to track of each user. The tracking module, which is based on a matching and updating process, is able to identify hand pose changes, circumstance that is used to reduce the system load. Also, this process does not need manual initialization for detecting the first hand to be tracked. Experimental results on different video streams demonstrate the robust and real-time performance of the proposed approach with diverse environments, hand shapes and illumination conditions. The recognition rates achieved according to the vocabulary complexity are certainly promising, considering that individuals have not received any prior training to perform the different poses.

References

1. Matthew Turk. "Computer Vision in the Interface". In Communications of the ACM, 47(1):61-67, January 2004.
2. A. P. Pentland. "Perceptual Intelligence". In Communications of the ACM, 43(3), pp. 35-44, 2000.
3. D. M. Gavrila. The Visual Analysis of Human Movement: A Survey. In Computer Vision and Image Understanding, Academic Press, 73(1):82-89, 1999.
4. Jochen Triesch and C. von der Malsburg.: A system for Person-Independent Hand Posture Recognition against Complex Bacgrounds. IEEE Transactions on Pattern Analysis and Machine Intelligence, 23(12):1449-1453. December 2001.
5. Lars Bretzner, Ivan Laptev and Tony Lindeberg.: Hand Gesture Recognition using Multi-Scale Colour Features, Hierarchical Models and Particle Filtering. 5th IEEE International Conference on Automatic Face and Gesture Recognition, pp. 423-428, Washington, DC, USA, May 21-22, 2002.
6. Ying Wu, Thomas S. Huang.: View-independent Recognition of Hand Postures. In Proc. IEEE Conf. on Computer Vision and Pattern Recognition, Vol II, pp. 88-94, Hilton Head Island, SC. June 2000.
7. A. Elgammal, V. Shet, Y. Yacoob, L. S. Davis. "Gesture Recognition using a Probabilistic Framework for Pose Matching". 7th International Conference on Control, Automation, Robotics and Vision, ICARCV 2002, Singapore, 2002.
8. Paul Viola and Michael J. Jones. "Rapid object detection using a boosted cascade of simple features". In Computer Vision and Pattern Recognition, 2001.
9. Christopher Wren, Ali Azarrbayejani, Trevor Darrell and Alex Pentland: Pfinder. "Real-Time Tracking of the Human Body". In IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 19, n.7, 1997.
10. Henry Schneiderman and Takeo Kanade. "A statistical method for 3d object detection applied to faces and cars". In IEEE Conference on Computer Vision and Pattern Recognition, 2000.
11. S. Edelman.: Representation and Recognition in Vision, The MIT Press, 1999.
12. C. Guerra, M. Hernndez, A. Domnguez, D. Hernndez. "A New Approach to the Template Update Problem". Lecture Notes in Computer Science, Volume 3522/2005.
13. W. Rucklidge. "Efficient computation of the minimum Hausdorff distance for visual recognition". Lecture Notes in Computer Science, 1173, Springer, 1996.
14. D. W Paglieroni. "Distance Transforms: properties and machine vision applications". In GVGIP:Graphical Models and Image Processing, 54(1):56-74, 1992.

An Effective Robotic Model of Action Selection

Fernando M. Montes González, Antonio Marín Hernández,
and Homero Ríos Figueroa

Facultad de Física e Inteligencia Artificial
Universidad Veracruzana
Sebastián Camacho No. 5, Centro
Xalapa, Veracruz, México
{fmontes, anmarin, hrios}@uv.mx

Abstract. In this paper we present a concise analysis of the requirements for effective action selection, and a centralized action selection model that fulfills most of these requirements. In this model, action selection occurs by combining sensory information from the non-homogenous sensors of an off-the-shelf robot with the feedback from competing behavioral modules. In order to successfully clean an arena, the animal robot (*animat*) has to present a coherent overall behavior pattern for both appropriate selection and termination of a selected behavior type. In the same way, an animat set in a chasing task has to present opportunist action selection to locate the nearest target. In consequence, both an appropriate switching of behavior patterns and a coherent overall behavior pattern are necessary for effective action selection.

1 Introduction

Action selection is rooted in ethology, where it is often described as the problem of 'behavior switching' or 'decision making'. However, it is in the field of robotics where this issue has recently become an important topic, since effective action selection is a requirement for successful robotic behavior. Furthermore, the design of animats that solve tasks in the same way as real animals; and the setup of real tasks, for the animat to solve, offer both limitations from the environment and the opportunity to improve selection. Therefore, the resolution of tasks related to foraging and prey-catching are a good choice for testing the relevance of an action selection mechanism. However, as Tyrrell [1] has pointed out, it is difficult to find purely algorithmic good implementations of action selection mechanisms. Even more difficult is it to find implementations that follow a biological approach for action selection besides offering good general solutions. In this paper we have employed a model that produces action selection by exploiting the centralized 'feature extractor' properties of a more biological model of robot action selection [2]. For our experiments we have controlled a commercial robot using a centralized model of action selection. This central model presents an effective selection in several tasks such as foraging and prey catching (Figure 1).

The organization of the paper is the following. In section 2 we describe the problem of action selection. Section 3 explains the requirements for an effective

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 123 – 132, 2006.

action selection. Next, in section 4 we present some recurring themes on computational models of action selection that can be identified in our model. Section 5 introduces the centralized model of action selection. In section 6, we further develop the foraging and chasing experiments reported in [3, 4]. Finally, we conclude that the central model of action selection fulfills most of the requirements for effective selection.

2 Action Selection

The task of deciding 'what to do next' can be summarized as the action selection problem. More specifically, the action selection problem is the task of deciding what to do next from a pool of possible actions in order to satisfy the goals or needs of the organism. This problem is not only concerned with the achievement of specific individual goals (e. g. finding food), but also the achievement of different overall goals (e. g. maintaining energy, avoiding threats, reproduction, etc.). These goals usually contain several sub-goals to be executed at different times. For example, in the task of maintaining energy supplies some possible actions are: looking for food, consuming the food, resting (saving energy), etc.

Whenever several processes try to gain access to shared motor systems, the action selection problem arises. All creatures, whether biological or artificial, can be viewed as having a limited number of sensorimotor systems that can be directed towards achieving certain goals. Ethology has produced several models for explaining how animals resolve the action selection problem. On the other hand, Artificial Intelligence has developed its own version of the problem in order to fill the gaps left by previous research on deliberative solutions. For people working in robotics, action selection consists of choosing the right action at the right time.

3 Effective Central Action Selection

The occurrence of action selection in the vertebrate brain highlights the importance of having localized selection. The work of Prescott et al. [5] offers clear evidence of the basal ganglia producing central action selection in the vertebrate brain. Furthermore, this claim is supported by the work of Snaith & Holland [6], who analyzed the cost of the connections using a distribution block as compared with reciprocal distributed inhibition. Reciprocal distributed inhibition requires $n(n-1)$ connections for n behavior types, plus further $2n$ connections with the addition of a new behavior type. In contrast, centrally distributed inhibition requires only $2n$ connections for n behavior patterns and even a new behavior type only requires two additional connections (Figure 2). A fully connected version may therefore be preferable when the number of connections is irrelevant; however, if we were to follow a close biological approach, then the size is important. Due to its lower cost of connectivity and the relative ease of adding new behavior patterns; central selection is a good answer to modular selection. Next, we analyze several views concerning *the requirements for effective action selection*.

Fig. 1. The four Khepera robots used in foraging and chasing behavior patterns. One Khepera was fitted with a camera and gripper, and the others with colored and shaped hats.

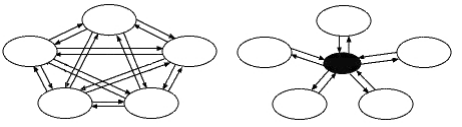


Fig. 2. Distributed reciprocal inhibition (left) compared with centrally distributed inhibition (right). Notice the difference in terms of economy of connection costs (Adapted from [5]).

Snaith & Holland (in the same paper) also provided for some key requirements that are: the execution of only one type of behavior at the time and the resolution of competition in favor of responding to the strongest stimulus. These authors also suggest that the switching mechanism should show persistence (hysteresis), lack of distortion, and clean switching. The mentioned requirements and the derived properties seem to be more oriented towards measuring how good the resolution of conflicts is, than the overall performance of the selection mechanism in generating appropriate behavior. In contrast, Tyrrell [1] has presented a set of requirements focused on obtaining a good benchmark criterion for measuring the capacity for observing performing selection. Werner [7] summarizes Tyrrell's criteria and adds some of his own; these criteria include the following requirements: deal with all sub-problems, exhibit persistence, allow sequences of behavior types, interrupt if necessary, exhibit opportunism, and have real-valued sensors. Redgrave et al., in summarizing some of the requirements for effective action selection, also highlighted the issue of the appropriate termination of behaviors. Basically, the action selection mechanism should terminate a behavior type when its expression has been successful or if it proves to be ineffective [8]. We have summarized some of the most important requirements proposed by these authors together with some of our own. Next, we shall present these requirements in order of relevance.

- *Coherent overall behavior* (dealing with all sub-problems). This is related to the ability to select the appropriate actions that lead to overcoming an overall goal. In order to allow correct sequencing of actions, the system must be able to remember the state in which a problem being solved is in, and must then be able to complete it. An action selection system must be able to show the simultaneous selection of compatible actions, but not the incompatible ones. Having real-valued sensors helps at the moment of integrating all the information coming from the different internal and external stimuli.
- *Adaptability and extensibility*. The system must be able to learn from experience. A system must be able to be easily extensible in design, and adding new actions to the given repertoire must be a straightforward operation. Addition of extra motor and sensory components must also be a straightforward operation to perform. Modularity is desirable in an action selection mechanism, in order for sensory and

motor mechanisms to be easily detached for debugging purposes. As far as possible, portability must be preserved in order to transport the mechanism to other platforms.

- *Appropriate selection* (selection proportional to current salience). The selection must be made by observing the needs of the system reflected in the current salience or urgency of competing actions. The system must respond not only to internal motivations, but also to relevant information reflecting the present state of the world.
- *Appropriate maintenance and termination of behavior* (hysteresis). Persistence is concerned with maintaining the present selection even when the initial value, that activated the action, has been decreased. This involves the capacity to ignore other stimuli, in favor of continuing to execute the currently selected type of behavior. While this is necessary in order to avoid excessive switching or dithering, it is not absolute; the system must be able to re-prioritize if an alternative activity is sufficiently salient. The system must be able to terminate an action, if it becomes clear that the action is no longer appropriate. An action must be terminated if it has been completed.
- *Effective switching*. When an activity has been selected, the performance of the action must be the same for different activation values whenever they are maintained above a given threshold. This is known as lack of distortion. Clean switching means that once a new selection takes over, the losing competitors are rapidly deselected and the selection of two incompatible behavior types at the same time never occurs.

4 Computational Models of Action Selection

Several known computational models of action selection exist in writing; however, so as not to extend the length of this paper we have focused on the work of Snaith & Holland [6]. The reader is referred to the direct sources of the works of Ludlow [9], Brooks [10] and Tyrrell [1] for a more complete description of their models. Therefore, we have identified a number of recurring themes in these models of action selection. One major theme concerns whether action selection is explicit or implicit. In other words, whether a control system should contain specialized components for action selection, or whether action selection is one of the functions that arise from the operation of components involved in wider aspects of motor control. The subsumption architecture, developed by Brooks, favors implicit action selection; while Tyrrell, based his on the proposal of Rosenblatt & Payton [11], and favors specialized action selection mechanisms. Prescott et al. [5] have argued that specialized action selection mechanisms may be preferable because of the flexibility that modularity can provide. In the next section, we present a modular base for centralized action selection that grants flexibility for modification purposes.

A second main theme relates to the advantages of resolving selection competitions by direct, mutual inhibition between modules (e.g. the Black Bean Aphid model of Ludlow), or by the use of a centralized mechanism such as McFarland's distribution block [12]. The greater number of connections in a mutual inhibition scheme may offer some advantages, though it is not clear what these might be. On the other hand,

mutual inhibition certainly carries with it a much higher overhead in terms of connection costs compared to a mechanism with a centralized switch. The avoidance of such overheads may be important for biological control systems because of their metabolic and space requirements. The centralized model clearly offers an example of a distribution block of action selection.

A third general theme is concerned with the choice between mechanisms that select the strongest drive or the most salient behavior (e.g. the mutual inhibition model of Ludlow), and those that seek compromise solutions that partly satisfy the goals of multiple drivers. Most ethological work (e.g. McFarland [13]) assumes that all else being equal, the most salient behavior is selected. In our work we employ the idea of a drive for the calculation of the urgency-to-be-executed of the competing behavior patterns. In this model, the use of a winner-takes-all selection allows only one type of behavior to be executed at a time.

A final theme is related to the way of selecting the final outcome. For instance, in an architectural model described by Rosenblatt and Thorpe [14], a centralized arbitration mechanism collects votes from all action-producing modules and performs command fusion to determine the most popular action. Humphrys [15] has evaluated a number of learning strategies to achieve the satisfaction of multiple conflicting heterogeneous goals using homogenous action-effectors. Although, it is not easy to identify all the requirements for effective selection in most computational models of action selection; the centralized model of action selection, which is presented below, covers many of these requirements.

5 The Centralized Action Selection Model

A centralized model of action selection with sensor fusion (CASSF) has been used to express a winning competing behavior type at a time. The development of CASSF was based on previous work related to developing a model of the robot basal ganglia [16, 17]. However, in this case we have decided to employ a reduced version of some of the nuclei in the robot basal ganglia model, which enables us to focus on the development of the tasks we intend to resolve. The setup of these tasks is explained in section 6, and below we introduce our model of central action selection. CASSF employs a main control loop that updates every step of the simulation sensor readings and motor commands. The different sensor readings from an off-the-shelf robot, a Khepera robot in our case: the infrared, the odometry, the RGB image from the CCD camera, and the optical barrier in the gripper, all form the raw sensory information that is to be fed into the model. Next, the raw sensory information takes the form of single perceptual variables that can be used to build a unified perception of the world. Therefore, the use of sensor fusion facilitates the integration of multiple non-homogenous sensors into a single perception of the environment.

The perceptual variables are used to calculate the urgency (salience) of a behavior type. However, not all the variables are equally relevant for a particular behavior type. For instance, the searching of a place for releasing a cylinder requires the presence of an object in the gripper. Additionally, behavior patterns contribute to the calculation of the salience with a busy-status signal indicating a critical stage where interruption should not occur. Therefore, the salience of a behavioral module is calculated by

weighting the relevance of the information from the environment, in the form of perceptual variables and its busy status. In turn, the behavior type with the highest salience wins the competition and can be expressed as motor commands that are directly sent to the motor wheels and gripper. Next, we shall explain how the salience is computed.

The salience is calculated by adding the multiplication of the perceptual variables, by the relevant behavioral weights, to the multiplication of the weighted busy-status. For example, in foraging behavior (Section 6.1) the perceptual variables, wall_detect(e_w), gripper_detect(e_g), cylinder_detect(e_c), and spinning(e_s); form the context vector, which is constructed as follows ($\mathbf{e} = [e_w , e_g , e_c, e_s]$, $e_w , e_g , e_c, e_s \in \{0,1\}$). Five different behavior patterns return a current busy status (c_i) indicating that ongoing activities should not be interrupted. Next, the current busy-status vector is formed as follows, $\mathbf{c} = [c_s , c_p , c_w , c_d, c_t]$, $c_s , c_p , c_w , c_d, c_t \in \{0,1\}$, for *cylinder-seek*, *cylinder-pickup*, *wall-seek*, *cylinder-deposit*, and *turning-around* respectively. The salience (**s**) or urgency is calculated by adding the weighted busy-status (w_b) to the weighted context vector (e). Then with $w_b = 0.7$ we have

$$
\begin{array}{lll}
\mathbf{s}_i = \mathbf{c}_i w_b + \mathbf{e}_i \mathbf{w}_i^e & for & \\
\text{cylinder} - \text{seek} & \mathbf{w}_s^e = [& 0.0, \quad -0.15 \quad -0.15, \quad 0.0 \quad], \\
\text{cylinder} - \text{pickup} & \mathbf{w}_p^e = [& 0.0, \quad -0.15, \quad 0.15, \quad 0.0 \quad], \\
\text{wall} - \text{seek} & \mathbf{w}_w^e = [& -0.15, \quad 0.15, \quad 0.0, \quad 0.0 \quad], \\
\text{cylinder} - \text{deposit} & \mathbf{w}_d^e = [& 0.15, \quad 0.15, \quad -0.15, \quad 0.0 \quad], \\
\text{turning - around} & \mathbf{w}_t^e = [& 0.0, \quad -0.15, \quad -0.15, \quad 0.15 \quad]
\end{array}
\tag{1}
$$

Behavior patterns are chosen according to the task is to be resolved. For instance the simulated foraging task resembles the food-retrieval of a laboratory rat when facing a novel environment. With this behavior type, rats present a preference for staying close to walls, and moving only to the center of the arena for collecting food. Thus, we decompose the task in algorithmic behavior patterns that can be assembled together to solve the simulated foraging task. Finally, we assume that behavior patterns were previously learnt, and can be considered as selected chunks of memory.

For instance, in chasing behavior (Section 6.2) selection occurs from three already-learnt behavior types, which are *tracking*, *locating*, and *exploring*. In this paper we propose that the selection component be thought of as a neural network (Figure 3). Therefore, instead of the delta rule, evolutionary learning could be used to set its weights. The single-layer feed-forward network (perceptron) has a distribution input layer with four neurons, and the piecewise-linear transfer function at the output layer of five neurons. Using sensor fusion, the Khepera raw sensory information is fed into the neural network, in the form of perceptual variables. Winner-takes-all is implemented at the output of the neural network by letting the highest output win the competition. In order to contribute to the calculation of the salience, the winning behavior type sends a busy signal to the computing output layer of the neural network. A behavior type is deselected when its salience is below the strongest salience.

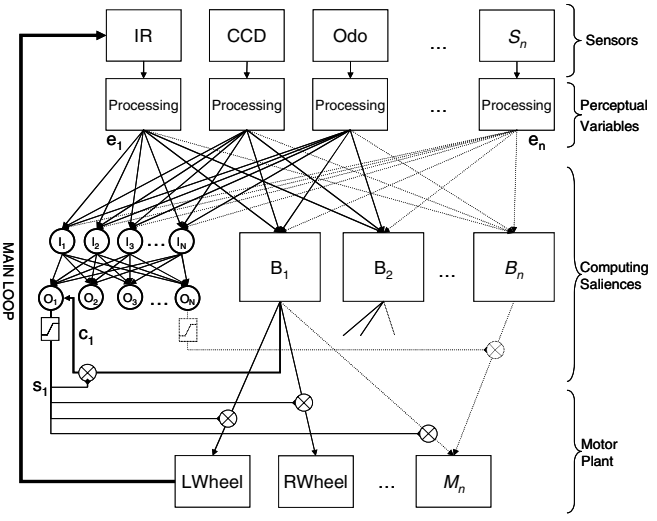


Fig. 3. Perceptual variables (e_i) form the input for the Salience Neural Network. The output selection of the highest salience (s_i) is gated to the motors of the Khepera. Notice the busy-status signal (c_1) from behavior $\mathbf{B_1}$ to the output neuron.

6 Experiments and Results

The relevance of developing animats, which solve tasks similar to those that face real animals in real controlled environments, offers clear proof of robustness and flexibility in the design of these animal robots. Therefore, the main focus of our experiments consists of facing CASSF with foraging and chasing tasks as a way of testing effective selection. We describe the setup for solving both tasks in our experiments below.

6.1 The Foraging Task

A square box was used as an arena for running the foraging task. Simulated food, such as wooden cylinders, was set in the middle of the arena. Two different kinds of food were employed, red for poisoned food, and blue for healthy food. Recharging facilities were provided, at one side of the arena, as some kind of solar plant battery. As long as the robot is close to the battery, recharging occurs for a limited period of time. For the experimental setup we employed a Khepera robot with a CCD camera, and a gripper, which was connected to the host computer using an RS232 interface and a BNC connector.

The definition of the various behaviors for the foraging task, as reported in Montes [2], was in this manner: *cylinder-seek* is used to travel the arena while avoiding obstacles, until some food is located; *cylinder-pickup* clears the space for grasping the cylinder; *wall-seek* finds a nearby wall; *cylinder-deposit* is employed for lowering and opening an occupied gripper; and *turning-around* makes a full spin for locating either the food or the battery. The experiments were carried out as follows; four blue

paper-wrapped cylinders and a red paper-wrapped cylinder were set in the center of a wooden square arena. The battery was simulated using red and green paper on both sides of a computer speaker. Usually the foraging task consists of four grasping-depositing bouts of blue cylinders, avoiding red cylinders, with sporadic battery-recharging periods. The foraging task is summarized in the form of an ethogram and some relevant statistics. The time resolution of the ethogram (Figure 4) is reported in seconds, and in seconds and milliseconds for the statistics analysis (Table 1).

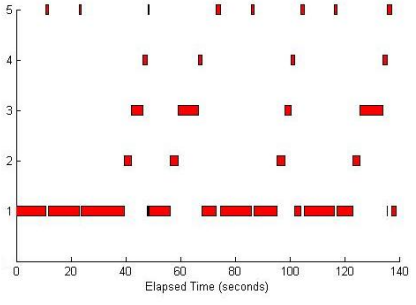


Fig. 4. Ethogram for a typical run of the foraging task. The bars indicate the time selection of one of these behavior types: (1) *can_seek*, (2) *can_pickup*, (3) *wall_seek*, (4) *wall_deposit*, and (5) *turn_around.*

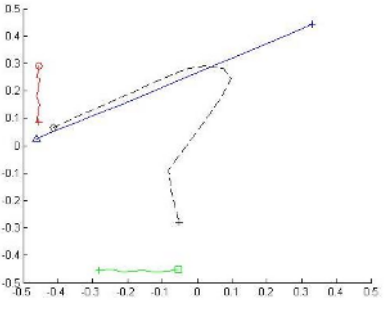


Fig. 5. Follower and leaders in the arena. A dashed line represents a follower. A cross indicates the starting point of the movement. The stop of the follower is marked with a diamond.

Table 1. Elementary statistics for a typical run of the foraging task

Behavioral Elements	Freq	Latency	TotDur	TotDur%	Mean	StdDev	StdErr	MinDur	MaxDur
1. can_seek	13.00	0.04	92.10	66.30	7.08	4.94	1.37	0.25	15.73
2. can_pickup	4.00	39.36	10.63	7.65	2.66	0.01	0.00	2.64	2.66
3. wall_seek	4.00	42.02	22.40	16.12	5.60	2.82	1.41	2.23	8.35
4. wall_deposit	4.00	46.38	6.17	4.44	1.54	0.01	0.01	1.53	1.56
5. turn_around	8.00	10.93	7.59	5.46	0.95	0.49	0.17	0.05	1.55
Total	33.00		138.89	99.97				0.05	15.73

None of the behavior types was selected before can-seek. However, it took only 0.04 seconds for can-seek to be selected. Later on, turn-around was selected after 10.93 seconds of the total elapsed time (138.93 seconds). Can-pickup was selected after 39.36 seconds, then wall-seek after 42.02 seconds, and finally wall-deposit after approximately 17 seconds of grasping the cylinder. The remaining three bouts, with different searching periods, were repeated in a similar fashion. Can-seek was selected most (13 times); secondly turn-around (8 times); finally, can-pickup, wall-seek and wall-deposit, all only 4 times.

6.2 The Chasing Task

Two different kinds of experiments were conducted for chasing behavior. On one hand, a predator with a frontal field of view hunts a prey with a peripheral field of

view [3]. On the other hand, a follower with a frontal camera chases one of three color-hatted leaders [4]. In both experiments central selection was used for the follower-predator robot with a choice of three types of behaviors, *wandering-exploring*, *locating* and *tracking*. The leader-prey uses a simple reactive controller for avoiding obstacles and running close to walls. In the experiments, the follower-predator and either the prey, or the three colored leaders (red, blue and green), were set in a short-walled arena. The follower-predator, equipped with the camera, was connected to the host computer as described for the foraging task. A radio modem was used to control the prey and the leaders. However, an aerial camera provided a peripheral view for the prey. The calculation of the minimal 'safe' distance between the prey and the predator was sent over TCP/IP. Therefore, in the case of a compromised situation, an avoidance reaction occurs. A typical chasing of the colored leaders, by the follower, is presented in Figure 5.

7 Conclusion

In this paper we have shown that CASSF effectively produces action selection. Firstly, in the above ethogram we observe four regular bouts of grasping-depositing food, which exhibits CASSF as been capable of obtaining overall coherent behavior. For example, the presence of an object in the gripper is enough to drive the selection of *wall-seek* behavior in the foraging task. Then, selection is terminated when the object slips from the gripper or a wall is found. Secondly, the central model of action selection is detached from the behavior patterns; thus, adaptability and extensibility is offered at relatively low cost by allowing independent modification of the selection module. Hence, the addition of more behavior patterns requires training and setting a new topology for the selection network. Thirdly, the calculation of the salience in the model offers appropriate selection for selected behavior patterns. Next, the busy-status signal accounts for boosting the salience when the behavior pattern has to be maintained, and switching-off this signal favors the termination of the behavior pattern. Therefore, the use of this signal avoids timing any behavior; thus, CASSF favors the expression of one type of behavior until its selection is proven ineffective. Moreover, in an off-status the busy signal facilitates the interruption of any of the behavior patterns during the execution of the action selection model.

Acknowledgments

This work has been sponsored by CONACyT-MEXICO grant SEP-2004-C01-45726.

References

1. Tyrrell, T., *Computational Mechanisms for action selection*, in *Centre for Cognitive Science*. 1993, University of Edinburgh.
2. Montes Gonzalez, F. and A. Marin Hernandez. *Central Action Selection using Sensor Fusion*. in *the 5th Mexican International Conference on Computer Science (ENC'04)*. 2004. Colima, México: IEEE Computer Society.

3. Montes Gonzalez, F. and A. Marin Hernandez. *The Use of Frontal and Peripheral Perception in a Prey-Catching System*. in *the 4th International Symposium on Robotics and Automation (ISRA 2004)*. 2004. Querétaro, México.
4. Montes Gonzalez, F. and D. Flandes Eusebio. *The Development of a Basic Follow-Behavior within a Distributed Framework*. in *the 1st IEEE Latin American Robotics Symposium (LARS 2004)*. 2004. México, D.F., México.
5. Prescott, T.J., P. Redgrave, and K.N. Gurney, *Layered control architectures in robots and vertebrates*. Adaptive Behavior, 1999. 7(1): p. 99-127.
6. Snaith, S. and O. Holland. *An investigation of two mediation strategies suitable for behavioural control in animals and animats*. in *From Animals to Animats: Proceedings of the First International Conference Simulation of Adaptive Behaviour*. 1990. Paris.
7. Werner M., G., *Using Second Order Neural Connections for Motivation of Behavioral Choices*, in *From Animals to Animats 3: Proceedings of the 6th International Conference on the Simulation of Adaptive Behavior*, D. Cliff, Editor. 1994, MIT Press: Cambridge, MA.
8. Redgrave, P., T. Prescott, and K.N. Gurney, *The basal ganglia: A vertebrate solution to the selection problem?* Neuroscience, 1999b. 89: p. 1009-1023.
9. Ludlow, A.R., *The behaviour of a model animal*. Behaviour, 1976. 58: p. 131-172.
10. Brooks, R.A., *A robust layered control system for a mobile robot*. IEEE Journal on Robotics and Automation, 1986. RA-2: p. 14-23.
11. Rosenblatt, K.J. and D.W. Payton. *A Fine-Grained Alternative to the Subsumption architecture for Mobile Robot Control*. in *Proceedings of the IEEE/INNS International Joint Conference on Neural Networks*. 1989.
12. McFarland, D.J., *Flow graph representation of motivational systems*. British Journal of Mathematical and Statistical Psychology, 1965. 18(1): p. 25-43.
13. McFarland, D.J., *Feedback Mechanisms in Animal Behaviour*. 1971, London: Academic Press.
14. Rosenblatt, J. K. and C. E. Thorpe, *Combining multiple goals in a behavior-based architecture.* in *Proceedings of 1995 International Conference on Intelligent Robots and Systems*. 1995.
15. Humphrys, S., *Action selection methods using reinforcement learning*, in *From Animals to Animats 4: Fourth International Conference on Simulation of Adaptive Behavior*. 1996.
16. Montes Gonzalez, F., T.J. Prescott, K. Gurney, et al., *An embodied model of action selection mechanisms in the vertebrate brain*, in *From Animals to Animats 6: Proceedings of the 6th International Conference on the Simulation of Adaptive Behavior*, J.A. Meyer, Editor. 2000, MIT Press: Cambridge, MA.
17. Prescott, T.J., F.M. Montes Gonzalez, K. Gurney, et al., *A robot model of the basal ganglia: behavior and intrinsic processing*. Neural Networks, (in press).

An Evaluation Method with Imprecise Information for Multi-attribute Decision Support*

Francesc Prats[1], Mónica Sánchez[1], Núria Agell[2], and Gaizka Ormazabal[3]

[1] Universitat Politècnica de Catalunya, Dept MA2, Jordi Girona, 1-3, 08034 Barcelona
Monica.Sanchez, Francesc.Prats@upc.edu
[2] ESADE Universitat Ramon Llull, Av. Pedralbes 62. 08034 Barcelona
Nuria.aAgell@esade.edu
[3] IESE Universidad de Navarra, Av. Pearson, 21. 08034 Barcelona
Gormazabal@iese.edu

Abstract. This paper presents a method using intervals for representing and synthesizing imprecise information for multi-attribute evaluation and decision-making support. An implementation is given for selecting an alternative for a project in a real case in the context of construction in civil engineering. As a previous step to aggregate the available information, a methodology is proposed for summarizing and normalizing values in an intervals context, representing the alternatives by means of rectangles in $\mathbb{R}^n$ (products of n finite closed intervals in $\mathbb{R}$). A distance is introduced in the set of rectangles, defining a total order once a reference rectangle is considered. A method is given for the choice of the best alternative based on the comparison of distances to a reference rectangle. The constraints which guarantee consistency are determined and the consistency of the method is established.

1 Introduction

In multiple attribute decision-making support processes, the evaluation of alternatives depends on the previous valuation of input variables [4], [9]. It is generally the case that the values of input variables are not precisely known, whereas an interval of possible values can be determined.

The approach presented is based on a unification of the initial information in an intervals context, a summarization of this information via a rectangle of $\mathbb{R}^n$, and evaluation by means of a distance to a reference rectangle. This in turn is based on an intervalar generalization of a kind of goal programming method known as the reference point method for vectorial optimization and decision-making support [3], [6], [7]. In general, reference point methods

* This work has been partially supported by the MCyT (Spanish Ministry of Science and Technology) projects MERITO (TIC2002-04371-C02) and MIVES (MAT2002-04310-C03-01).

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 133–142, 2006.

for optimization in $\mathbb{R}^n$ choose the alternative corresponding to the point at a shorter distance from a previously fixed reference point in $\mathbb{R}^n$(the "goal" to be reached).

In this work, optimization in the set of rectangles in $\mathbb{R}^n$ is performed by selecting, not an independent fixed reference rectangle but a "realistic" reference rectangle for the problem to be solved: The proposed reference rectangle is the supreme, with respect to the natural order, of the set of available alternatives, guaranteeing consistency with the order between rectangles.

The proposed methodology provides an appropriate system for comparison of alternatives in project evaluation in Civil Engineering [5]. In this domain, decision-making is generally based on intervalar values of involved variables, given that data is not a priori precisely known.

In Section 2, a qualitative representation is given of alternatives in the partially-ordered set $\mathcal{R}$ of the rectangles of $\mathbb{R}^n$ with sides parallel to the coordinates axis. In Section 3, some possible distances in $\mathcal{R}$, either weighted or otherwise, are defined, and one in particular is proposed. In Section 4, a total order in $\mathcal{R}$ is defined, in such a way that the set of rectangles corresponding to the available alternatives becomes a chain, and then the alternative chosen is the one represented by the maximum of this chain.

The property of consistency for the method of choice is established. Analysis of the necessary conditions under which consistency can hold leads to the determination of the reference rectangle as the supreme of the rectangles corresponding to the available alternatives with respect to the natural partial order in $\mathcal{R}$.

2 Alternatives Representation: The Partially Ordered Set $\mathcal{R}$

Let $\mathcal{R} = \{[a_1, b_1] \times \ldots \times [a_n, b_n] \mid a_i, b_i \in \mathbb{R}, a_i \leq b_i \; \forall i = 1, \ldots n\}$ be the set of the rectangles of $\mathbb{R}^n$ with sides parallel to the coordinate axis.

Each rectangle is interpreted as a set of n finite real intervals (each associated to an input variable) that define an alternative in such a way that, on every side, higher values always mean better results [2].

The order relation $\leq$ that respects this fact is considered, i.e., $R \leq R'$ means that alternative R' is better than alternative R. This order relation is built from the total order relation $\leq$ in $\mathbb{R}$, which in turn, induces a partial order relation between intervals of $\mathbb{R}$, $[a, b] \leq [a', b'] \iff a \leq a', \, b \leq b'$, which in turn, by extension to the Cartesian product, induces a relation in $\mathcal{R}$:

$$\prod_{i=1}^{n} [a_i, b_i] \leq \prod_{i=1}^{n} [a'_i, b'_i] \iff a_i \leq a'_i, \; b_i \leq b'_i \; \forall i = 1, \ldots, n. \tag{1}$$

This relation is trivially an order relation in $\mathcal{R}$, but a partial order, since there are pairs of non-comparable rectangles.

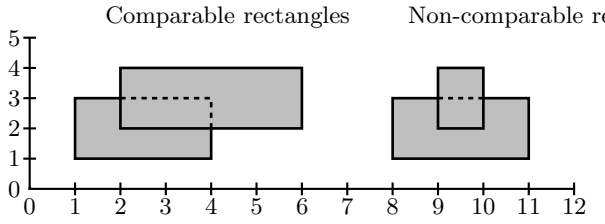


Fig. 1. The partial order $\leq$ in $\mathcal{R}$

3 Distances in the Set $\mathcal{R}$

A method is presented for building distances between n-dimensional rectangles. Although the analytic expressions of these distances, and so their qualitative interpretations, are different, all are mathematically equivalent in the sense that the open sets of the topologies induced are the same.

3.1 Non-weighted Distances

Proposition 1. If $(E_i, d_i)_{i=1,\ldots l}$ are metric spaces, then the cartesian product $\prod_{i=1}^{l} E_i$ is a metric space with the distances built from the distances d_i in E_i and either the Euclidean distance or the distance of the maximum, or the distance of Manhattan in $\mathbb{R}^n$:

$$d_e((x_1,\ldots,x_l),(y_1,\ldots,y_l)) = \sqrt{\sum_{i=1}^{l} d_i^2(x_i,y_i)} \tag{2}$$

$$d_m((x_1,\ldots,x_l),(y_1,\ldots,y_l)) = \max_{i=1,\ldots,l}\{d_i(x_i,y_i)\}) \tag{3}$$

$$d_s((x_1,\ldots,x_l),(y_1,\ldots,y_l)) = \sum_{i=1}^{l} d_i(x_i,y_i) \tag{4}$$

Proposition 2. Given a set X, and a metric space (E,d), any injective mapping $\Phi : X \hookrightarrow E$ induces a metric space structure in X, by means of $d_X(x,y) = d(\Phi(x),\Phi(y))$.

Therefore, it is possible to define a distance in the set of rectangles $\mathcal{R}$ by using the mapping Φ from $\mathcal{R}$ into the cartesian product of $\mathbb{R}^n$ for n copies of $\mathbb{R}$, $\Phi : \mathcal{R} \hookrightarrow \mathbb{R}^n \times \mathbb{R} \times \cdots \times \mathbb{R}$ given by

$$\Phi(R = \prod_{i=1}^{n}[a_i,b_i]) = (c(R), l_1(R), \ldots, l_n(R)) \tag{5}$$

where $c(R) = (\frac{a_1+b_1}{2},\ldots,\frac{a_n+b_n}{2})$ is the center of the rectangle R and $l_1(R) = b_1 - a_1, \ldots, l_n(R) = b_n - a_n$ are the lengths of the sides of R.

This mapping is evidently injective, given that any rectangle with sides parallel to the coordinate axis is determined by its center and the lengths of its sides. Therefore, from proposition 2, any distance in $\mathbb{R}^n \times \mathbb{R} \times \cdots \times \mathbb{R}$ induces a distance in $\mathcal{R}$.

From proposition 1, distances in the product $\mathbb{R}^n \times \mathbb{R} \times \cdots \times \mathbb{R}$ can be built from d_e, d_m or d_s and different distances in each factor of $\mathbb{R}^n$, $\mathbb{R}$,...,$\mathbb{R}$.

In the case of the study presented as an implementation in this work, the Euclidean distance is taken in the first factor of the product $\mathbb{R}^n \times \mathbb{R} \times \cdots \times \mathbb{R}$,the distance of Manhattan in the other n factors $\mathbb{R}$ and the distance d_s of proposition 1 for their combination, but considering different weights (assessed by experts) for the different factors. Without weights, the induced distance in $\mathcal{R}$ would be:

$$d_{\mathcal{R}s}(R, R') = d_s(\Phi(R), \Phi(R')) = d_{euc}(c(R), c(R')) + \sum_{i=1}^{n} |l_i(R) - l_i(R')|. \quad (6)$$

This distance provides an intuitive notion of proximity between n-dimensional rectangles, since it takes into account the position of their centers, related to the value magnitude of the alternatives, and the lengths of their sides, related to the imprecision of the data of the input variables. Moreover, this distance applied to the case of maximum precision, in which the intervals are reduced to points, is precisely the Euclidean distance in $\mathbb{R}^n$.

3.2 Weighted Distances

If in the rectangles $R = \prod_{i=1}^{n} [a_i, b_i]$ each of the intervals $[a_i, b_i]$ has a different importance, due to the fact of different variables having different relevance for the evaluation, certain positive weights $\alpha_1, \ldots, \alpha_n$ can be applied. Distances in $\mathcal{R}$ taking into account these weights can be built by changing the injection Φ by a weighted injection. Taking $\Psi(R) = (c(R), \alpha_1 l_1(R), \ldots, \alpha_n l_n(R))$, the following weighted distance is obtained:

$$d_{\mathcal{R}spond}(R, R') = d_s(\Psi(R), \Psi(R')) = d_{euc}(c(R), c(R')) + \sum_{i=1}^{n} \alpha_i \, |l_i(R) - l_i(R')|, \quad (7)$$

an expression in which the weights of each side of the rectangles in $\mathcal{R}$, when existing.

If it is also wished to weight the distance between the centers of the rectangles with respect to the lengths of the sides, it is only necessary to take a number $\beta > 0$ and choose a suitable weighted injection to obtain:

$$d_{\mathcal{R}spond}(R, R') = \beta d_{euc}(c(R), c(R')) + \sum_{i=1}^{n} \alpha_i \, |l_i(R) - l_i(R')|, \quad (8)$$

4 Choice of the Best Alternative

Starting from a distance in $\mathcal{R}$ and a reference rectangle $\overline{R}$, a total order $\trianglelefteq$ can be defined in $\mathcal{R}$, such that the set of rectangles $R^1, \ldots, R^k$ corresponding to the available alternatives become a chain: $R^{i_1} \trianglelefteq \cdots \trianglelefteq R^{i_k}$. Then alternative R^{i_k} corresponding to the maximum of the chain will be chosen.

4.1 A Total Order in $\mathcal{R}$

Let $\overline{R} \in \mathcal{R}$ be any rectangle and let us call it the reference rectangle.

Let d be any of the distances defined in $\mathcal{R}$ in Section 3, then the following binary relation in $\mathcal{R}$:

$$R \preceq R' \Longleftrightarrow d(R', \overline{R}) \leq d(R, \overline{R}) \tag{9}$$

is a pre-order, i.e. it is reflexive and transitive.

This pre-order relation induces an equivalence relation in $\mathcal{R}$ by means of:

$$R \equiv R' \Longleftrightarrow R \preceq R' \ , \ R' \preceq R \Longleftrightarrow d(R', \overline{R}) = d(R, \overline{R}). \tag{10}$$

Then, in the quotient set $\mathcal{R}/\equiv$ the following relation between equivalence classes:

$$\text{class}\,(R) \trianglelefteq \text{class}\,(R') \Longleftrightarrow R \preceq R' \Longleftrightarrow d(R', \overline{R}) \leq d(R, \overline{R}) \tag{11}$$

is a total order relation.

In this way, given a set of alternatives $R^1, \ldots, R^k$, these can be ordered as a chain with respect to their proximity to the reference rectangle: $\text{class}\,(R^{i_1}) \trianglelefteq \cdots \trianglelefteq \text{class}\,(R^{i_h})$. Alternatives belonging to the same equivalence class, i.e. alternatives at the same distance from $\overline{R}$, will be regarded as alternatives with the same value, therefore, from now on an abuse of notation will be made by changing $\preceq$ into $\trianglelefteq$: $R^{i_1} \trianglelefteq \cdots \trianglelefteq R^{i_k}$.

4.2 Consistency of the Method of Choice

The method of choice of the best alternative via a distance to a reference rectangle is really necessary when no alternative is better than all the rest with respect to every variable, i.e., when the set $\{R^1, \ldots, R^k\}$ has no maximum with respect to the order relation $\leq$.

But when $\{R^1, \ldots, R^k\}$ has a maximum R^m with respect to $\leq$, that is to say, when there already exists a priori an alternative better than the other R^m, the proposed method for choice will be consistent if it provides the same R^m as the best alternative. Formally, given any$R^1, \ldots, R^k \in \mathcal{R}$:

$$\exists m \in \{1, \ldots, k\} \ \ R^i \leq R^m \ \forall\, i = 1, \ldots, k \Longrightarrow R^i \trianglelefteq R^m \ \forall i = 1, \ldots, k \tag{12}$$

The property of consistency is equivalent to the following, which is more manageable:

$$(\forall\, R, R' \in \mathcal{R})\,[R \leq R' \Longrightarrow R \trianglelefteq R'] \tag{13}$$

Therefore, the total order $\trianglelefteq$ that provides the best alternative has to be consistent with the initial parcial order $\leq$ in $\mathcal{R}$ induced by the order in $\mathbb{R}^n$.

The following proposition determines the construction of the reference rectangle $\overline{R}$ for any set of rectangles $R^1, \ldots, R^k$ which have a maximum with respect to the partial order $\leq$:

Proposition 3. If for any set of rectangles $R^1, \ldots, R^k$ with maximum R^m the rectangle $\overline{R} = R^m$ is chosen as reference, then the property of consistency is accomplished. Otherwise, this property can not be assured.

Proof: The first statement is trivial, it is sufficient to assume $R \leq R'$, to take $\overline{R} = R'$ and to check that $d(R', \overline{R}) = d(R', R') = 0 \leq d(R, R')$.

In the case $\overline{R} \neq R^m$, a counterexample of the property of consistency can always be found. In particular, for the distance $d_{\mathcal{R}s}$ introduced in Section 3.1, the following counterexample can be considered:

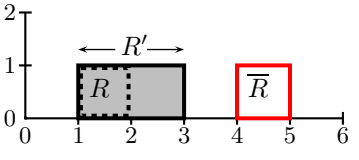


Fig. 2. Counterexample of the consistency

We have $d(R, \overline{R}) = 3$, $d(R', \overline{R}) = 3.5$, so the "nearest" to $\overline{R}$ is R, therefore the alternative chosen would be the one corresponding to R, and, nevertheless, $R \leq R'$.

4.3 Selection of the Reference Rectangle

The consistency of the proposed method determines the reference rectangle in the case of a set of rectangles with a maximum, as has been proved. The natural generalization to any set of rectangles is the following:

Given any $R^1, \ldots, R^k$, the supreme of these rectangles with respect to the partial order $\leq$ will be taken:

$$\overline{R} = \sup\{R^1, \ldots, R^k\}. \tag{14}$$

Note that in the case of rectangles with a maximum, this supreme is in fact the maximum, so this choice of the reference rectangle holds the property of consistency of the method.

The maximum $\max\{x_1, \cdots, x_n\}$ is the supreme of $x_1, \cdots, x_n$ with respect to $\leq$ in $\mathbb{R}$, so is immediate to prove that: if $R^j = \prod_{i=1}^{n} [a_i^j, b_i^j]$, for $j = 1, \ldots, k$, then

$$\overline{R} = \sup\{R^1, \ldots, R^k\} = \prod_{i=1}^{n} [\max\{a_i^1, \ldots, a_i^k\}, \max\{b_i^1, \ldots, b_i^k\}]. \tag{15}$$

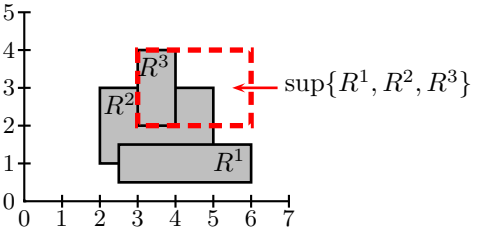


Fig. 3. Reference rectangle: the supreme

4.4 Selection of the Best Alternative

Finally, the steps of the proposed method for selecting the best alternative are:

- To fix a distance d in $\mathcal{R}$: $d_{\mathcal{R}s}$.
- To choose a reference rectangle $\overline{R}$: the supreme of the set of alternatives.
- To assign to each rectangle R the value $d(R, \overline{R})$; so, the alternatives are ordered (with respect to the defined total order) as a chain,
- To choose as the best alternative the maximum of the chain, that is to say, the one (or ones) of minimum distance.

5 Application to a Real Case of Decision-Making Support for the Performance of an Engineering Project

The study presented is framed within the project for Line 9 of the Underground in Barcelona, which is now in the process of construction. In this project the possibility was considered of taking into account a constructive solution different from others chosen in similar public works. Specifically, two alternatives were raised:

- The first alternative would be the innovative solution of adopting a tunnel diameter of 12 m with the stations included in the tunnel.
- The traditional design with a 9 m diameter and stations built in the openair, because they do not fit into the tunnel layout. This was the solution adopted in Metrosur in Madrid.

In the first alternative, the underground would circulate at two heights, and in the 9 m diameter solution, the railway traffic would be at the same height in both directions. The essential difference between the two is that in the case the 12 m diameter the station platforms fit into the section of the tunnel, and with the other alternative it is necessary to carry out extra excavation in order to ensure that the platforms are at the same height as the tunnel layout. This difference is very important because it will have a substantial effect on several aspects of the project. Specifically, the criteria which are representative and make the two alternatives different are represented in Table 1.

Table 1. Representative Variables

Aspect	**Weight**	**Criterion**	**Weight**
Economical	0,3	Construction cost (layout + stations)	1
Functionality	0,3	Platform comfort	0,625
		Entrance and exit comfort	0,375
Security	0,133	Evacuation capacity	0,5
		Fire resistance	0,5
Environment Impact	0,133	Effect of site treatment	0,5
		Residues generated	0,5
Social impact	0,133	Inconvenience for neighbors	1

In Table 1 it can be seen that a weight has been assigned to each aspect, and also to each requirement included within each aspect. The assignation of the weights was performed by engineers using the AHP method (Analytical Hierarchy Process) [1], [8].

For the measurement of the above criteria, the following indicators or variables were considered:

Table 2. Variable estimation of the indicators used in the evaluation of alternatives. Remark: the values correspond to an estimation by intervals.

Criterion	**Indicator**	**Alt. 1**	**Alt. 2**
Construction cost (layout + stations)	Budget in euros of layout and stations	217,923,467 € 266,350,904 €	457,788,153 € 559,518,854 €
Platform comfort	Platforms capacity (m^3) (per unit of area)	267.63 m^3	374.54 m^3
Entrance and exit comfort	Section width (m)	14.875 m 20.125 m	21.675 m 29.325 m
Evacuation capacity	Section width usable for evacuation (m)	13.3 m 14.7 m	16.15 m 17.85 m
Fire resistance	REI	REI280	REI120
Effect of site treatment	Area affected by ground subsidence (m^2)	358.521 m^2 438.193 m^2	264.478 m^2 323.251 m^2
Generated residues	Non-usable materials extracted volume (m^3)	343.804 m^3 416.539 m^3	498.007 m^3 608.675 m^3
Inconvenience for neighbors	Area affected by ground subsidence in all stations (m^2)	656.169 m^2 801.984 m^2	742.024 m^2 906.918 m^2

Because the indicators taken are very heterogeneous, a transformation to a non-dimensional scale, with values between 0 and 1, was applied. To achieve this, a value function for each indicator will be defined. These functions are defined in a linear way between two variation values. For example, the value function of the indicator corresponding to the construction cost will be taken between the values $x_0 = 150.000.000$, with assigned value $v(x_0) = 1$, and $x_1 = 600.000.000$, with

assigned value $v(x_1) = 0$. The linear value function is defined from these two values corresponding to the most probable rank of variation of the indicator. In the case of the former indicator, the value function will be decreasing, although it can be increasing for other indicators. Obviously, this characteristic of the value function will depend on whether the increase in the numerical value of the indicator is advantageous or disadvantageous for the project.

Once these value functions have been defined and applied to each indicator, the methodology proposed in this article is implemented, from the standardized non-dimensional values (all between 0 and 1). The results are shown in Table 3.

Table 3. Results

Parameter	**Alternative 1**	**Alternative 2**
Weighted sum of the absolute values of the difference between the sides of each alternative and the sides of the reference rectangle (supreme)	0.103	0.098
Distance between the center of the rectangle corresponding to each alternative and the center of the reference rectangle (supreme)	0.535	0.592
Total distance	**0.638**	**0.690**

As can be seen, the two alternatives offer similar values, both in their dispersion (difference between the sides of each alternative and the sides of the reference rectangle) and in the distance between the centers of the rectangles associated to the alternatives and the center of the reference rectangle. This highlights the difficulty of decision-making, and is in line with the fact that in Madrid and Barcelona, different solutions have been adopted, because a few variations of the indicators can lead to opting for the other alternative.

In any case, the fact that the distance corresponding to alternative 1 is less than the other alternative implies that, according to this calculation, this solution seems better, and so supports the decision adopted in the case of the Barcelona underground.

6 Conclusion

In this work, a methodology is proposed for the evaluation of multi-attribute alternatives based on the use of distances to a reference point. The use of intervalar variables allows application problems with imprecise information to be confronted in a consistent way. The application of the proposed methodology to the comparison of alternatives in a real case of Civil Engineering proves its utility.

The presented methodology allows, on the one hand, the imprecise concepts of the specific application to be handled, and, on the other, the methods of "goal programming" to be generalized without the need for previous knowledge of the ideal goal.

Software implementing this method is being developed as part of the MIVES research project (Integrated model of Quantification of a sustainable constructive projectś value) supported by the Spanish Ministry of Science and Technology MCyT (MAT2002-04310-C03-01), implementing the method presented for the automatic evaluation of alternatives for construction projects.

A future work will study the efficiency of other intervalar distances in the methodology and the applicability to other domains. Moreover, the system will be adapted to the case of variables defined over qualitative spaces of absolute order of magnitude.

References

1. Al-Subhi, K.M. "Application of the AHP in Project management". International Journal of Project Management, 19 (2001) 19-28.
2. González Pachón, J., Romero López, C. Aggregation of partial ordinal rankings. An interval goal programming approach. Computers and operations research. 28 (2001) 827-834
3. Kallio,M., A. Lewandowski and W. Orchard-Hays An Implementation of the Reference Point Approach for Multi-Objective Optimization. WP-80-35, IIASA, Laxenburg (1980)
4. Keeney, R.L., and Raiffa, H. (1993). Decisions with multiple objectives preferences and value trade-offs, Cambridge University Press.
5. Ormazabal, G. (2002). "IDS: A new integrated decision system for construction project management". PhD Thesis, Department of Construction Engineering, Technical Univ.of Catalonia (UPC), Barcelona, Spain.
6. Romero López, Ca. Extended lexicographic goal programming: A unifying approach. Omega, The International Journal of Management Science. 29 (2001) 63-71
7. Romero López, C., Tamiz, M., Jones, D. Comments on goal programming, compromise programming and reference point method formulations: linkages and utility interpretations-A reply. Journal of the Operational Research Society 52 (2001) 962-965
8. Saaty,T.L. (1990). The Analytical Hierarchy Process. Ed. Willey. New York,
9. Wierzbicki, A.P. The use of reference objectives in multiobjective optimization. In G. Fandel, T. Gal (eds.): Multiple Criteria Decision Making; Theory and Applications, Lecture Notes in Economic and Mathematical Systems. 177 Springer-Verlag, Berlin-Heidelberg (1980) 468-486

Classification Algorithms for Biomedical Volume Datasets

Jesús Cerquides[1,⋆], Maite López-Sánchez[1], Santi Ontañón[1], Eloi Puertas[1,2], Anna Puig[1], Oriol Pujol[1], and Dani Tost[3]

[1] WAI: Volume Visualization and Artificial Intelligence Group, Dept. MAiA, UB
cerquide@maia.ub.es
[2] IIIA: Institut d'Investigació en Intel·ligència Artificial, CSIC
[3] CREB: Centre de Recerca en Enginyeria Biomèdica, UPC

Abstract. This paper analyzes how to introduce machine learning algorithms into the process of direct volume rendering. A conceptual framework for the optical property function elicitation process is proposed and particularized for the use of attribute-value classifiers. The process is evaluated in terms of accuracy and speed using four different off-the-shelf classifiers (J48, Naïve Bayes, Simple Logistic and ECOC-Adaboost). The empirical results confirm the classification of biomedical datasets as a tough problem where an opportunity for further research emerges.

1 Introduction

Volume rendering has emerged as one of the most active fields in Scientific Visualization. It consists of rendering property values measured at points of a 3D volumetric region. One of the major applications of volume rendering is the visualization of medical data captured with 3D imaging devices such as Computer Tomographies. In these applications, the 3D region is sampled according to a regular 3D grid, by parallel image planes. For example, a typical data set is composed by 512^3 or 1024^3 samples. The representation of the volume is a voxel model consisting of a set of parallel cubical and face-adjacent cells called *voxels* with a property value at each voxel vertex and such that the reconstruction of the property inside a voxel can be done by interpolation of the voxel vertices property values. The property is usually *tissue density*, scaled as an *intensity* level between 0 and 2^n, being n the number of bits allocated for the storage of the intensity.

During rendering, the voxel model is traversed. The intensity is computed at a set of 3D positions in the volume called *rendering samples*. Every rendering sample is then shaded according to the lighting conditions and to the optical properties of the anatomical structure to which they belong. Finally, rendering samples are ordered to compute the final 2D projection.

The definition of the optical properties can be viewed as an elicitation process which extracts user knowledge about the anatomical structures contained in the data, the selection of the visualization preferences and the appearance of the

⋆ Corresponding author.

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 143–152, 2006.

different tissues structures. This elicitation process defines the Optical Property Function (OPF) which is a 3D continuous function defined for all spatial points (x, y, z) contained into the data voxel model to the optical properties, such as emission (R, G, B) and absorption (α).

The elicitation process is often performed through the user definition of *transfer functions*. These functions directly associate optical properties to the different data values. Thus, the OPF at each point is computed as a mapping of its property value to the corresponding optical properties. Obviously, users should assign coherently similar optical properties to data values corresponding to the same regions. Selection of the anatomical regions to be visualized is accomplished indirectly by assigning to zero the optical property of opacity, since totally transparent samples do not contribute to the final image.

The use of transfer functions presents a major advantage: they can be stored as look-up tables, directly indexed by the intensity data values during the visualization, which significantly speeds up rendering. However, their manual definition is complicated even for skilled users. A lot of effort has been put on developing user friendly interfaces that make this definition more intuitive. Nevertheless, it is still an open problem. In addition, it has been proven [1] that using only one-dimensional transfer functions, based on the intensity data levels fails at accurately detecting complex combinations of material boundaries. The use of multidimensional transfer functions based on the first and second derivative of the intensity values brings major refinements to the classification but it makes harder its definition and increases the memory requirements for its storage.

The transfer function between intensity values and optical properties is often broken into two: the Classification Function (CF) and the Structure to Optical Properties Assignment Function (SOPAF). CF is a continuous function which determines, for each point inside the voxel model, the specific anatomical structure it belongs to. SOPAF is a function that assigns to each anatomical structure a set of optical properties.

In this case, the elicitation is a two-step process. During the classification step a labelled voxel model is created that contains a unique identifier of the region to which the voxel belongs. This classified model is then used together with the original voxel model to build a (R, G, B, α) model to be visualized.

In this approach, the classification function is used to skip non-selected regions, and thus reduces the cost of model traversal. It simplifies the edition of the transfer functions, separating the problem of selection and classification from the optical properties assignment problem. A drawback of this approach is that the usage of an intermediate labelled model increases memory requirements, which are critical during rendering.

On the contrary, the advantage of this approach is that, since the classification is carried on as a preprocess before rendering, it can cope with the usage of more complex and computationally expensive classification methods than transfer functions. In particular, a first step of segmentation can be applied to separate the regions of interest, followed by a labelling of the segmented regions which results in the assignment of a region identifier per voxel [2]. This strategy cannot

be applied if classification is done on-the-fly, during rendering, but it is suitable for a pre-process. Another promising approach is the application of probabilistic classifiers. It was early described in [3], but has nevertheless been little addressed since then. Its major advantage is that it can be automatized for similar datasets. This paper focuses on this direction.

Many papers in volume rendering literature address classification [4]. Most of them are based on the edition of transfer functions and specially on the design of user friendly interfaces for their specification. Recently, some preliminary work based on learning methods have been published: supervised methods such as bayesian networks [5], neural networks [6] , decision trees [7] and non-supervised methods [8]. Additionally, in [9], clustering-based supervised and non-supervised learning methods are compared for the classification of magnetic resonance images (MRI). However, there is a lack of a systematic comparative study of the application of different learning methods to classification.

In this paper, we address classification as a *data mining* problem. We interpret voxels as objects to classify and their property values, derivatives and positions as the attributes to evaluate. We apply different learning methods to a subset of already classified voxels and after that, in order to test our method, we classify various voxel models. Our goal is three-fold: to define an optical property function elicitation process based on machine learning approach; to compare the adequacy of five different learning methodologies to classify data; and to determine the size and type of sampling of classified subsets used for learning that are more suitable for rendering.

2 Incorporating Machine Learning into the Optical Property Function Elicitation Process

Machine learning in the most general sense allows a computer program to learn to perform a task by the analysis of previously solved tasks. If we consider the most general setting for the optical property function elicitation process, the task that we would be interested in automating is the transformation of a data voxel model into an optical property function. From a formal point of view, the task has as input a set of pairs, each containing a data voxel model and an optical property function used to visualize it (i.e. a problem and its solution). The output is a function that maps a data voxel model into its corresponding optical property function. Due to the particular structure of this problem, most of the machine learning techniques in the literature cannot be applied "out-of-the-box". Furthermore, the possibilities for the user to control this process are— very limited.

These two problems, unsuitability of common machine learning tools and lack of user control, can be overcome if we decide to apply machine learning to the class based process. The proposed resulting process can be seen in Figure 1. In this case, learning is applied only to the process of classification. Machine learning methods are much easier to apply to this process and the user gains control over the processes of selection and assignment of optical properties to the different classes or materials.

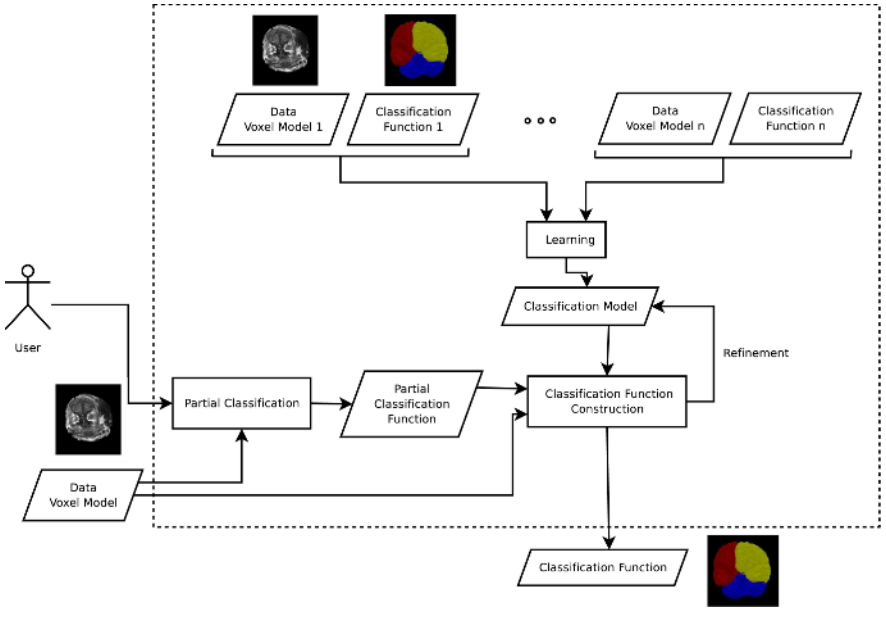


Fig. 1. The class based direct volume rendering process incorporating machine learning

Figure 1 introduces the concept of *classification model*. A classification model contains the information needed to map a data voxel model into a classification function. It can be understood as an actionable compilation of the information residing in the training data provided as input to the process of *learning*.

In order to build a CF for an incoming data voxel model, we use the process of *classification function construction*, where the classification model is applied. Furthermore, sometimes it can be useful to incorporate additional information in the form of a *partial classification function*, provides information about the classification of a subset of the space. This can be defined by the user by means of the *partial classification* process. For example, this gives us the possibility to ask the user to classify several 2D slices of the data and incorporate this information into the CF construction process. The information provided by the partial CF can also be used to improve or refine the classification model.

2.1 Incorporating Attribute-Value Learning

Several alternatives can be used to implement the learning process. A common assumption underlying many of the algorithms proposed by the pattern recognition and machine learning communities is the fact that the objects that are going to be classified can be described by a set of attributes, where each attribute can take a set of values. There are a plethora of algorithms embracing this assumption, and they are usually known as attribute-value learning algorithms.

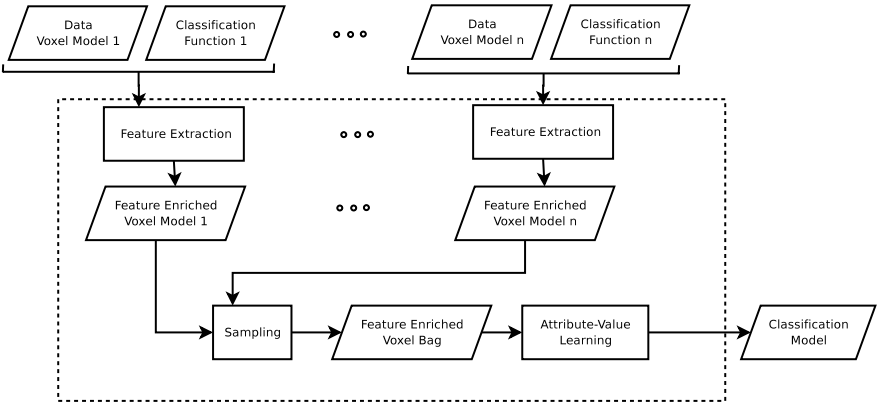


Fig. 2. The learning process adapted to use attribute value learning algorithms

The application of these algorithms turns out to be much easier if we divide the problem of classifying a data voxel model into, say 256^3, independent problems, one per voxel. Accepting this assumption, however hinders the detection of contiguous regions, introducing discontinuities in the classifications. In order to alleviate this process, usually a *feature extraction* process is used, where for each voxel we can incorporate information about its neighborhood. In Figure 2, we can see the steps when the learning process is adapted to attribute-value algorithms, including a sampling step, where a subset of all the voxels available for learning is selected.

3 Simulations

3.1 Metrics and Analyzed Algorithms

In order to evaluate the accuracy of the learning methods, we have used manually-guided expert segmentation models as the target classification functions of the data voxel models. From now on, we will call reference model (RM) these labelled models. Moreover, we have constructed the set of classified instances, SCI, by sampling these reference models. We have used SCI as input for both processes, the partial and the complete classification process. Finally, we have compared the results of the models classified with the learning methods with the corresponding reference models RM.

We have chosen five different sampling methods to construct the set of classified instances: random, lowres, systematic, slice and stratified sampling. The random sampling selects a subset of data set instances at a random choice. Both lowres and systematic sampling extract a subset of instances at regular sampling steps. However, systematic uses a random initial offset. These sampling can be used for scaling our data sets as a 3D geometrical transformation, and they can be interpreted as if our original data had been captured with lower resolution.

As its name points out, slice sampling implies to choose a few 2D slices of the 3D voxel model. Finally, the stratified sampling divides the data set into disjoint subsets, one for each final class in order to guarantee that the learning algorithm will have at least one instance in each class to be learned.

In the present study, we analyze four different attribute-value learning algorithms in order to evaluate their performance and their trade-off between time and accuracy in similar volume data. The evaluated algorithms are:

- **Naïve Bayes** [10]**:** It is a widely known, simple but very robust algorithm that is commonly used as a reference method. It can only capture simple dependencies between the non-class attributes and the class attribute.
- **J48:** Induction of decision trees [11] is one of the most used off-the-shelf classification algorithms that provides a good balance between accuracy, speed and interpretability of results.
- **Simple Logistic:** Logistic regression models are commonly used by the statistical community. The simple logistic [12] algorithm is an up-to-date representative of this family that uses boosting [13] to calculate regressors.
- **ECOC-AdaBoost:** Error correcting output codes (ECOC) [14] is a general framework for handling multiclass problems by reducing them to simpler binary classification ones. We have chosen AdaBoost [13] as a basis classification technique. This combined technique is introduced in the comparison as a state-of-the-art classifier in terms of accuracy and speed.

In the original experimental design we included a fifth algorithm based on neural networks, but backpropagation turned out to be much too expensive in terms of learning time for this size of datasets.

To evaluate the quality of the classification results, the Overlap Metric, OM, is used. Let C_{v_A} be the classification of class C_v using the learning method A. The Overlap Metric is defined as follows,

$$OM = \frac{|C_{v_A} \bigcap C_{v_{RM}}|}{|C_{v_A} \bigcup C_{v_{RM}}|}$$

where $C_{v_{RM}}$ stands for the Reference Model for the class C_v. For each class, this metric approaches a value of 1.0 for results that are very similar and it is near 0.0 when they share no similarly classified voxels.

The classification algorithm results are evaluated on the basis on their learning time, the memory they used, their accuracy and their time of classification of a new instance.

3.2 Experimental Results

In order to compare these different learning algorithms we have used datasets from The Internet Brain Segmentation Repository (IBSR) [15], which provides manually-guided expert segmentation results along with magnetic resonance 3D brain image data. In particular, they correspond to coronal three-dimensional T1-weighted spoiled gradient echo MRI scans performed on two different imaging

systems to male and female patients. These images have been segmented in four different regions: cerebrospinal fluid (D128), grey matter (D192), white matter (D254), and other (D0), which includes background and corresponds to more than 90% of the data sets.

The datasets used were the ones under 6_10, 15_3, 16_3 directories, each of them containing a variable number of consecutive images (ranging from 51 up to 61) of size 256 x 256.

The training process was done using case 6_10. Each data voxel model involves about four million instances. Each instance is composed of a 5-dimensional attribute vector. The attributes used are the classical features reported in literature: 3-D position, intensity value and gradient magnitude.

The test set is composed by two perturbed cases (15_3 and 16_3). Those test sets suffer from different artifacts associated to the input device, such as abrupt contrast changes, high intensity peak values, and so on.

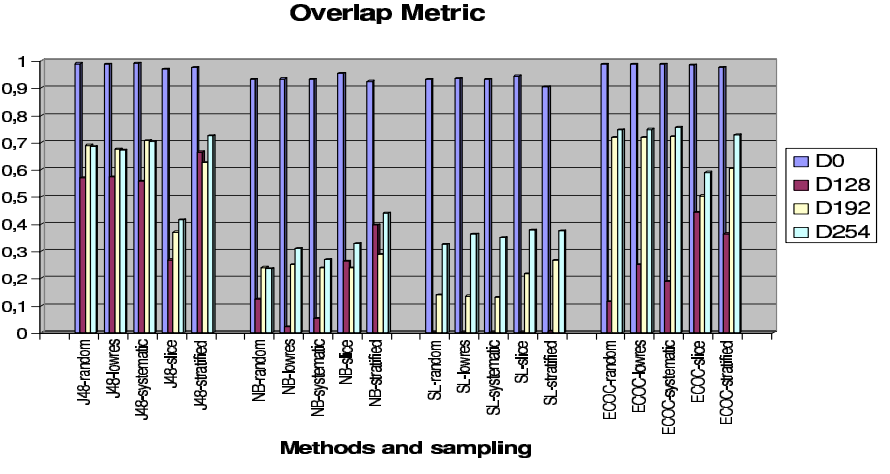


Fig. 3. Overlap metric evaluation for each class, sampling and method for case 6_10

Figure 3 shows the overlap metric evaluation for J48, Naïve Bayes, Simple Logistic and ECOC-AdaBoost, using 6_10 as test. The analysis of this graphic illustrates the completion of a partial classification function containing 2% of the voxels of 6_10. The overlap metric evaluation is closely related to the accuracy rate achieved for each class. For all the methods, the classification accuracy of the different classes keeps the same relationship: D0 is clearly the best classified, followed by D192, D254 and D128. These differences can be explained by the characteristics of the classes. For example, D0 corresponds to a non-anatomical region. It concentrates most of the samples and covers a neat range of property values. On the contrary, class D128 is the smallest class and thus harder to learn. We observe in the figure that J48 and ECOC-Adaboost have an overall higher accuracy rate. However, although ECOC-Adaboost and J48 are fairly similar in the recognition rate of three of the classes, ECOC-Adaboost fails to obtain as

good accuracy rates as J48 does for the remaining class (D128). On the other hand, Naïve Bayes and Simple Logistic behave in a similar way. However, we must note that Simple Logistic fails to find class D128.

Figure 3 also shows the performance when the sampling policy is changed. Again, we observe that J48 has the most regular behavior while maintaining a good Overlap Metric value. On the other hand, Naïve Bayes and Simple Logistic show low Overlap Metric values, though they are robust to the various sampling policies.

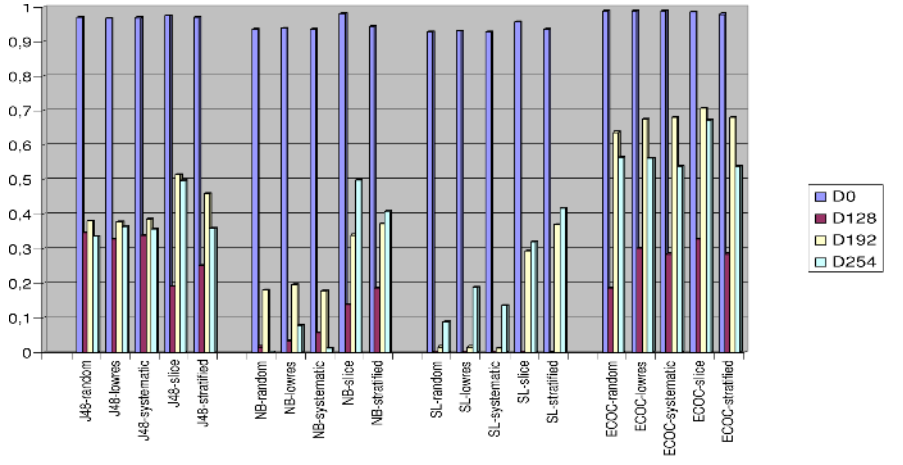


Fig. 4. Overlap Metric evaluation for each class, sampling and method for case 16_3

Figure 4 shows the overlap metric evaluation using 16_3 as a test case. The evaluation of this figure illustrates the performance of the classification function construction process. In this case, ECOC-Adaboost outperforms J48 in accuracy and robustness to the sampling policy. On the other hand, either Naïve Bayes and Simple Logistic have low recognition rates in the tissue classes. We can observe that Simple Logistic misses the D128 class again.

When analyzing the different sampling policies, if we observe figures 3 and 4, it is interesting to note the fact that stratified and slice methods provide more robustness for Simple Logistic and Naïve Bayes. However, J48 and ECOC-Adaboost perform fairly regular with all policies. Stratified seems the preferred method while slice sampling behaves the worst. The low Overlap Metric value in the slice method is due to the fact that it can easily fail to represent the variability in the training domain if the slices are not properly chosen. The results of the tests performed using 15_3 voxel model confirm these analysis.

As a result of the former experiments we have evaluated the training and test time performance. Since ECOC-Adaboost uses a predefined number of classifiers the training and test time are constant for each sampling method. For reference, the training time using Matlab is 40 seconds and test time for a whole voxel model is 150 seconds. However, the J48 training time depends on the characteristics of the input dataset. Figure 5 shows the training and test time. We

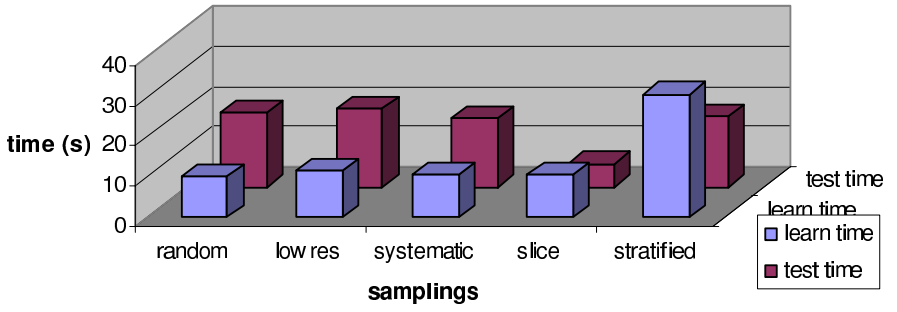


Fig. 5. Training and test time performance for J48 on 16_10

observe that using the stratified method, the amount of time increases due to the fact that the resulting decision tree is deeper. This result is expected since the stratified method changes the priors for each class. The evaluation of J48 has been done using Weka [16]. The average training time is around 10 seconds and test time is 17 seconds. Note also the decrease in test time when applying slice sampling, that reflects a smaller variability of the training set.

4 Conclusions and Future Work

In this paper we have analyzed how to introduce machine learning algorithms into the process of direct volume rendering. We have developed a conceptual framework for the optical property function elicitation process and have particularized it for the use of attribute-value classifiers. We have tested the suitability of this process through the experimental evaluation in terms of accuracy and speed of four different learning algorithms.

Our results regarding efficiency point out that the process can be considered feasible when implemented as a preprocessing step in the visualization pipeline.

From the point of view of accuracy, it should be noted that average accuracy is not providing all the information needed. The Overlap Metric is more informative since error information relative to each class highlights which classes are more prone to error. Under this perspective, our classification results are still not usable for real life applications. However, it should be noted that all the evaluated learning methods are general purpose classifiers. Better results should be expected if specific algorithms are designed, taking benefit from the specificity of our classification problem. In addition, a richer learning set is expected to increase classification performance. Evaluating the relevance of the different attributes provided to the learning algorithms arises as future work.

Automating the process of classification has confirmed itself as a tough problem, requiring the accurate use of all sources of data. In this sense, the proposed process can benefit from integrating the information provided by both previously classified voxel models and the partial CFs. The former could be embedded in the system whilst the obtention of the latter requires the design of specific user interfaces which should be highly intuitive and usable.

Acknowledgements

This work has been funded by the spanish network IM3: Imagen Médica Molecular y Multimodalidad of the FIS.

References

1. J. Kniss, G. Kindlmann, and C. Hansen, "Interactive volume rendering using multi-dimensional transfer functions and direct manipulation widgets," in *Proceedings of the conference on Visualization 2001*, pp. 255–262, IEEE Press., 2001.
2. M. Hadwiger, C. Berger, and H. Hauser, "High-quality two-level volume rendering of segmented data sets on consumer graphics hardware," in *Visualization 2003*, pp. 40–45, IEEE-CSP, 2003.
3. R. Drebin, L. Carpenter, and P. Hanrahan, "Volume rendering," *ACM Computer Graphics*, vol. 22, pp. 65–74, August 1988.
4. H. Pfister, B. Lorensen, C. Baja, G. Kindlmann, W. Shroeder, L. Avila, K. Raghu, R. Machiraju, and J. Lee, "The transfer function bake-off," *IEEE Computer Graphics & Applications*, vol. 21, no. 3, pp. 16–22, 2001.
5. K. V. Leemput, F. Maes, D. Vandermeulen, and P. Suetens, "Automated model-based tissue classication of mr images of the brain," *IEEE Transactions on Medical Imaging*, vol. 18(10), pp. 897–908, 1999.
6. F. Tzeng, E. Lum, and K. Ma, "A novel interface for higher dimensional classification of volume data," in *Visualization 2003*, pp. 16–23, IEEE-CSP, 2003.
7. M. Ferré, A. Puig, and D. Tost, "A fast hierachical traversal strategy for multimodal visualization," *Visualization and Data Analysis 2004*, pp. 1–8, 2004.
8. F.-Y. Tzeng and K.-L. Ma, "A cluster-space visual interface for arbitrary dimensional classification of volume data," in *EG- IEEE TCVG Symposium on Visualization 2004, O. Deussen, C. Hansen, D.A. Keim, D. Saupe Eds.*, 2004.
9. G. Gerig, J. Martin, R. Kikinis, O. Kubler, M. Shenton, and F. Jolesz, "Unsupervised tissue type segmentation of 3-d dual-echo mr head data.," *Image and Vision Computing*, vol. 10, no. 6, pp. 349–36, 1992.
10. P. Domingos and M. Pazzani, "On the Optimality of the Simple Bayesian Classifier under Zero-One Loss," *Machine Learning*, vol. 29, pp. 103–130, 1997.
11. J. Quinlan, *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1992.
12. N. Landwehr, M. Hall, and E. Frank, "Logistic model trees.," in *ECML*, pp. 241–252, 2003.
13. Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," in *European Conference on Computational Learning Theory*, pp. 23–37, 1995.
14. E. L. Allwein, R. E. Schapire, and Y. Singer, "Reducing multiclass to binary: A unifying approach for margin classifiers," in *Proc. of ICML'00, 17th International Conference on Machine Learning*, pp. 9–16, Morgan Kaufmann, 2000.
15. "Internet Brain Segmentation Repository." http://www.cma.mgh.harvard.edu/ibsr.
16. I. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kauffman, second ed., 2005.

Coalition Formation in P2P File Sharing Systems

M.V. Belmonte, R. Conejo, M. Díaz, and J.L. Pérez-de-la-Cruz

E.T.S.I. Informática. Bulevar Louis Pasteur, N.35
Universidad de Málaga,
29071 Málaga (SPAIN)
{mavi, conejo, mdr, perez}@lcc.uma.es

Abstract. P2P file sharing systems are distributed systems consisting of interconnected nodes able to organize themselves in networks, with the purpose of sharing content. Recent empirical studies have shown that they suffer from freeloaders, that is, peers that consume many more resources or content than they contribute. In this paper we propose a coalition formation based incentive mechanism for P2P file sharing systems, that improves the system performance for the coalition participant peers. In addition, it discourages free-loader like behavior. The mechanism presents a formal approach to the problem based on game theory that takes into account the rational and self-interested behavior of the peers.

1 Introduction

P2P systems are distributed systems consisting of interconnected nodes able to organize themshelves in networks, with the purpose of sharing resources such as content, bandwidth, CPU cycles,... [1]. More specifically, P2P file sharing systems set up a network or pool of peers on Internet and provide facilities for searching and transferring files between them. Since these systems provide a cheap platform for data-sharing that is highly scalable and robust, a great number of commercial and academic projects have been developed using this technology [1]. However, the performance and availability of these systems relies on the voluntary participation of their users, and so they may be highly variable and unpredictable [3]. Recent studies have shown that a large proportion of the participants (20 to 40 % of Napster and almost 70% of Gnutella peers) share few or no files [5]. This phenomenon is known as free-loading: peers that consume more resources than they contribute.

One of the reasons for this problem is that the mechanisms used for downloading and sharing in the P2P file sharing systems do no take the selfish or self-interested behavior of the peers into account at the design stage. In fact, the P2P system's users act rationally trying to maximize the benefits obtained from using the system's shared resources [4]. Therefore, it will be necessary to find mechanisms that provide incentives and encourage cooperative behavior between the peers. Since they show a rational behavior, a possible solution could

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 153–162, 2006.

be to use an economic framework that provides them with incentives. Reviewing the bibliography we have found two main types of solutions that follow this approach: i) incentive mechanisms based on monetary payments: one party offering a service to another is remunerated and inversely, resources consumed must be remunerated or paid for. This category mainly included different micro payment mechanisms [4] [7], ii) incentive mechanisms based on differential services: peers that contribute more get a better quality of service. Reputation mechanisms belong to this category [3] [5]. In general, the peer's reputation reflects its overall contribution to the system. Our approach could be included in this category, although its foundation is different and innovative. We present a formal approach to the problem based on the game theory that takes into account the rational and self-interested behavior of the peers. The approach proposes the application of a coalition formation scheme based on game theory to P2P file sharing systems. The main idea of the coalition formation scheme is based on the fact that peers contributing more obtain a better quality of service. For that, we define an internal reputation value that reflects the peer's overall contribution to the system, and we use the game theory utility concept to calculate it. The coalition formation scheme will reward the peers with a higher internal reputation giving them greater bandwidth to download files, and will penalize the peers that only consume resources decreasing their internal reputation and consequently their bandwidth.

The paper is organized as follows: in the next section we briefly describe our coalition formation model. In the third section we describe the main characteristics and operation of P2P file sharing systems, and apply the above model to this type of systems. We will finish, in the fourth section, with some conclusions and future work.

2 Coalition Formation Model

Coalition formation is an important mechanism for cooperation in Multi-Agent Systems (MAS) [6]. Desired goals of a coalition formation process include: i) maximize the agents' profit or utility, ii) divide the total utility among agents in a fair and stable way, such that the agents in the coalition are not motivated to abandon it -stable payment configuration-, iii) do this within a reasonable time and using a reasonable amount of computational efforts. Our coalition formation model allows cooperation to take place among autonomous, rational and self-interested agents in a class of superadditive task oriented domains [2]. Each agent has the necessary resources to carry out its own task, however it is possible to form a coalition among agents with a new re-distribution of the task that may allow them to obtain benefits. The proposed model guarantees an optimum task allocation and a stable payoff division. Furthermore, computational complexity problems are solved.

Let us consider a set of n agents $N = \{a_1, a_2, \ldots, a_n\}$ that can communicate, and that each agent a_i must perform a certain initial amount t_i^o of task units (assumed infinitely divisible). Agent a_i can perform a maximum of k_i task units

($t_i^o \in k_i$) and the unitary cost is c_i. We will define the surplus capacity h_i of a_i by $h_i = k_i - t_i^o$ ($0 \leq h_i$). The agents can change the amounts of tasks initially assigned to each one, in order to achieve a better global efficiency; let $t_{i,j}$ be the number of task units transferred from a_i to a_j. However, to allow this new deal, they must incur in a transfer cost; namely, we will assume that the cost of transferring a task unit from a_i to a_j is $c_{i,j}$. In this way, the unitary profit, $b_{i,j}$, attached to the transfer of a task unit between a_i and a_j is $b_{i,j} = (c_i - c_j) - c_{i,j}$ and $B_{i,j} = b_{i,j} * t_{i,j}$ is the transfer profit. On the other hand, we will assume that coalition formation costs (communication and coordination) are negligible compared to actual task transfer costs. This assumption leads to a superadditive problem. The characteristic function or coalitional value, $v(S)$, of a game with a set of players N, is the total utility or profit that the members of S can reach by coordinating and acting together. In our domain, the coalitional value of any coalition will be stated as:

$$v(S) = \sum_{i,j \in S/i \neq j} B_{i,j} = \sum_{i,j \in S/i \neq j} b_{i,j} * t^*_{i,j}, \forall i, j \in S/i \neq j \tag{1}$$

where $t^*_{i,j} \forall i, j \in S/i \neq j$, are the task transferences obtained from an optimum task allocation T^*. In order to establish T^*, we state the task allocation as a linear programming problem, in which the costs are minimized subject to the capacity constraints of each agent [2]. If we impose the following constraint: $c_{i,j} + c_{j,k} \geq c_{i,k}$, there is always a solution in which any agent may be a producer or a consumer but not both. Producer agents are those that only transfer tasks and consumer agents are those that only receive tasks. Depending on the situation, there could be a fringe agent, a_f, a producer agent that does not transfer its entire task, or a consumer that receives task but does not consume its entire surplus. The payoff division is the process of dividing the utility or benefits of a coalition among its members. The problem is to distribute the utility of the coalition in a fair and stable way, so that the agents in the coalition are not motivated to abandon it. The game theory provides different concepts (core, kernel, Shapley value, etc.) for the stability of coalitions. Our proposed payoff division scheme is calculated by means of the marginal profit concept. Thus, the payment to each agent will be given by the marginal profit according to the resource supplied by the agent, and multiplied by the quantity of this resource. Since the concept of marginal profit is really that of partial derivative, the payment vector x, according to the formulae (1), will be computed as follows:

$$x_i = t_i \frac{\partial v}{\partial t_i} + h_i \frac{\partial v}{\partial h_i}, \forall i \in N \tag{2}$$

In [2] we prove that this payment vector computes in one step a stable payoff division in the sense of the core, and that it has a polynomial complexity in terms of n.

3 Coalition Formation in P2P File Sharing System

In general, a P2P content distribution system creates a distributed storage medium that allows the publishing, searching and retrieval of files by members of the network. We have selected for our work a P2P system with a partially centralized architecture and an unstructured network. Unstructured systems are adequate for highly transient peer populations and P2P file exchange systems fall into this category [1]. When a peer wants to download a file, it directs its request to a supernode and this searches the file in its index. In case of locating the file, the supernode sends to the "requester" peer an indexed result with the set of nodes that store the requested file. Then, the requester peer opens a direct connection with one or more peers that hold the requested file, and downloads it. Some of these systems [5] include a reputation index for the peers. The reputation is a global value that reflects the peer's overall contribution to the system, in terms of the upload bandwidth assigned to the file sharing system and the number and quality of the files shared by the peer. The file sharing system will reward the peers with a greater reputation giving higher priority to their request in file transfer queues. In order to simplify the model, we can think of the file sharing system as a black box that receives requests from the peers, sends them to the corresponding queue in terms of their reputations, and transfers "bytes" to "requester" peers in accordance with its reputation based scheduling algorithm.

In this section we propose the application of the above coalition formation scheme to P2P file exchange systems. We think that this scheme may improve the system performance and alleviates the free riding problem. The central idea is based on sharing the task of downloading a file among the agents forming a coalition. From the point of view of the peer that wants to download the file there is a clear advantage, since the task of transferring the file is divided between the members of the coalition, and the total download time is reduced.

In order to show how this reduction is achieved, let us consider the simplified scenario illustrated in figure 1. In this scenario, p_b, asks p_a for a file Z (producer in coalition formation terms). This peer forms a coalition with three other nodes (p_h, p_l and p_m, consumers in the coalition terms) to transfer that file. In P2P file sharing systems, every node has a download and upload bandwidth dedicated to file sharing. Usually this bandwidth is user defined and indicates a maximum value. The "real" bandwidth will vary on time depending on the computational and network charge for every node. Also it is common to define an upload bandwidth limit (k_{i-in}) which is much lower than the download one (k_{i-out}). Considering just the declared bandwidth is limitative. Indeed, often, the actual bandwidth that can be exploited is significantly lower. We will use as bandwidth value an estimation based on average values of past experiences. Let us suppose that the three peers have the file that has to be downloaded and that each of them dedicates its total upload capacity ($k_{h-in} + k_{l-in} + k_{m-in}$) to transfer a portion of the file. An estimation of a lower bound for the file transfer time is calculated in (3). Adding members to the coalition will reduce the time in the case where the addition of the upload bandwidths remains below the download bandwidth of p_b.

$$t = \frac{size(Z)}{(k_{h-in} + k_{l-in} + k_{m-in})}, k_{h-in} + k_{l-in} + k_{m-in} < k_{b-out} \quad (3)$$

The other aspect of using coalitions in this way that it motivates the peer that participates in the coalition to carry out the task (downloading files for other peers). Our scheme is based on providing a better quality of service to the peers that participate in the coalitions. In order to do so, we define an internal reputation value, Ir, for every peer. This value reflects the peer's overall contribution to the system (i.e. how much work it has carried out for the other peers in the system). In accordance with the above model, in the proposed payoff division each agent obtains a utility which is proportional to the resources that it supplies. Therefore, the agents that supply a greater bandwidth (resource) will obtain a greater utility, and this utility will increase its internal reputation. This internal reputation, Ir, must not be confused with the peer reputation in the global file sharing system (which we will call r in our model). This value is normally used to give a higher priority in the downloads to the peers that contribute with more files and bandwidth to the global P2P file sharing system. Our concept of Ir is similar, but this global reputation r is assigned and controlled by the P2P file system for a different purpose. It indicates the priority of that peer for downloading a file from other given nodes. When more than one peer is downloading files from a node, this node uses this priority to schedule the downloading of the different file pieces using a priority queue based scheduling algorithm.

3.1 Assumptions and Definitions

We treat each peer as a rational, autonomous and self-interested agent that tries to maximize its own utility by participating in the P2P system. We assume there are $N = \{p_1, p_2, \ldots, p_n\}$ peers in the system that can communicate among themselves and with a global file sharing system modeled as described above (fig. 1). We will use the following assumptions and definitions:

- t_i^o is the initial job that p_i must perform, i.e. the number of bytes (the size of the file) that p_i has to transfer (when another p_j downloads a file of size t_i from p_i). t_i^o is considered as a divisible job, since it is possible to transfer file blocks instead of a whole file. We will call t_i the task that the peer i will have to carry out after the task assignment in the coalition.
- k_i is the uploading capacity of p_i and is stated as $k_i = \min(k_{i-in}, k_{i-out})$. Where k_{i-in} is the reserved upload bandwidth (in bytes/sec) by p_i for transferring t_i inside the coalition formation system and k_{i-out} is the reserved download bandwidth. This download bandwidth will be used to download files from the P2P sharing file system in the case where p_i does not have the file previously and works for another node in the coalition (it would be 0 if the peer had the file) (fig.1). k_{i-out} also represents the maximum reserved bandwidth for downloading, when the peer i is the one that asks the system for the file. As we shown in the previous section, the size of the coalition will be limited by this value. The real upload capacity k_i will be limited by the

minimum of the internal (k_{i-in}) and the external (k_{i-out}) bandwidth. If the peer has to download the file from the global system, it cannot supply data at a greater rate than it receives from the P2P system. The same happens inside the coalition.

Since the reserved bandwidth is not a good indicator of the download speed in the system (as we said before, it also depends on other aspects such as reserved upload bandwidth, number of shared files, etc), k_{i-out} is multiplied by r_i, which is the reputation that p_i has in the file sharing system. The r_i value is in the interval $[0 \dots 1]$. In the case where it has the maximum value, $r_i = 1$, p_i may use all its reserved bandwidth and transfers the file very quickly, otherwise with $r_i = 0$, p_i does not transfer anything. The reason for this value is to maximize the processing capacity of the peer or its reserved bandwidth. So, if the peer has a low reputation it will be placed towards the end of the file transfer priority queues and it will not be able to complete its reserved bandwidth.

- c_i represents the cost of transferring t_i (its block of the file) and is the parameter that we try to minimize with our coalition formation scheme. In our case, this cost will be represented by the time that each node needs to upload its block of the file. In the coalition formation scheme presented in section 2, the optimum task assignment is computed by solving a linear programming problem in which we try to minimize the costs subject to the capacities of each node. In our case, these costs are not constant, since the capacity changes with time (the final download time will depend on the variations of the bandwidth, as previously discussed) and the problem cannot be solved in this way.

  Calculating the real values of these costs a priori is not possible, since we would have to know the total time consumed by each peer for transferring the amount of task assigned by the coalition formation algorithm. The upload and download bandwidths of the peers are only upper limits and cannot be used directly to calculate this time. As the real bandwidth varies with time depending on the node charge, the network and even the global file sharing system, we can only know this value when the global task is finished. However, we would need to know these values in order to partition the tasks among the different peers. In order to solve this we have adopted the following approach:

  1. We estimate a lower bound of the time needed to carry out the task as a function of the known bandwidth limits, corrected by the coalition internal reputation Ir (we explain how this value is calculated in the following section) and the global P2P reputation r. Following this, the lower bound for the cost(time) can be estimated as follows:

$$c_i = t_i^o / \sum_{i \in S} \min(Ir_i k_{i-in}, r_i k_{i-out}). \tag{4}$$

     Both capacities are multiplied by the reputation in order to obtain a better estimation. This estimated cost is the same initially for all the consumer peers in the coalition S.

2. Once we know the estimation of the cost/time for each peer, we can carry out a task assignment partitioning the file taking into account the capacities of each node. Every node will have to transfer a number of bytes $t_i = k_i * c_i$. The file is divided into blocks of this size that are assigned to the corresponding peers.
3. After completing the transfer, we will know the real values of the time consumed by all the peers (when the last byte from that peer arrives). These real values will be used in the payoff division process that will affect the Ir values for the next time, providing a better estimation of the time in the next coalition formation process.

– $c_{i,j}$ has a constant value, κ, for all the peers involved in the coalition formation system. κ reflects the cost of communicating p_i to p_j the quantity of task that have to be carried out, i.e. the file blocks that have to be transferred and the IP address of the peer that requests the download of the file.

3.2 Computing Internal Reputation

As we mentioned above the peers that participate in a coalition will "lend" bandwidth to other members of the coalition. So, the coalition formation system must "reward" the agents that participate in the coalition formation process and contribute to the system with their resources (bandwidth): "uploading" peers (peers participating in the coalition formation as producers or consumers), and "penalize" the agents that consume resources: "downloading" peers (peers that request to download a file from the P2P file sharing system). In this way, it will be possible to achieve parity between the resources that peers contribute and consume, and therefore avoid free-loaders in the P2P file sharing system. Since in our model the utility obtained by each agent is proportional to the resources that it supplies, the value of Ir_i will be calculated as a heuristic function of x_i that can be adjusted with data from the real system behavior or from simulation results. As mentioned above, x_i is calculated according to (2) and using the real values of c_i (time consumed by the peer to transfer its corresponding block of file). The value of Ir_i should be reduced when p_i acts as a downloading peer and does not contribute to other coalitions, and incremented when it provides files to the other peers (uploading peer).

The internal reputation of p_i, Ir_i, reflects the resources contributed by p_i to the coalition formation system. Ir_i is a value included in the interval $[0 \dots 1]$. This value is used to correct both, the download and upload bandwidths of each peer. In this way, we obtain a better estimation of the real bandwidths, avoiding consider just the defined bandwidths. The correction for the upload bandwidth reflects its possibility of filling the reserved upload bandwidth k_{i-in} when p_i is participating in a coalition and has been taken into account in the capacity calculation in the previous section. The correction of download bandwidth ($Ir_i k_{i-out}$) is achieved in order to reward or penalize a peer that wants to download a file. A higher internal reputation (Ir_i closer to 1) will mean that p_i will be able to fill all its reserved bandwidth, since it could add more peers to the coalition in order to complete its bandwidth, reducing the download time. Otherwise, an Ir_i

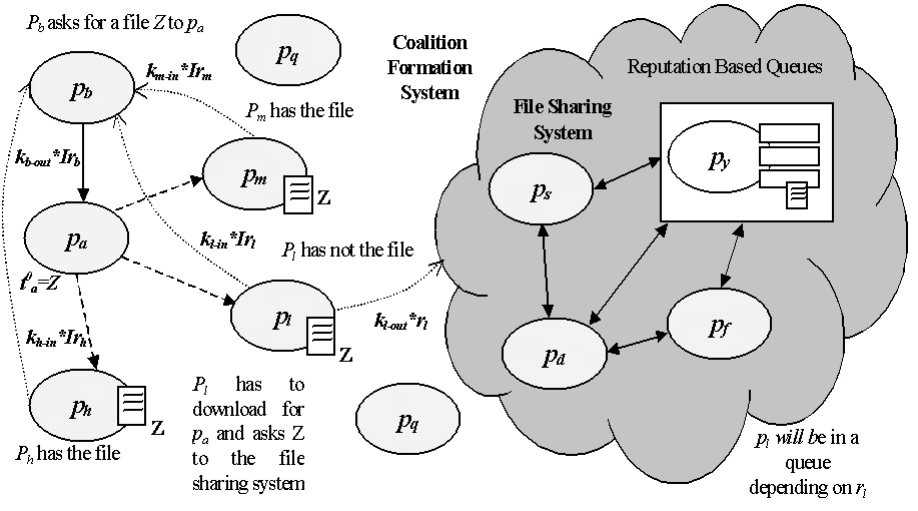


Fig. 1. Coalition Formation in P2P File Sharing System

closer to 0 will limit the possibility of adding peers to the coalition (in fact, in some cases it will avoid creating any coalition for the download). The size of the coalition S will always be limited by the following expression:

$$\sum_{s \in S} \min(Ir_s k_{s-in}, r_s k_{s-out}) < Ir_i k_{i-out}. \tag{5}$$

In order to calculate Ir_i for the uploading peers, their utility values x_i (obtained from their participation in the coalition formation process) are normalized in the interval $[0 \ldots 1]$, obtaining δ_i; and $Ir_i = Ir_i^o * (1 + \delta_i)$ where Ir_i^o is the initial value of the internal reputation of p_i (we will suppose initially $Ir_i^o = 0.5$ for all the agents registered in the coalition formation system). For the downloading peer, if n is the set of uploading peers in S, Ir_i^o is multiplied by 1 minus the normalized average utility δ_j obtained by the uploading peers, $j \in n$, $Ir_i = Ir_i^o * (1 - (\sum_{j \in n} \delta_j / \mid n \mid))$. Then, all the calculated Ir_is are sent to the coalition formation system. Since the $Ir_i s$ must be values included in the interval $[0 \ldots 1]$, all the internal reputation indexes are again normalized. However, this time the operation will affect all the peers registered in the coalition formation system and not only the peers belonging to the actual coalition formation process. In this way, the Ir of the peers only registered will also be decreased in order to "penalize" their passive behavior.

3.3 An Example

Let us suppose that the peers p_a, p_b, p_m, p_h and p_l, among others, are registered in the coalition formation system. Previously, in the file sharing system p_b has sent a request to p_a to download a file Z. So, p_a has to transfer $t_a^o = 600$MB

(the size of the file) to p_b corresponding to the file Z (fig. 1). In order to carry out the transfer, p_a decides to initiate a coalition formation process. First, p_a communicates its intention to the coalition formation system (the coalition formation system provides a yellow pages service to the peers interested in this service, and in addition stores the reputation values of the peers with the goal of avoiding subversion by "malicious" peers [2]). Then, the system communicates to p_a the IP addresses, r, and I_r values of a set of interested peers that satisfy the constraint (5). Let us suppose that these peers are p_m, p_h and p_l (consumers in coalitional terms). Then, p_a (producer) requests their k_{i-in} and k_{i-out} values from these nodes in order to calculate the task allocation. In table 1 we show the values of all the parameters for this coalition formation process.

Next, and according to (4) p_a calculates the estimated lower bound for the cost, c_i (time). If we suppose that p_m and p_h have the file and p_l has to download it from the global P2P file sharing system (fig. 1), $c_i = 600/(5 * 0.5 + \min(10 * 0.5, 128 * 0.5) + 15 * 0.5) = 40$. Therefore, 40 sec. is the lower bound for the time estimated to transfer the file between p_m, p_l, and p_h. With this value we can calculate the number of bytes that each peer must transfer ($t_i = k_i * c_i$): $t_m = 100$MB, $t_l = 200$MB and $t_h = 300$MB. p_a divides the file into blocks of these sizes and communicates to each peer which block it has to transfer and the IP address of p_b. Then, each peer transfers its block to p_b.

Next, p_a calculates the stable payment division x for the uploading peers according to (2). For this, the real values of time consumed by each peer are used, $c_m = 50$, $c_l = 42$ and $c_h = 41$ seconds. In order to simplify the calculation we suppose $\kappa = 0$. If $c_{i,j} = 0$, x may be calculated in the following way [2]: $x_i = t_i(c_i - c_f)$ if p_i is a producer peer, $x_i = h_i(c_f - c_i)$ if p_i is a consumer and $x_i = 0$ if $p_i = p_f$, where p_f is the fringe peer (in this approach the fringe peer will be the consumer peer with the largest c_i, in the example p_m). So, $x_a = 600*(60-50)$ ($c_a = 60$, since $t_a = 600$ and $k_a = 20*0.5$), $x_h = 300*(50-41)$, $x_l = 200 * (50 - 42)$ and $x_m = 0$. Finally, p_a communicates this information to the coalition formation system. With these values the coalition formation system updates the internal reputation of the peers involved in the coalition formation process, according to the heuristic functions described in 3.2. The resulting Ir values are shown in table 1. Finally, the Irs of all the registered peers in the coalition formation system are normalized again. These updated values will be the used by the system in the next coalition formation process.

Table 1. Coalition formation process parameters

parameters	p_a	p_b	p_m	p_l	p_h
k_{i-in}	20	10	5	10	15
k_{i-out}	128	128	128	128	128
t_i^o	600	-	0	0	0
t_i	0	-	100	200	300
r_i	0.5	0.5	0.5	0.5	0.5
Ir_i^o	0.5	0.5	0.5	0.5	0.5
Ir_i	1	0.215	0.5	0.63	0.725

4 Conclusions and Future Work

In this paper we have proposed a coalition formation based incentive mechanism for P2P file sharing systems, with the purpose of discouraging freeloader-like behaviour and increasing overall performance of these systems. This mechanism presents a formal approach to the problem based on game theory, taking into account the rational and self-interested behaviour of the peers, and we have shown that the approach is feasible. Wang et al. also [8] use coalition formation in P2P file sharing systems. However, unlike our model, this mechanism uses a heuristic approach. It proposes a mechanism for forming trust-based "communities" of peers. Following this approach, an interesting extension of our work could also be that of generating coalitions by grouping sets of allied peers that have similar characteristics (contents, near locations, bandwidth,...). In order to evaluate our approach, we are now designing and developing a simulation model to study different patterns of behaviour among peers in a file sharing system, and the impact of the coalition formation on the free-loaders peers. In addition, we plan to compare experimental results of this mechanism with results of other more common incentive-based mechanisms as micro payment. If the final experimental results are successful, it would be possible to use the proposed coalition formation scheme as a plug-in on some popular P2P file sharing systems such as eMule or kazaa.

References

1. Androutsellis-Theotokis, S., Spinellis, D.: A survey of Peer-to-Peer Content Distributing Technologies. ACM Computing Surveys, 36, No.4 (2004) 335-371.
2. Belmonte, M.V., Conejo, R., Perez-de-la-Cruz, J.L., Triguero,F.: Coalitions among Intelligent Agents: A Tractable Case. Computational Intelligence. An International Journal, 22-1 (2006), 52–68.
3. Buragohain, C.,Agrawal, D., Suri, S.: A Game Theoretic Framework for incentives in P2P Systems. Proceedings of the 3th International Conference on Peer-to-Peer Computing (2003)
4. Golle, P., Leyton-Brown, K., Mironov, I., Lillibridge, M.: Incentives for Sharing in Peer-to-Peer Networks. In Proceedings of the ACM Conference on Electronic Commece (2001).
5. Ramaswamy, L., Liu, L.: Free Riding: A New Challenge to Peer-to-Peer File Sharing Systems. In Proc. of the 36th Hawaii International Conference on System Sciences (2003).
6. Sandholm, T.: Distributed Rational Decision Making. In Multi-agent Systems. A modern Approach to Distributed Artificial Intelligence. MIT Press, (1999) 201-258.
7. Vishnumurthy, V., Chandrakumar, S., Gn Sirer,E.: KARMA: A Secure Economic Framework for Peer-to-Peer Resource Sharing. In Proceedings of the Workshop on the Economics of Peer-to-Peer Systems, Berkely, California (2003).
8. Wang, Y.: Trust-Based Community Formation in Peer-to-Peer File Sharing Network. In Proc. of IEEE/WIC/ACM International Conference on Web Intelligence, WI-2004 (2004)

Combining Human Perception and Geometric Restrictions for Automatic Pedestrian Detection

M. Castrillón-Santana[1] and Q. C. Vuong[2]

[1] IUSIANI
Edificio Central del Parque Científico-Tecnológico
Campus Universitario de Tafira
Universidad de Las Palmas de Gran Canaria
35017 Las Palmas - Spain
mcastrillon@mcastrillon.ulpgc.es
[2] Max Planck Institute for Biological Cybernetics
Cognitive & Computational Psychophysics
Spemannstrasse 38
72076 Tübingen, Germany
quoc.vuong@tuebingen.mpg.de

Abstract. Automatic detection systems do not perform as well as human observers, even on simple detection tasks. A potential solution to this problem is training vision systems on appropriate regions of interests (ROIs), in contrast to training on predefined and arbitrarily selected regions. Here we focus on detecting pedestrians in static scenes. Our aim is to answer the following question: *Can automatic vision systems for pedestrian detection be improved by training them on perceptually-defined ROIs?*

1 Introduction

The present study investigates the detection of pedestrians by humans and by computer vision systems. This simple task is accomplished easily and quickly by human observers but still poses a challenge for current vision systems.

In the Computer Vision community, different automatic detection systems have been designed in the past using simple features for people detection based on the detection of different body elements: the face [4,17], the head [1,2], the entire body [15] or just the legs [11], as well as the human skin [6].

These systems make use of some selected regions where representations based on local features [9,14], sometimes combined with global cues [7], are employed for detection. Such systems perform fairly well but still have high miss rates. In order to overcome this problem, more recently a combination of body parts have been used to improve the performance as the false positive for an individual detector is higher than for several detectors [10].

However, the criteria to select the different body parts or regions have not been the focus in these earlier works. Rather, the parts or regions have been chosen *ad hoc* or arbitrarily. That said, we have observed a small correlation between

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 163–170, 2006.

the performance of these systems and human observers. This finding motivated us to systematically analyze human performance on a pedestrian detection task that tests whether these regions are the most semantically useful and whether other regions can also provide useful information. This study therefore allows us to determine which body parts should be included in an automatic detection system.

For that purpose, we used a psychophysical "bubbles" technique [3], described in Section 2, to isolate those regions used by humans for pedestrian detection– what we call perceptually-defined regions of interest (p-ROI). Section 3 describes the general object detection framework designed by Viola and Jones [14] used to train a p-ROI detector. Section 4 presents the geometric restrictions employed to make use of the global configuration of these parts, with the aim to reduce false detections. Results are presented in Section 5, and some conclusions are outlined in Section 6.

2 The Bubbles Technique

To investigate which ROIs are used more by humans, we used a psychophysical "bubbles" technique [3] to isolate the regions which help human observers determine the presence of pedestrians in an image. The technique was originally used in [3] to identify internal facial features that provided diagnostic information for gender and expression classification. For example, with high-resolution face images, the gender was correctly determined using just the eyes and mouth.

In the current study, images containing aligned pedestrians were revealed through a mask of small randomly distributed Gaussian windows ("bubbles"). That is, the presence of a bubble over a region showed that region, as shown in Figures 1 and 2. Eight subjects were shown stimuli masked by Gaussian bubbles and had to judge if a human was present. Half the trials contained a human. Across observers, masks leading to correct responses are summed and normalized to reveal image regions that were useful for this task. This procedure is illustrated in Figure 1. The result of the bubbles paradigm is a diagnostic image, which is presented in Figure 2.

The diagnostic image (Figure 2) indicates that observers relied predominantly on head and leg regions, and to a lesser extent on arm regions. These results confirm some of the regions already considered by automatic pedestrian detectors.

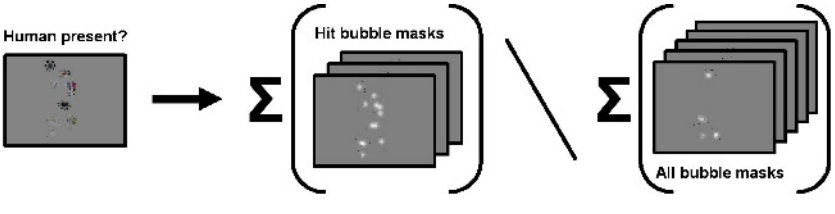


Fig. 1. Building the diagnostic image using bubbles

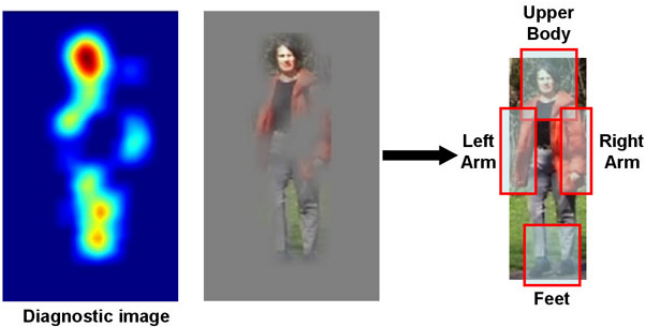


Fig. 2. The diagnostic image produced from eight human observers for the pedestrian detection problem

3 Viola-Jones General Object Detection Framework

The Viola-Jones [14] general object detection framework achieves fast and robust performance by means of a cascade of weak classifiers trained using boosting. This approach has been used to train the p-ROI detector in order to confirm if those regions are particularly discriminable for certain pattern matching problems, and to what extent an area is important. This framework has already been successfully applied to different objects categories, and it is well known in the face detection community because it provides real-time performance.

Recently, an implementation has been made available in OpenCV (Open Computer Vision Library) [5]. This framework, designed for rapid object detection, is based on the idea of a boosted cascade of weak classifiers [13] but extends the original feature set and provides different boosting variants for learning [8].

The cascade learning algorithm is similar to decision-tree learning. Essentially, a classifier cascade can be seen as a degenerated decision tree. For each stage in the cascade a separate subclassifier is trained to detect almost all target objects while rejecting a certain fraction of the non-object patterns. The resulting true detection rate, D, and the false detection rate, F, of the cascade is given by the combination of each single stage classifier rates:

$$D = \prod_{i=1}^{K} d_i \qquad\qquad F = \prod_{i=1}^{K} f_i \tag{1}$$

For example, given a 20-stage cascade of weak classifiers designed to reject 50% of non-object patterns (false detection rate) while accepting 99.9% of object patterns (detection rate) at each stage, the overall detection rate will be $0.999^{20} \approx 0.98$ with a false detection rate of $0.5^{20} \approx 0.9 * 10^{-6}$. This scheme allows for a high image processing rate, due to the fact that background regions of the image are quickly discarded. Consequently, more processing time can be dedicated to promising object-like regions. Thus, to achieve a desired trade-off

between accuracy and speed for the resulting classifier, the designer can choose the desired number of stages, the target false detection rate and the target true detection rate per stage.

4 Geometric Restrictions

As stated in recent works, the probability of a false detection for an individual detector is higher than for several detectors [10]. Using this assumption, the coocurrence of coherently located detections can provide the system with evidence to reject detections which are inconsistent with the majority of detections. This fact is evident in Figure 3. The left column reflects the hits obtained for a frame using the p-ROI detector, while the right column indicates the resulting filtered detection image after applying some heuristic geometric restrictions for typical standing pedestrians. The basic rules applied to determine if a detection is coherent in the image are applied only when at least two different detectors of different nature reported a hit. These rules are summarized as:

- The feet must be below the head, to the right of the left arm and to the left of the right arm.
- The head must be above the feet, to the right of the left arm and to the left of the right arm.
- The centroid of the left arm must be to the left of head, feet and right arm.
- The centroid of the right arm must be to the right of head, feet and left arm.

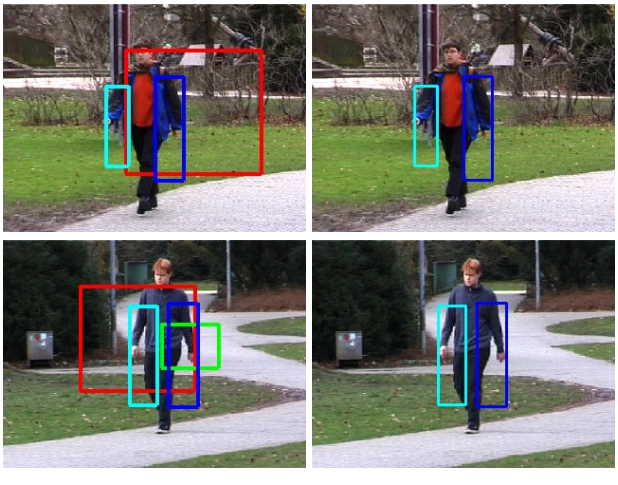


Fig. 3. Left column shows detection samples using the p-ROIs detector. Right column shows hits accepted after geometric filtering. Colors are related to the detector: red for heads and shoulders, green for feet, blue for right (in the image) arm and cyan for left (in the image).

5 Experiments

The detectors used in these experiments were trained using the OpenCV implementation of the Viola-Jones framework. The training set consisted of the CBCL pedestrian images [12] augmented with additional positive and negative samples

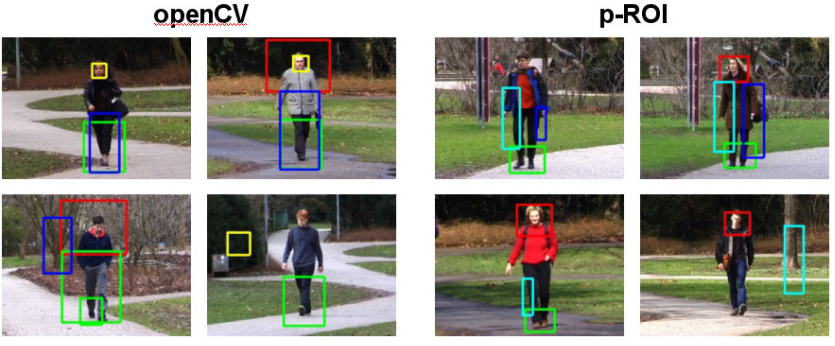


Fig. 4. Results achieved using the detectors included in the OpenCV release, and results achieved using the basic p-ROI approach. Colors meaning: left) yellow for faces, red for heads and shoulders, green for lower body, blue for full body , and right) red for heads and shoulders, green for feet, blue for right (in the image) arm and cyan for left (in the image).

Table 1. Default OpenCV detectors results. Double detections refers to overlapping detections of the same target, true detections consider multiple correct detections of the same target as a single one.

	Total Detections	Double Detections	True Detections	False Detections	d'
Face	248	1	170	77	0.42
Upper Body	136	0	134	2	1.66
Lower Body	711	132	505	74	1.23
Full Body	213	3	69	41	0.24
Total			14.5%	3.5%	0.89

Table 2. p-ROI detection results

	Total Detections	Double Detections	True Detections	False Detections	d'
Upper Body	1166	56	984 (66%)	126 (8%)	1.79
Feet	909	89	636 (43%)	184 (12%)	0.97
Left Arm	616	81	407 (27%)	128 (9%)	0.76
Right Arm	601	70	353 (24%)	178 (12%)	0.46
Total			40%	10%	

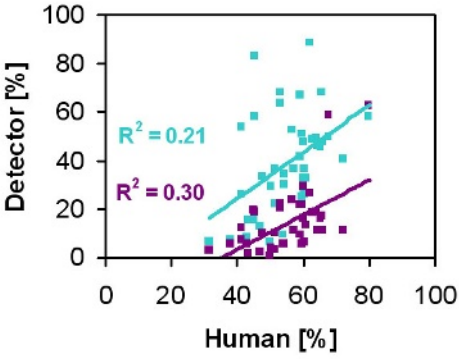


Fig. 5. For each of the 41 pedestrian test video sequences, we observed a correlation between how well human observers performed in a pedestrian detection task and how well the openCV (purple) and p-ROI (cyan) detectors on a detection task. Each dot represents one of the pedestrians. The human data are from [16].

of upper bodies and faces for the default openCV detectors and (a different set of) upper bodies for the p-ROI detectors. The test consisted of image sequences of 41 pedestrians walking through a park [16]. In total there were 1448 images.

Figure 4 illustrates different detection results achieved with the two detectors. Tables 1 and 2 compared the results achieved using the default detectors included in the OpenCV release, and the results achieved with the basic p-ROI detector. As evident in the results tables, the true detection rate is increased with the new approach, however, the false detection rate is also increased. As raised above, we also found a small correlation between both detectors' true detection rate with human detection performance, as shown in Figure 5.

In order to test possible benefits of the geometric restrictions, the hits achieved by the p-ROI detector have been filtered using the geometric coherence tests. The results summarized in Table 3 reflect an important reduction of false detection rates together with a slight reduction in true detection rates. Specifically, false detection are reduced by $\sim 50\%$ (averaging across the individual detectors) whereas true detections are reduced by $\sim 8\%$.

Table 3. p-ROI with geometric restrictions detection results

	Total Detections	Double Detections	True Detections	False Detections	d'
Upper Body	1062	38	963 (65%)	61 (4%)	2.12
Feet	675	47	547 (37%)	81 (5%)	1.27
Left Arm	507	60	373 (25%)	74 (5%)	0.98
Right Arm	464	43	315 (21%)	106 (7%)	0.67
Total			37%	5%	

6 Conclusions

We have described a pedestrian detector which is based on body parts selected according to their perceptual importance. The results achieved with this set of features improved hits for automatic detection of pedestrian compared with the default detectors included in OpenCV. The method provides a means to select training features to improve automatic vision systems.

Occlusions are not considered explicitly, but the use of a body part approach allows occlusions up to a certain level due to the fact that the system requires only the presence of a single part.

In sum, we believe that this perceptually-based approach combined with simple rules (i.e., our geometric constraints) can be useful for assigning different weights to regions and points not only based on their discrimination features but also on their perceptual significance for the problem considered.

Future work will consider the integration of temporal coherence to improve the detection rate and reduce the false detections when a sequence is available. The application to crowded scenes will also be a challenging problem.

Acknowledgments

Work partially funded by research projects Univ. of Las Palmas de Gran Canaria UNI2003/06, UNI2004/10 and UNI2004/25, Canary Islands Autonomous Government PI2003/160 and PI2003/165 and the Spanish Ministry of Education and Science and FEDER funds (TIN2004-07087).

References

1. Stan Birchfield. Elliptical head tracking using intensity gradients and color histograms. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 232–237, June 1998.
2. Marco La Cascia and Stan Sclaroff. Fast, reliable head tracking under varying illumination: An approach based on registration of texture-mapped 3d models. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(4):322–336, April 2000.
3. F. Gosselin and P. G. Schyns. Bubbles: a technique to reveal the use of information in recognition tasks. *Vision Research*, pages 2261–2271, 2001.
4. Erik Hjelmas and Boon Kee Low. Face detection: A survey. *Computer Vision and Image Understanding*, 83(3):236–274, 2001.
5. Intel. Intel Open Source Computer Vision Library, b4.0. www.intel.com/research/mrl/research/opencv, August 2004.
6. Michael J. Jones and James M. Rehg. Statistical color models with application to skin detection. Technical Report Series CRL 98/11, Cambridge Research Laboratory, December 1998.
7. B. Leibe, E. Seemann, and B. Schiele. Pedestrian detection in crowded scenes. In *Proceedings of the CVPR'05)*, 2005.

8. Rainer Lienhart, Alexander Kuranov, and Vadim Pisarevsky. Empirical analysis of detection cascades of boosted classifiers for rapid object detection. In *DAGM'03*, pages 297–304, Magdeburg, Germany, September 2003.
9. D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Comnputer Vision*, 60(2):91–110, 2004.
10. Krystian Mikolajczyk, Cordelia Schmid, and Andrew Zisserman. Human detection based on a probabilistic assembly of robust part detectors. In *European Conference on Computer Vision*, volume I, pages 69–81, 2004.
11. C. Papageorgiou, M. Oren, and T. Poggio. A general framework for object detection. In *Proceedings of the International Conference on Computer Vision*, pages 555–562, 1998.
12. C. Papageorgiou and T. Poggio. A trainable system for object detection. *International Journal of Computer Vision*, 38(1), 2000.
13. Paul Viola and Michael J. Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition*, pages 511–518, 2001.
14. Paul Viola and Michael J. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57(2):151–173, May 2004.
15. Paul Viola, Michael J. Jones, and Daniel Snow. Detecting pedestrians using patterns of motion and appearance. In *Proc. of the International Conference on Computer Vision*, volume 2, pages 734–741, October 2003.
16. Q. C. Vuong, A. Hof, H. H. Bülthoff, and I. M. Thornton. An advantage for detecting human targets in dynamic versus static composite stimuli. In *4th annual meeting of the Vision Sciences Society*, 2004.
17. Ming-Hsuan Yang, David Kriegman, and Narendra Ahuja. Detecting faces in images: A survey. *Transactions on Pattern Analysis and Machine Intelligence*, 24(1):34–58, 2002.

Comparative Analysis of Artificial Neural Network Training Methods for Inverse Kinematics Learning

Juan Pereda, Javier de Lope, and Darío Maravall

Perception for Computers and Robots Group
Universidad Politécnica de Madrid
Campus de Montegancedo, 28660 Madrid
{jpgranados, jdlope, dmaravall}@dia.fi.upm.es

Abstract. A method for obtaining an approximate solution to the inverse kinematics of a articulated chain is proposed in this paper. Specifically, the method is applied to determine the joint positions of a humanoid robot in locomotion tasks, defining the successive stable robot configurations needed to achieve the final foot position in each step. Our approach is based on the postural scheme method, using artificial neural networks to solve the problem. In this paper we define the restrictions that must be accomplished by the networks and make an exhaustive study about the learning algorithms, transfer functions, training sets composition, data normalization and artificial neural network topologies.

1 Introduction

Artificial neural networks (ANN) have been successfully used for solving problems of a considerable difficulty. The most commonly mentioned are in fields such as pattern recognition, image analysis, natural language processing, etc. ANN have been also applied in robotics to control the robot actuators or to filter sensor information, for example, and also to provide approximate solutions to problems in which an analytic one is unavailable or complex. One of these cases is the resolution of the inverse kinematics of articulated joins using neuronal methods [1,2].

A humanoid robot can be modeled by means of one or more articulated chains. Our robot model [3] is composed by its lower limbs and a mass over the hip that represents the rest of the body. We model the lower limbs of the robot by means of a framework systems chain, associating the base frame to the fixed foot, i.e. the foot that stays on the ground, and the final frame to the free foot which describes the movement. Each leg contains six joints. Two joints are for the ankle movements, one for the knee, and the rest of joints are for the hip movements.

Usually two methods are used for humanoid robot walking. The *static balance method* uses the center of masses projection to maintain the robot stability, the *dynamic balance method* uses the ZMP (Zero Moment Point) to achieve the

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 171–179, 2006.

stability [4]. Basically, both methods apply corrections to the joint positions to maintain the respective index in robot support polygon. Our postural scheme method proposes the use of previously defined joint configurations which are safes, therefore, it is not strictly necessary to apply the corrections in this case. Then, the problem is reduced to find a "system", on one hand, that allows store efficiently the aprioristic information about the stability for a given robot configuration, and, on the other, that can generalize a set of basic generated data.

In this paper we propose the use of ANN to store the knowledge on the robot stability. One of our objectives is to select the type of networks with the lowest resource requirements due to the best trained networks are implemented on a FPGA which is in charge of to make the robot movements. We show the different training methods and networks obtained. Finally the defined criteria to select the best fit network for the problem resolution are discussed.

2 Postural Scheme Method

The postural scheme method allows to determine the values which must be assigned to each humanoid robot joint, in general, a multi-joint mobile robot, to achieve a particular position during its movement, maintaining the robot stability [5,6]. It is based on the definition of several sets of joint and Cartesian coordinates pairs that define different configurations or *postures*, stables all of them, related to a specific type of movement.

As an example, we have defined several schemes [6]. The aim of the first postural scheme (Fig.1a) is to determine a subset of positions in which the robot can stay in such a way it maintains both feet in the ground but with a displacement in the frontal plane —opening or closing the legs— and also in the saggital plane —moving forward or backward a leg with respect to the other—. The second postural scheme (Fig.1b) integrates the poses that the robot could use during a single step in a forward or backward movement.

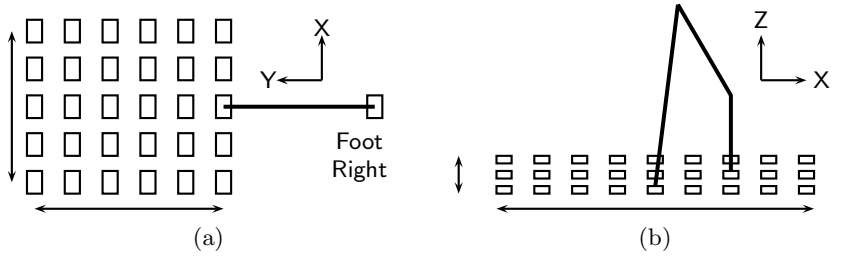


Fig. 1. Postural schemes: (a) both feet in the ground and (b) for step execution

Every scheme is made up of pairs which relate the robot joint coordinates, i.e. the position of each joint, and the free foot Cartesian coordinates, i.e. position and orientation, with respect to some framework system. Considering the robot as a articulated chain, we get the Cartesian coordinates from a specific joint

coordinates by means of the direct kinematics which, in the general case, is easily calculated.

To generate the robot movements it is also interesting the inverse procedure. Thus we want to determine the position in which the joints must be moved, i.e. the joint coordinates, to achieve a specific relative position between the feet, in general to achieve a the posture, during a movement execution, for example, a step. Specifically we are interested in the inverse kinematic of the articulated chain.

An analytical method to find the inverse kinematics of articulated chains is more difficult to solve than the corresponding direct kinematics. Therefore a technique to find an approximate solution is well suited in this case. Moreover, our approach allows remove solutions or configurations that can leave to the robot in an unstable posture or filter multiple solutions due to the joint redundancy with parallel rotation axes.

Thus, we can generate sets of joint and Cartesian coordinates pairs in which are only included pairs associated to stable postures in the robot. To generate the sets in such a way we have to apply a set of stability criteria and some restrictions to ensure that the robot does not fall down and achieves natural postures [6].

Finally, the coordinate pairs sets can be used for training artificial neural networks. Thus each ANN that achieve learn and generalize those values, can be used to solve the robot's inverse kinematics for an specific conditions. Such conditions will be determined by the used postural scheme.

3 Basic Considerations on ANN

Previously to comment the different topologies that we have used in the artificial neural networks, it is needed to make several remarks on some design decisions.

First, we have to remember that we are working with ANN whose goal is to be implemented in hardware. Thus, it is compulsory to maintain as a constant the number of hidden layers of the ANNs, one in our case. For this reason we also try to minimize the number of neurons in the input, hidden and output layer. A change in the topology of a software network is a change of one or more of the control program parameters. In hardware, this change implies, probably, to modify the hardware architecture, adding more bits or inputs or outputs in the FPGA, rewrite the VHDL or block program, simulate the whole system in order to verify the correct operation and, finally, to load the program in the FPGA and to verify again that it works correctly. We are developing a ANN on a FPGA library [7] which includes several topologies, allowing network modularity and even to execute a same network over several FPGA. The main advantage of this library is the optimization of the previously described procedure, automatizing the creation of a hardware network from a software network specification.

We can also make some remarks on the number of inputs. As we have fixed the feet orientation during the training set construction, making that the feet are always parallel to the ground, we can remove the three angles needed to define a orientation in the 3D space. Therefore, in the general case, we will use

ANN with three input neurons, associating each neuron to a coordinate in a tridimensional space. As we will show below, there are some cases in which one of the coordinates is a constant —for example, when the robot is walking on a straight line, we can consider that the free foot does not make any lateral displacement— some topologies with two input neurons are also tested.

The ANN outputs are associated to every joint position and, as we mentioned above, our model robot has twelve joints. Some similarities in the values of some joints can be discovered analyzing the robot structure and preprocessing the data before its use as a training set. These similarities are produced by the symmetrical position of some joints or because some joints are not used in some movements, keeping its values as a constant. Some optimizations in the ANN depends on this preprocessing phase, thus, the number of output neurons will also depend on the way is preprocessed and solved the problem. Therefore, we will justify each solution independently in the next sections.

We have used four different learning algorithms for the ANN training. The *flexible backpropagation* is a variation of the classical *backpropagation* that tries to fix the slow convergence problem when the gradient is small. The *BFGS* is a so-called Quasi-Newton method which tends to converge with a lower number of iterations than other methods based on the gradient although it requires more execution time and more storage space per iteration than the others. The *One-Step-Secant* method reduces the execution time and the storage space per iteration compared to the previous method. The *Levenberg-Marquardt* method also uses a few of iterations for the convergence although the execution time and storage per iteration is greater than for the *backpropagation*.

We have selected these four methods for the final experimentation due to the fact that they offered better results than other training methods tested in a previous, partial experiments. Moreover, we want to maintain for comparing algorithms well suited for function approximation —as the BFGS and the Levenberg-Marquardt—, for classifying —*flexible backpropagation*— or general purpose algorithms —the *One-Step-Secant* method—. Thus we can also verify if the use of a method family has an effect on the result generalization.

We have used two types of sigmoid transfer functions for the hidden layer: logarithmic functions, similar to $f(x) = \frac{1}{1+e^{-x}}$, and tangential functions, $f(x) = \frac{e^n - e^{-n}}{1+e^{-n}}$. The output layer has always associated a linear function.

We have used three different training sets. Each set corresponds to the same postural scheme and the same workspace, the difference is the generated points density: a greater density implies a greater number of examples in the training set. The sets are composed by 31, 61 and 161 examples. The aim is to test the effect of the density in the learning and generalization.

Finally, we have used the training sets in three different ways. The first one is directly created, without any preprocessing. The set includes the ranges of each joint and coordinate generated by means of the direct kinematics equations. The second one processes the values dividing by 100. Basically, this was made because the hardware implementation requires a lower number of bits to represent the entire part. In the third one the values are normalized in the interval $[-1, 1]$.

4 Network Topologies

As we have previously commented, several topologies are been defined, looking for the best suited network with the lower number of elements. Generally this is a desirable premise during network design phase and a must in our case due to after the learning the network have to be implemented on a FPGA and the available area in this kind of devices imposes an important restriction on the size of the program.

The number of hidden layers will be fixed using one hidden layer for all the tests, but we must calculate the number of neurons for this layer in the experiments of each learning method, maintaining this value as lower as possible.

The first topology could be named as *monolithic*. We use only a network with three input neurons, each one associated to a coordinate in the tridimensional space which correspond to the free foot position with respect to the fixed foot, and twelve output neurons, one neuron for each angle associated to each robot joint.

Using a only network implies in our case that the whole number of neurons is lightly high due to the twelve neurons needed for the output layer, having to add the required ones for the hidden layer. On the other hand, this type of implementation implies a requirements on the FPGA area which are very difficult to achieve in small/medium size devices.

Therefore, we have also defined networks with a lower number of neurons in the output layer, associating each network to a robot part. Specifically, we have established two cases. In the former the network is composed by *two modules* and each module is in charge of determining the positions of every joint in a leg. Thus, a module has three neurons in the input layer and six neurons in the output layer.

In the second case, the network is composed by *three modules*. The first module is in charge of the four hip joints and the others are for the four joints in each leg. Thus, this type of networks have three neurons in the input layer and four neurons in the output layer.

As we mentioned in the previous section, the number of neurons can be reduced by *eliminating neurons* in the input and output layers, analyzing the data and the robot structure. Specifically, for the used postural schemes, linear dependencies between different joints can be established in such a way that the information to be learned by the network results redundant. Also, some input coordinates maintain a constant value (or *approximately constant*) for each example. Therefore, although it is a particular case, we have defined a type of networks which respond to these characteristics. Such networks have three neurons in the input layer and seven in the output layer and also two neurons in the input layer and twelve in the output layer.

5 Error Measurements

Some error measurements have been analyzed to select the best type of network. To obtain the errors, we have taken the training network and simulated with

some test values that contain all the correct positions where the foot can stay, obtaining the robot joint coordinates. These coordinates are introduced into the direct kinematics equations, obtaining the positions where must be the foot stay. If all the process is correct and the network is perfectly trained, this position will be equals to first one. Using the initial positions (x_i, y_i, z_i) that express the desired one and the final positions (x'_i, y'_i, z'_i) that are the one obtained using the process, we have obtained the following measurements:

- *Mean Squared Error*. We consider the mean squared error in the model like the average of the differences of the square between the observations and the values that predict the model.
- *Mean Squared Error per Coordinate*. It consists of the mean squared error in all separated Cartesian coordinates.
- *Maximum Distance*. It is the maximum Euclidian distance between the initial and final points. This error is very important because if only one point has a high error, the robot can lost steady state and fall down. Moreover, if the maximum distance is stable, we will know that the robot will be stable in the other positions.

To obtain this error measure, we made a maximum of 10.000 iterations in every training network if the training error does not achieve the required value. Each network was trained 5 times with three different training sets. When the previously commented errors are obtained, we get three new measurements:

- *Maximum Distance Mean Error*. The mean error is obtained for every training set, considering the 5 repetitions. We consider a good mean error when it is less than 10 mm. We are trying to discard the random results in the process.
- *Minimum Distance*. This measurement looks for the minimum distance in the 5 repetitions and the same training set. We consider a good minimum distance if it is less than 5 mm. We pretend to found the networks with at least one correct case. Also we can discard all the networks without any correct case.
- *Minimum Distance Deviation*. The deviation permits appreciate, in general terms, how the minimum distance errors are in respect to the 5 repetitions.

6 Experimental Results

The high number of trained networks and the obtained results permit to make many remarks on the aspects that we want to consider. In general, when we said that a network was correct, we express that the results are sufficiently adjusted to the criteria that we comment in the previous section and are based on the robot stability.

The training algorithm with the biggest number of correct results (at least the 50% of all the correct training networks) is the Levenberg-Marquardt. The BFGS and the One-Step-Secant algorithms have similar results and are in charge of the

rest correct networks. An interesting result is that the flexible backpropagation algorithm does not train any correct result on our criteria. This results are normal beacuse the problem that we have proposed it is better solved with function approximation algorithms and the flexible backpropagation is more suitable to pattern recognition problems.

There are no differences between the logarithmic or the tangential tranfer function results. A half of the training network with correct results use one function type and the rest use the other one. About the size of the training set, we verify that we have better results with middle density sets, specifically the one with 61 examples (60% of the correct networks). The worse set its the one with the biggest density.

We get better results when we use some normalization procedure, scale adjust or canonical normalization. Only the 2% of the training networks with no normalized values have correct results.

With all the network topologies used we have obtain correct results. This fact permits to use the one that we want to implement in hardware. Specifically, the one that need less area resource, in our case, the three module network with four output neurons per module [7].

The only network that have less correct cases (under 10% when the others have more than 20%) is the one that only have 2 values in the input layer. The input values that apparently do not contribute with anything have an effect on the training. Also we could consider that number of exits is high, as we said above, in that case 12 exits, one for every joint.

One of the more important and common problems with ANNs is the overfitting. It is well-know this problem consists of the network get a very low training error which does not correspond to the network response with different values to the included in the training set. It is due to the network is specialized in the training examples but it is unable to generalized the new situations.

The overfitting makes that some networks reports a very low error values during the training stage but the same networks do not achieve a similar error using real values. Fig. 2a. shows the results that a network produces when the training

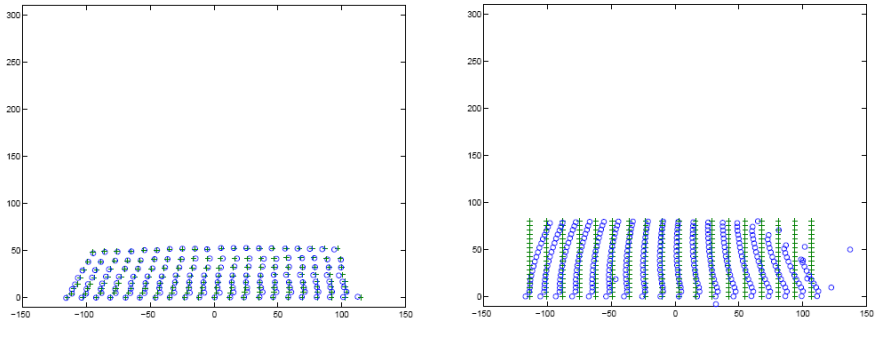


Fig. 2. Results (a) with the training set points and (b) with point belonged to the postural scheme without restrictions

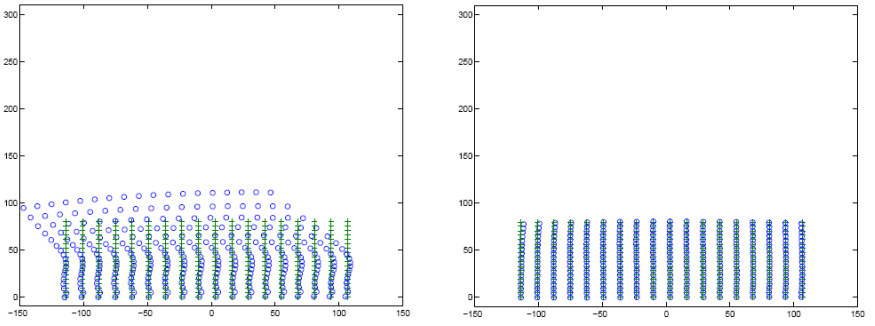


Fig. 3. Results of a netwok (a) without *overfitting* correction and (b) with *overfitting* correction

Table 1. Results summary

ANN	Case 1	Case 2	Error	Distance (mm)
Monolithic	$3 \times 10 \times 12$	$3 \times 6 \times 12$	0,037 ± 0,004	2,979 ± 0,428
Two modules	$3 \times 4 \times 6$	$3 \times 5 \times 6$	0,030 ± 0,005	1,992 ± 0,531
Three modules	$3 \times 6 \times 4$	$3 \times 4 \times 4$	0,049 ± 0,021	3,374 ± 1,254
Reduced output	$3 \times 5 \times 7$	$3 \times 4 \times 7$	0,040 ± 0,014	3,518 ± 0,714
Reduced input	$2 \times 10 \times 12$	$2 \times 6 \times 12$	0,059 ± 0,019	5,147 ± 2,528

values are used, in this case, in a postural scheme corresponding to a robot step. The "+" symbol represents the initial points and the "o" symbol the final points, the ones that the neuronal network return. As we can see, every final point is near the initial point. Fig. 2b represents the results when the points are not equal to the ones used in the training stage, very near to the original but differents. The points in the left and right sides have a big error and the initial and final points are not coincident. When these points are used, the robot will loose the stability.

Once the problem was detected, we have applied common improvements that permit to control that type of situations. Specifically, we modify the performance function, which generally consists of the square sum of network errors, adding a term to control the network weights and thresholds. Thus, networks that with the normal method give very poor results (Fig. 3a), the results are really good using the overfitting correction (Fig. 3b).

Finally, different values for the hidden layer number of neurons. It is a normal situation in ANN that that number depends on the problem, training data set, network topology, and so on. We have verified that with between 4 and 10 neurons the networks can be trained with the desired error.

Table 1 summarizes of the different studied topologies. In the three first columns, the network type identifier and the number of neurons in each layer in the two best cases are shown. The fourth column indicates the mean squared error and the deviation achieved during the training stage. The fifth column shows the real error in the final foot position, in milimeters, with the deviation.

7 Conclusions and Further Work

We have presented the results of an exhaustive study that determines the best artificial neuronal network type to be used for solving the inverse kinematics of humanoid robots using the posture scheme approach. We have defined a restriction set that the network have to satisfy to stay in the solution space, studying the different aspects which make some influence in the solutions. Specifically, we have consider different aspects like training algorithm, transfer function, training set generation, input values normalization and networks topologies.

We can emphasize two cases with special good results which are different to the general conclusions. First, we get a network with two neurons in the input layer that maintains the stability of the robot at every positions and the error is smaller than 2 mm in all of the points. On the other hand, we have a case, with three neurons in the input layer that have an error smaller than 1 mm.

Now we are working on the application of the best networks to a trajectory generator to define the different robot configurations for a simple step. Also we follow with the ANN hardware implementation on FPGA.

References

1. Jordan, M.I., Rumelhart, D.E. (1992) Forward Models: Supervised learning with a distal teacher. Cognitive Science, **16**, 307–354
2. Xia, Y., Wang, J. (2001) A dual network for kinematic control of redundant robot manipulator. IEEE Trans. on System, Man and Cybernetics, Part B, **31**(1):147–154
3. De Lope, J., González-Careaga, R., Zarraonandia, T., Maravall, D. (2003) Inverse kinematics for humanoid robots using artificial neural networks. In R. Moreno-Diaz, F.R. Pichler (eds.) Computer Aided Systems Theory, 448–459
4. Vukobratovic M., Brovac, B., Surla, D., Sotkic, D. (1990) Biped Locomotion. Springer-Verlag, Berlin
5. De Lope, J., Zarraonandia, T., González-Careaga, R., Maravall, D. (2003) Solving the inverse kinematics in humanoid robots: A neural approach. In J. Mira, J.R. Álvarez (eds.) Artificial Neural Nets Problem Solving Methods, LNCS 2687, 177–184
6. Zarraonandia, T., De Lope, J., Maravall, D. (2004) Definition of postural schemes for humanoid robots. In R. Conejo *et al.* (eds.) Current Topics in Artificial Inteligence, LNAI 3040, 241–250
7. Prieto, B., De Lope, J., Maravall, D. (2005) Reconfigurable hardware implementation of neural networks for humanoid locomotion. In J. Mira, J.R. Álvarez (eds.) Artificial Intelligence and Knowledge Engineering Applications: A Bioinspired Approach, LNCS 3562, 395–404

Comparison of Heuristics in Multiobjective A* Search

L. Mandow and J.L. Pérez de la Cruz

Dpto. Lenguajes y Ciencias de la Computación. ETSI Informática. Universidad de Málaga.
29071 - Málaga (Spain)
{lawrence, perez}@lcc.uma.es

Abstract. The paper reconsiders the importance of monotonicity and consistency properties on the efficiency of multiobjective A* search. Previous works on the MOA* algorithm (Multi-objective A*) concluded that the importance of the monotone property of heuristics was not as important as in A*. The recent development of an alternative algorithm (NAMOA*), gives a chance to review these results. The paper presents a formal analysis on the comparison of heuristics in NAMOA* and concludes that the properties of consistency and monotonicity are of fundamental importance in search efficiency.

1 Introduction

Heuristic search has traditionally been one of the fundamental problem solving tools in Artificial Intelligence. The A* algorithm [2] is an efficient solution for the case of admissible graph search guided by evaluation functions. When heuristic functions are monotone, better informed heuristics reduce search effort in A*, measured by the number of expanded nodes [5]. Furthermore, A* is optimal among the class of admissible algorithms when applied to problems with monotone heuristics [1].

The application of the ideas behind A* to multiobjective decision making resulted in a first algorithm denominated MOA* (Multi-Objective A*) [6]. This work showed the admissibility of the new algorithm, and presented results on node expansion and comparison of heuristics. Unfortunately, these results did not have the relevance of their A* analogues. Particularly, it was not possible to prove that search effort, measured by the number of nodes selected for expansion, could be reduced by improved heuristic information, concluding that "the significance of this comparative measure varies widely among different problem domains" (p.802).

More recently, a systematic analysis of the possibilities of multicriteria heuristic search [3] resulted in a new approach to multiobjective A* that we shall denominate NAMOA* [4]. This work showed that the new algorithm is admissible, as well as the existence of an upper bound on the number of paths considered that is violated by MOA*. Experimental results confirmed that the new algorithm offered significant reductions in memory consumption.

This paper continues the formal development of NAMOA*, analyzing the effect of heuristic information and the monotone property on the efficiency of the algorithm. Section 2 describes the peculiarities of multiobjective search, the NAMOA* algorithm, and an example. Section 3 presents results on the effect of heuristic information and

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 180–189, 2006.

the monotone property on search efficiency. These results are compared to those previously obtained for MOA*. Finally some conclusions are summarized and future work is outlined.

2 Multiobjective A* Search

2.1 Multiobjective Search Problems

The multiobjective search problem can be stated as follows: given a labelled, directed, and locally finite graph $G = (N, A, \boldsymbol{c})$, with $|N|$ nodes, and $|A|$ arcs (n, n') labelled with positive cost vectors $\boldsymbol{c}(n, n') \in \mathbb{R}^q$, a start node $s \in N$, and a set of goal nodes $\Gamma \subseteq N$, find the set of paths with nondominated costs in G from s to nodes in Γ.

In multiobjective search problems, cost vectors $\boldsymbol{c}(n, n')$ induce a partial order preference relation $\prec$ denominated dominance that reads *"dominates"*,

$$\forall \boldsymbol{f}, \boldsymbol{f'} \in \mathbb{R}^q \quad \boldsymbol{f} \prec \boldsymbol{f'} \quad \Leftrightarrow \quad \forall i \quad f_i \leq f'_i \ \wedge \ \boldsymbol{f} \neq \boldsymbol{f'} \tag{1}$$

where f_i denotes the i-th element in vector $\boldsymbol{f}$.

Given two q-dimensional vectors $\boldsymbol{f}$ and $\boldsymbol{f'}$ (where $q > 1$), it is not always possible to tell which is better than the other. For example, in a bidimensional cost space, vector $(2, 3)$ dominates $(2, 4)$, but no dominance relation exists between $(2, 3)$ and $(3, 2)$.

Important differences can be established between multiobjective and single-objective search problems. First of all, the search tree used in A* to store the best known path to each node is no longer sufficient. A directed acyclic graph can be used instead to store the set of nondominated known paths to each node.

Heuristic estimates will normally involve a set of vectors $H(n)$ for each node n. These estimate the costs of nondominated paths from n to each goal node. Let $\boldsymbol{g}(P)$ be the function that returns the cost of path P, defined by the sum of the costs of all its component arcs. Therefore, for each path P_{sn} from s to n with cost $\boldsymbol{g}(P_{sn}) = \boldsymbol{g}_P$, there will be a set of heuristic evaluation vectors $F(P_{sn})$. This function is the multiobjective analogue of $f(n)$ in A*,

$$F(P_{sn}) = F(n, \boldsymbol{g}_P) = nondom(\{\boldsymbol{f} \mid \boldsymbol{f} = \boldsymbol{g}_P + \boldsymbol{h} \ \wedge \boldsymbol{h} \in H(n)\})$$

where $nondom(X)$ denotes the subset of nondominated vectors in set X.

Finally, note that at each iteration A* selects and expands an open node, i.e. considers all possible extensions of the path stored in the search tree to that node. In A* each open node stands for a single partial solution path than can be further explored. However, this is not the case in multiobjective search, where an acyclic graph is used instead of a search tree to store interesting partial solution paths. While the MOA* algorithm was designed preserving *node* selection and expansion as basic operations, the NAMOA* algorithm was designed preserving *path* selection and expansion as basic operations.

2.2 NAMOA* Algorithm

The pseudocode for the NAMOA* multiobjective search algorithm [4] is shown in table 1. All the distinguishing features of multiobjective search discussed in section 2.1 are incorporated in the algorithm:

- An acyclic search graph SG is used to store all interesting partial solution paths. For each node n in SG, two sets $G_{cl}(n)$ and $G_{op}(n)$ denote the sets of nondominated cost vectors of paths reaching n that have and have not been explored yet respectively (i.e. closed and open). Each cost vector in these sets labels one or more arcs in the graph from n to their parents. Initially, s is the only node in SG.
- The algorithm keeps an $OPEN$ list of partial solution paths that can be further explored. For each node n in SG and each nondominated cost vector $\boldsymbol{g} \in G_{op}(n)$, there is a corresponding tuple $(n, \boldsymbol{g}, F(n, \boldsymbol{g}))$ in $OPEN$. Initially, $(s, \boldsymbol{g}_s, F(s, \boldsymbol{g}_s))$ is the only tuple in $OPEN$.

Table 1. The NAMOA* algorithm

1. CREATE:
   - An acyclic search graph SG rooted in s.
   - List of alternatives, $OPEN$ = $\{(s, \boldsymbol{g}_s, F(s, \boldsymbol{g}_s))\}$.
   - Two empty sets, $GOALN$, $COSTS$.
2. CHECK TERMINATION. If $OPEN$ is empty, then backtrack in SG from the nodes in $GOALN$ and return the set of solution paths with costs in $COSTS$.
3. PATH SELECTION. Select an alternative $(n, \boldsymbol{g}_n, F)$ from OPEN with $\boldsymbol{f} \in F$ nondominated in $OPEN$, i.e.

$$\forall (n', \boldsymbol{g}_{n'}, F') \in OPEN \quad \nexists \boldsymbol{f}' \in F' \quad | \quad \boldsymbol{f}' \prec \boldsymbol{f} \tag{2}$$

   Delete $(n, \boldsymbol{g}_n, F)$ from $OPEN$, and move $\boldsymbol{g}_n$ from $G_{op}(n)$ to $G_{cl}(n)$.
4. SOLUTION RECORDING. If $n \in \Gamma$, then
   - Include n in $GOALN$ and $\boldsymbol{g}_n$ in $COSTS$.
   - Eliminate from $OPEN$ all alternatives (x, g_x, F_x) such that all vectors in F_x are dominated by $\boldsymbol{g}_n$ (FILTERING).
   - Go back to step 2
5. PATH EXPANSION: If $n \notin \Gamma$, then For all successors nodes m of n that do not produce cycles in SG do:
   (a) Calculate the cost of the new path found to m: $\boldsymbol{g}_m = \boldsymbol{g}_n + \boldsymbol{c}(n, m)$.
   (b) If m is a new node
      i. Calculate $F_m = F(m, \boldsymbol{g}_m)$ filtering estimates dominated by $COSTS$.
      ii. If F_m is not empty, put $(m, \boldsymbol{g}_m, F_m)$ in $OPEN$, and put $\boldsymbol{g}_m$ in $G_{op}(m)$ labelling a pointer to n.
      iii. Go to step 2.

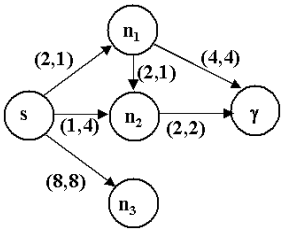
   else (m is not a new node), in case
   - $\boldsymbol{g}_m \in G_{op}(m)$ or $\boldsymbol{g}_m \in G_{cl}(m)$: label with $\boldsymbol{g}_m$ a pointer to n, and go to step 2.
   - If $\boldsymbol{g}_m$ is non-dominated by any cost vectors in $G_{op}(m) \cup G_{cl}(m)$ (a path to m with new cost has been found), then :
      i. Eliminate from $G_{op}(m)$ and $G_{cl}(m)$ vectors dominated by $\boldsymbol{g}_m$
      ii. Calculate $F_m = F(m, \boldsymbol{g}_m)$ filtering estimates dominated by $COSTS$.
      iii. If F_m is not empty, put $(m, \boldsymbol{g}_m, F_m)$ in $OPEN$, and put $\boldsymbol{g}_m$ in $G_{op}(m)$ labelling a pointer to n.
      iv. Go to step 2.
   - Otherwise: go to step 2.

- At each iteration, the algorithm considers the expansion of an open tuple $(n, \boldsymbol{g}, F)$ that stands for a partial solution path from s to n with cost $\boldsymbol{g}$.
- Two sets $GOALN$ and $COSTS$ record all known goal nodes and all nondominated cost vectors to goal nodes respectively. Each time a solution is found, its cost vector can be used to rule out dominated tuples (this operation is denominated *filtering*).
- Search terminates only when the $OPEN$ list becomes empty, i.e. when all open tuples have been either selected, filtered, or pruned.

2.3 Example

This section illustrates the fundamental features of NAMOA* with the help of a simple example. Figure 1 shows a sample graph, a heuristic function $H(n)$, and a trace of the $OPEN$ list ($(n, \boldsymbol{g}, F)$ tuples). Figures 2 and 3 show SG at the start of each iteration with the values of the G_{op} and G_{cl} sets, (sets that do not change from previous iterations are omitted).

At iteration 1 SG has only the start node s at its root with $G_{op}(s) = \{(0,0)\}$, $G_{cl}(s) = \emptyset$. The only path in $OPEN$ is selected, and its extensions to nodes n_1, n_2, y n_3 are added to SG and $OPEN$. At iteration 2 there are two alternatives with nondominated costs in $OPEN$ (one to n_1 with cost $(4,3)$, and the other to n_2 with cost $(2,5)$). Let us assume that ties are always broken choosing the cost vector with smaller first component. The path to n_2 would be selected, and its extension to γ generated. At iteration 3, the new tie between n_1 and γ would be broken choosing the path to γ. Since this is a goal node, the following sets are updated, $GOALN = \{\gamma\}$ and $COSTS = \{(3,6)\}$. The path to n_3 would be *filtered* from $OPEN$, since its estimate is dominated by the cost of the solution path found. At iteration 4 the only open alternative is the path to n_1. This expansion generates new nondominated paths to n_2 and γ. At iteration 5 the path to n_2 is the only one nondominated in $OPEN$. This expansion generates a new path to γ that dominates (and, therefore, *prunes*) the path reaching that same node from n_1. At iteration 7 a new solution path is generated, resulting in $COSTS = \{(3,6),(6,4)\}$ and an empty $OPEN$ list.



n	$H(n)$
s	$\{(2,2)\}$
n_1	$\{(2,2)\}$
n_2	$\{(1,1)\}$
n_3	$\{(1,1)\}$
γ	$\{(0,0)\}$

It.	$OPEN$
1	$(s \ \ (0,0)\ (2,2))* \leftarrow$
2	$(n_1\ (2,1)\ (4,3))*$ $(n_2\ (1,4)\ (2,5))* \leftarrow$ $(n_3\ (8,8)\ (9,9))$
3	$(n_1\ (2,1)\ (4,3))*$ $(\gamma\ \ (3,6)\ (3,6))* \leftarrow$ $(n_3\ (8,8)\ (9,9))$
4	$(n_1\ (2,1)\ (4,3))* \leftarrow$
5	$(n_2\ (4,2)\ (5,3))* \leftarrow$ $(\gamma\ \ (6,5)\ (6,5))$
6	$(\gamma\ \ (6,4)\ (6,4))* \leftarrow$

Fig. 1. Sample graph, heuristic function $H(n)$, and trace of $OPEN$ at each iteration (* = non-dominated; $\leftarrow$ = selected)

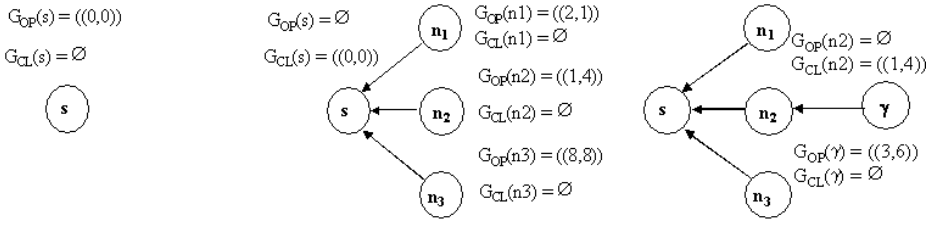


Fig. 2. Search graph SG at iterations 1 (left), 2 (center) and 3 (right)

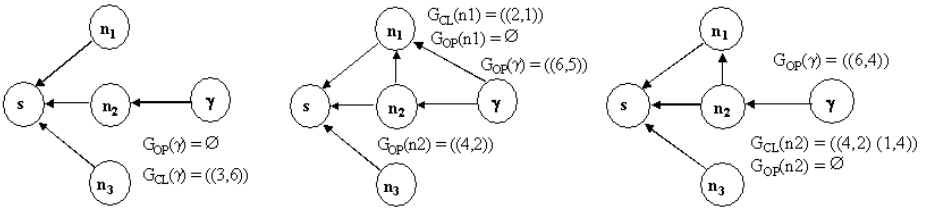


Fig. 3. Search graph SG at iterations 4 (left), 5 (center) and 6 (right)

3 Properties

This section introduces necessary and sufficient conditions for path expansion in NAMOA*. This allows a comparison between NAMOA algorithms depending on the heuristic information available. Finally, the importance of the monotonicity property in search efficiency is formally established.

Let C^* be the set of cost vectors of nondominated solutions, and $\preceq$ the relation "dominates or equals". The relation "is indifferent to", denoted by $\sim$, is defined as follows, $\boldsymbol{x} \sim \boldsymbol{y} \iff \boldsymbol{x} \not\prec \boldsymbol{y} \wedge \boldsymbol{y} \not\prec \boldsymbol{x}$. The relation "dominates or is indifferent to", denoted by $\precsim$, is defined as follows, $\boldsymbol{x} \precsim \boldsymbol{y} \iff \boldsymbol{x} \prec \boldsymbol{y} \vee \boldsymbol{x} \sim \boldsymbol{y}$.

A heuristic function $H(n)$ is said to be admissible when for all nondominated solutions $P^* = (s = n_0, n_1, \ldots, n_i, n_{i+1} \ldots n_k), n_k \in \Gamma$ and all subpaths $P_i^* = (n_0, \ldots, n_i)$ of P^* the following holds,

$$\exists \boldsymbol{h} \in H(n_i) \quad | \quad \boldsymbol{g}(P_i^*) + \boldsymbol{h} \preceq \boldsymbol{g}(P^*)$$

3.1 Necessary and Sufficient Conditions for Path Expansion

Let us start introducing some preliminary results. The following theorem is taken from the admissibility proofs of NAMOA* [4]. It is the starting point for a first necessary condition that will be useful later on.

Theorem 1. *([4], theorem 1) For each non-dominated solution path $P^* = (s, n_1, \ldots, n_i, n_{i+1} \ldots \gamma)$ with cost $\boldsymbol{g}(P^*) = \boldsymbol{c}^*$, there is always before its discovery a subpath $P_i^* = (s, n_1, \ldots, n_i)$ of P^* such that: a)P_i^* is recorded in SG; b) $\boldsymbol{g}(P_i^*) \in G_{op}(n_i)$; c) $\exists \boldsymbol{f} \in F(P_i^*) \mid \boldsymbol{f} \preceq \boldsymbol{c}^*$.*

Theorem 2. *Each path $P = (s, \ldots, n)$ selected from $OPEN$ for expansion satisfies upon selection that,*

$$\exists \boldsymbol{h} \in H(n) \quad | \quad \nexists \boldsymbol{c}^* \in C^*, \quad \boldsymbol{c}^* \prec \boldsymbol{g}(P) + \boldsymbol{h}$$

PROOF: Let's assume, for the purpose of contradiction, that a path P is selected and that for all heuristic evaluation vector $\boldsymbol{f'} \in F(P)$ exists $\boldsymbol{c}^* \in C^* \mid \boldsymbol{c}^* \prec \boldsymbol{f'}$. It must be that such vectors $\boldsymbol{c}^* \notin COSTS$, for otherwise P would have been filtered. Now, from theorem 1, for each cost vector $\boldsymbol{c}^* \in C^*$ there is at least one path $P_i = (s, \ldots, n_i)$ with $\boldsymbol{f} \in F(P_i)$ such that $\boldsymbol{f} \preceq \boldsymbol{c}^*$, and $(n_i, \boldsymbol{g}(P_i), F(P_i))$ is in $OPEN$. From the transitivity of the dominance relation we get $\boldsymbol{f} \preceq \boldsymbol{c}^* \prec \boldsymbol{f'}$ for all $\boldsymbol{f'} \in F(P)$. Therefore, P can never be selected for expansion. □

Note that theorem 2 is a necessary condition for path expansion. However, it relies on the assumption that the path in question is already in $OPEN$. The following theorems introduce necessary and sufficient conditions for path expansion that are independent of each particular run of the algorithm. The following definition is the multiobjective equivalent of the one presented for the scalar case ([5], p. 80).

Definition 1. *We say that a path $P = (s = n_0, n_1, n_2, \ldots, n_k)$ is* C-bounded *with respect to $H(n)$ if for all subpaths $P_i = (n_0, n_1, \ldots, n_i)$ of P the following holds,*

$$\exists \boldsymbol{h} \in H(n_i) \quad | \quad \nexists \boldsymbol{c} \in C, \quad \boldsymbol{c} \prec \boldsymbol{g}(P_i) + \boldsymbol{h}$$

An equivalent condition is,

$$\forall \boldsymbol{c} \in C \quad \exists \boldsymbol{h} \in H(n_i) \quad | \quad \boldsymbol{g}(P_i) + \boldsymbol{h} \precsim \boldsymbol{c} \quad \vee \quad \boldsymbol{g}(P_i) + \boldsymbol{h} = \boldsymbol{c}$$

When it becomes necessary to identify the heuristic function we shall use the notation C(H)-bounded.

Note that, by definition, a C^*-bounded path will never be filtered. Therefore, once such paths enter $OPEN$ they can only be either pruned or selected for expansion.

Theorem 3. *A sufficient condition for NAMOA* to select a path $P = (s, \ldots, n)$ for expansion is that: a)P be a nondominated path to n; b) P be C^*-bounded.*

PROOF: Let's assume, for the purpose of contradiction, that NAMOA* terminates and $P = (s = n_o, n_1 \ldots, n_k = n)$ was not selected. Since P is C^*-bounded, its subpaths can never be filtered. Since P is nondominated, its subpaths can never be pruned. Therefore, there will always be an open subpath $P_i = (n_0, \ldots, n_i)$ of P before termination. However, the algorithm can only terminate when $OPEN$ becomes empty, and all subpaths of P and P itself will have to be selected. □

Theorem 4. *A necessary condition for NAMOA* to select a path P for expansion is that P be C^*-bounded.*

PROOF: For a path $P = (s = n_0, n_1, n_2, \ldots, n_k)$ to be selected it must be in $OPEN$ and also, by virtue of theorem 2,

$$\exists \boldsymbol{h} \in H(n) \quad | \quad \nexists \boldsymbol{c}^* \in C^*, \quad \boldsymbol{c}^* \prec \boldsymbol{g}(P) + \boldsymbol{h}$$

Now, if P is in $OPEN$ then all its subpaths $P_i = (n_0, n_1, \ldots, n_i)$ must have been in $OPEN$ and were selected for expansion. At that time each subpath must have satisfied theorem 2. In summary, all subpaths P_i of P satisfy the following,

$$\exists \boldsymbol{h} \in H(n_i) \quad | \quad \nexists \boldsymbol{c}^* \in C^* \quad \boldsymbol{c}^* \prec \boldsymbol{g}(P_i) + \boldsymbol{h}$$

which is the definition of C^*-bounded path (definition 1). □

Theorems 3 and 4 introduce a fundamental classification of paths in the graph. Those that are not C^*-bounded will never be considered for expansion. Among those C^*-bounded, nondominated ones will always be considered, while the fate of dominated ones is unclear: in the worst case all of them will be considered, and in the best case none of them.

3.2 Comparison of Heuristics

The necessary and sufficient conditions developed in the previous subsection are used here to broadly characterize when a heuristic function is more convenient than other. The intuition is that, in general, the greater the heuristic estimates, the smaller the set of C^*-bounded paths. The following definition introduces the notion of 'more informed heuristic' [1].

Definition 2. *A heuristic function $H_2(n)$ is said to be* at least as informed *as other $H_1(n)$ when both are admissible and all nodes n in the graph satisfy that,*

$$\forall \boldsymbol{h}_2 \in H_2(n) \quad \exists \boldsymbol{h}_1 \in H_1(n) \quad | \quad \boldsymbol{h}_1 \preceq \boldsymbol{h}_2$$

Theorem 5. *Let $H_1(n)$ and $H_2(n)$ be two admissible heuristics for the same problem. Let $NAMOA_1$ and $NAMOA_2$ be two versions of algorithm NAMOA* that differ only in the use of heuristic functions $H_1(n)$ and $H_2(n)$ respectively. If $H_2(n)$ is at least as informed as $H_1(n)$, then all nondominated and $C^*(H_2)$-bounded paths selected for expansion by $NAMOA_2$ will also be selected by $NAMOA_1$.*

PROOF: It suffices to show that if a path is $C^*(H_2)$-bounded, then it will also be $C^*(H_1)$-bounded. Since paths are also nondominated, application of theorem 3, proves the paths will also be expanded by $NAMOA_1$.

A path $P = (s = n_0, n_1, n_2, \ldots, n_k)$ is C^*-bounded according to $H_2(n)$ if every subpath $P_i = (n_0, n_1, \ldots, n_i)$ satisfies (definition 1),

$$\forall \boldsymbol{c}^* \in C^* \quad \exists \boldsymbol{h}_2 \in H_2(n_i) \quad | \quad \boldsymbol{g}(P_i) + \boldsymbol{h}_2 \precsim \boldsymbol{c}^*$$

[1] Note there is some erratum in the analogous definition in [6].

Now, since H_2 is at least as informed as H_1, for all $\boldsymbol{h}_2 \in H_2$ exists a vector $\boldsymbol{h}_1 \in H_1$ such that $\boldsymbol{h}_1 \preceq \boldsymbol{h}_2$ (definition 2). Therefore,

$$\boldsymbol{g}(P_i) + \boldsymbol{h}_1 \preceq \boldsymbol{g}(P_i) + \boldsymbol{h}_2 \precsim \boldsymbol{c}^*$$

Since $\prec$ is transitive, and all natural q and all vectors $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z} \in \mathbb{R}^q$ satisfy the relationship $\boldsymbol{x} \prec \boldsymbol{y} \wedge \boldsymbol{y} \sim \boldsymbol{z} \Rightarrow \boldsymbol{x} \precsim \boldsymbol{z}$, it is possible to conclude that, $\boldsymbol{g}(P_i) + \boldsymbol{h}_1 \precsim \boldsymbol{c}^*$.

□

Regrettably, the implications of theorem 5 are limited. In general, it is not possible to determine which dominated C^*-bounded paths will be expanded by each version of the algorithm.

3.3 Consistency and Monotonicity

Previous sections have shown for NAMOA* properties analogous to those found for A*. This section further shows that the monotonicity property has also an analogous importance in NAMOA*. The following definitions are equivalent to those presented in [6](p. 806)

Definition 3. *A multiobjective heuristic function $H(n)$ is* monotone *if for all arcs (n, n') in the graph, the following condition holds,*

$$\forall \boldsymbol{h}' \in H(n') \quad \exists \boldsymbol{h} \in H(n) \quad | \quad \boldsymbol{h} \preceq \boldsymbol{c}(n, n') + \boldsymbol{h}'$$

Definition 4. *A multiobjective heuristic function $H(n)$ is* consistent *if for all pairs of nodes n, n' in the graph, for all nondominated path between them $P = (n, \ldots, n')$, and for all heuristic cost vector $\boldsymbol{h}' \in H(n')$, the following condition holds,*

$$\exists \boldsymbol{h} \in H(n) \quad | \quad \boldsymbol{h} \preceq \boldsymbol{c}(P) + \boldsymbol{h}'$$

It can be shown that $H(n)$ is consistent if and only if it is monotone ([6] lemma 18), and that if $H(n)$ is monotone, then it is also admissible ([6] lemma 19).

The following theorem introduces a new necessary condition for path expansion when heuristics are monotone.

Theorem 6. *If the heuristic function $H(n)$ is monotone, a necessary condition for NAMOA* to select a path $P = (s, \ldots, n)$ for expansion is that P be a nondominated path to n.*

PROOF: Let's assume, for the purpose of contradiction, that NAMOA* selects a path $P' = (s, \ldots, n)$ dominated to n, and represented in $OPEN$ by the tuple $(n, \boldsymbol{g}(P'), F(P'))$. There must be some path $P^* = (s = n_0, n_1, \ldots, n_k = n)$ yet to be discovered that is nondominated to n and dominates P'.

Since P^* is nondominated its subpaths will never be pruned. Since $\boldsymbol{g}(P^*) \prec \boldsymbol{g}(P')$ and the heuristic is monotone, if P' was not filtered, none of the subpaths of P^* will have been filtered either. Therefore, there must be some subpath $P_i^* = (n_0, n_1, \ldots, n_i)$ of P^* in $OPEN$.

Let $P^*_{i2} = (n_i, \ldots, n)$ be the subpath of P^* from n_i to n, such that $\boldsymbol{g}(P^*) = \boldsymbol{g}(P^*_i) + \boldsymbol{g}(P^*_{i2})$. Since costs are nonnegative and the relationship $\prec$ is transitive, we have that,

$$\boldsymbol{g}(P^*_i) \preceq \boldsymbol{g}(P^*) \prec \boldsymbol{g}(P') \quad \Rightarrow \quad \boldsymbol{g}(P^*_i) \prec \boldsymbol{g}(P')$$

From the definition of consistency,

$$\forall \boldsymbol{h} \in H(n) \quad \exists \boldsymbol{h}' \in H(n_i) \quad | \quad \boldsymbol{h}' \preceq \boldsymbol{g}(P^*_{i2}) + \boldsymbol{h}$$

Combining both results,

$$\forall \boldsymbol{f} \in F(P') \quad \exists \boldsymbol{f}' \in F(P^*_i) \quad | \quad \boldsymbol{f}' \prec \boldsymbol{f}$$

Therefore P' could never be selected from $OPEN$ until all subpaths P^*_i of P had been selected. However, if $P^*_i = P^*$ ever entered $OPEN$, P' would be pruned. □

Thanks to this result, the theorems presented in previous sections can be restated for the case of monotone heuristics.

Theorem 7. *If H is consistent, then a* necessary and sufficient *condition for NAMOA to select some path $P = (s, \ldots, n)$ for expansion is that: a) P be nondominated to n; b) P be C^*-bounded.*

PROOF: Theorems 4 and 6 establish as necessary conditions for path expansion the same sufficient conditions established by theorem 3. □

Theorem 8. *Let $H_1(n)$ and $H_2(n)$ be two admissible heuristics for the same problem. Let $H_2(n)$ be additionally* **monotone***. Let $NAMOA_1$ and $NAMOA_2$ be two versions of NAMOA* that differ only in the use of heuristic functions $H_1(n)$ and $H_2(n)$ respectively. If $H_2(n)$ is at least as informed as $H_1(n)$, then* **all** *paths selected for expansion by $NAMOA_2$ will also be selected for expansion by $NAMOA_1$.*

PROOF: Obvious from theorems 5 and 6, since all paths selected by $NAMOA_2$ will be nondominated. □

It can be easily seen that, just as with A*, the monotone property guarantees that only the strictly necessary C^*-bounded paths will be selected for expansion (in other words, the pruning of C^*-bounded paths is maximum). Additionally, if some heuristic H_2 is monotone and at least as informed as another one H_1, then for every possible problem instance the set of paths selected with H_2 will always be a subset of those selected with H_1.

3.4 Comparison with MOA*

Although for space reasons it is not possible to present here details on the workings of MOA* [6], it is interesting to draw a comparison between the results obtained for this algorithm and those presented in previous sections for NAMOA*.

One of the main difficulties is that the results obtained for MOA* try to measure the algorithm's efficiency in terms of the number of nodes selected for expansion. To

complicate things further, MOA* may expand dominated paths even when the heuristics are monotone (see the example presented in [4]). Thus, the necessary and sufficient condition for *node* expansion in MOA* is the existence of some C^*-bounded path to that node ([6] lemma 17). If there is some C^*-bounded path, there will exist at least a C^*-bounded and nondominated one, therefore the property is coherent with the conditions presented in section 3.1. However, the conditions for NAMOA* are more precise, and distinguish exactly which C^*-bounded paths to each node will be unavoidably expanded and which not.

In a similar fashion, if some heuristic H_2 is at least as informed as another one H_1, then all *nodes* selected by MOA* with H_2 will also be selected with H_1 ([6] theorem 4). The result is again correct, but falls short of our expectations, since it cannot tell *how many times* each node will be expanded with each heuristic.

Finally, the analysis of MOA* concluded that the properties of monotonicity and consistency were not as important as in the single-objective case, because "they are not sufficient to guarantee that nodes will not need to be reopened [in MOA*]"([6] p. 806). The results presented in section 3.3 prove this statement does not apply to NAMOA*, showing that when heuristics are monotone, pruning is maximum just as in the single-objective case.

4 Conclusions and Future Work

The results presented in the paper restore the importance of consistency and monotonicity in multiobjective A* search. The results on necessary and sufficient conditions for path expansion and comparison of heuristics establish that, when heuristics are monotone, pruning is maximum. Therefore, a monotone heuristic will always be preferable to some other equally or less informed. This result is in sharp contrast with those previously established for the MOA* algorithm.

The monotonicity property plays an important role in the optimality proofs of the A* algorithm over the class of admissible search algorithms [1]. The analysis of analogous conditions of optimality for NAMOA* over the equivalent class of multiobjective search algorithms is an important future work.

References

1. Dechter, R., Pearl, J.: Generalized Best-First Search Strategies and the Optimality of A*. Journal of the ACM **32**(3) (1985) 505–536.
2. Hart, P. E., Nilsson, N. J., Raphael, B.: A formal basis for the heuristic determination of minimum cost paths. IEEE Tr. Systems Science and Cybernetics SSC-4 **2** (1968) 100–107.
3. Mandow, L., Pérez de la Cruz, J. L.: Multicriteria heuristic search. European Journal of Operational Research **150** (2003) 253–280.
4. Mandow, L., Pérez de la Cruz, J. L.: A new approach to multiobjective A*. Proc. of the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI-05) 218–223.
5. Pearl, J.: Heuristics. Addison-Wesley, Reading, MA., 1984.
6. Stewart, B. S., White, C. C.: Multiobjective A*. Journal of the ACM **38**(4) (1991) 775–814.

Contour-Based Shape Retrieval Using Dynamic Time Warping

Andrés Marzal, Vicente Palazón, and Guillermo Peris*
{amarzal, palazon, peris}@lsi.uji.es

Dept. Llenguatges i Sistemes Informàtics. Universitat Jaume I de Castelló. Spain

Abstract. A dissimilarity measure for shapes described by their contour, the Cyclic Dynamic Time Warping (CDTW) dissimilarity, is introduced. The dissimilarity measure is based on Dynamic Time Warping of cyclic strings, i.e., strings with no definite starting/ending points. The Cyclic Edit Distance algorithm by Maes cannot be directly extended to compute the CDTW dissimilarity, as we show in the paper. We present an algorithm that computes the CDTW dissimilarity in $O(mn \log n)$ time, where m and n are the lengths of the cyclic strings. Shape retrieval with the new dissimilarity measure is experimentally compared with the WARP system on a standard corpus.

1 Introduction

Contour matching is an important problem in shape classification and retrieval. Contours can be represented with *cyclic strings*: strings where no starting point can be univoquely defined. A cyclic string can be viewed as the set of strings obtained by cyclically shifting a representative string. Let $A = a_0 a_1 \dots a_{m-1}$ be a string from a (possibly infinite) alphabet Σ and let Σ^* be the closure under concatenation of Σ. A cyclic shift σ of A is a mapping $\sigma : \Sigma^* \to \Sigma^*$ defined as $\sigma(a_0 a_1 \dots a_{m-1}) = a_1 a_2 \dots a_{m-1} a_0$. Let σ^k denote the composition of k cyclic shifts and let σ^0 denote the identity. Two strings A and A' are cyclically equivalent if $A = \sigma^k(A')$ for some k. A cyclic string is an equivalence class $[A] = \{\sigma^k(A) : 0 \le k < m\}$. Any of its members is a representative (non-cyclic) string. For instance, let $\Sigma = \{x, y\}$ be an alphabet; the set $\{yxxx, xxxy, xxyx, xyxx\}$ is a cyclic string and $yxxx$ (or any other string in the set) can be taken as its representative. The contour of a shape can be described in terms of cyclic strings using different sets Σ: Freeman chaincodes, 2D points, curvature and/or distance to centroid values, etc.

Levenshtein defined the Edit Distance (ED) between two (non-cyclic) strings A and B as the minimum (weighted) number of edit operations (insertions, deletions, and substitutions) transforming A into B. Wagner and Fisher proposed in [11] an $O(mn)$ time algorithm to compute the ED, where m and n are the lengths of A and B. The Cyclic Edit Distance (CED) between the cyclic strings

* This work has been supported by the Spanish *Ministerio de Ciencia y Tecnología* and FEDER under grant TIC2002-02684.

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 190–199, 2006.

$[A]$ and $[B]$ is the minimum (weighted) number of edit operations needed to transform $[A]$ into $[B]$ and is defined in terms of the minimum ED between any pair of representatives. A trivial, $O(m^2n^2)$ time algorithm to compute the CED consists in obtaining the ED for all mn pairs of representatives and choosing the minimum ED value. Maes proposed in [4] a Divide-and-Conquer algorithm to compute the CED in $O(mn \log n)$ time. In [6,8], Marzal *et al.* improved the running time of this algorithm by proposing a Branch-and-Bound exploration of its search space. In [2], Bunke and Bühler obtained an approximate value for the CED in $O(mn)$ time. Mollineda *et al.* proposed in [7] other heuristics to approximate the value of the CED.

Edit operations naturally arise in the comparison of some contour parameterizations such as cyclic Freeman chaincodes, but are difficult to define properly when contours are represented by sequences of points or real values. Maes applied the (exact) cyclic edit distance computation to the recognition of shapes described with polygons [5]. He pointed out that the CED has some drawbacks when applied to this problem: it is sensitive to segmentation inconsistencies in the polygons. Each primitive of one polygon is either aligned with one and only one primitive of the other polygon (a substitution) or deleted/inserted: similar regions of polygons represented by a different number of edges cannot be aligned.

A different dissimilarity measure for (non-cyclic) strings can be defined in terms of Dynamic Time Warping (DTW), which is based on the weight of an optimal alignment of two (non-cyclic) strings [9]. DTW is similar to ED, but the only "edit operations" allowed are (possibly one-to-many) substitutions. DTW has been successfully applied to speech recognition, on-line handwritten text recognition, time series alignment, etc. Some approaches to shape matching represent contours with global features such as Fourier descriptors or invariant moments [12]. Recently, Bartolini *et al.* [1] have designed a shape retrieval system (WARP) that uses DTW to compare cyclic sequences of 2D points that have been "normalized" by means of Fourier descriptors manipulation. The WARP system has been tested on a shape retrieval standard database and their results outperform other DTW-based methods.

In this paper, we propose a different Cyclic Dynamic Time Warping (CDTW) algorithm which is inspired in the CED algorithm by Maes. We show that Maes' algorithm cannot be directly extended to compute the CDTW and provide $O(mn \log n)$ time algorithm for the CDTW dissimilarity computation. Experiments run on the same shape retrieval task presented in [1] show that our method provides significantly better results.

2 Dynamic Time Warping

Let $A = a_0a_1 \dots a_{m-1}$ and $B = b_0b_1 \dots b_{n-1}$ be two strings in Σ^*. An *alignment* between two sequences A and B is a sequence of pairs (i_0, j_0), (i_1, j_1), $\dots$, (i_{k-1}, j_{k-1}) such that (a) $0 \leq i_\ell < m$ and $0 \leq j_\ell < n$; (b) $0 \leq i_{\ell+1} - i_\ell \leq 1$ and $0 \leq j_{\ell+1} - j_\ell \leq 1$; and (c) $(i_\ell, j_\ell) \neq (i_{\ell+1}, j_{\ell+1})$. The pair (i_ℓ, j_ℓ) is said to *align* a_{i_ℓ} with b_{j_ℓ}. Each pair is weighted by means of a local dissimilarity function

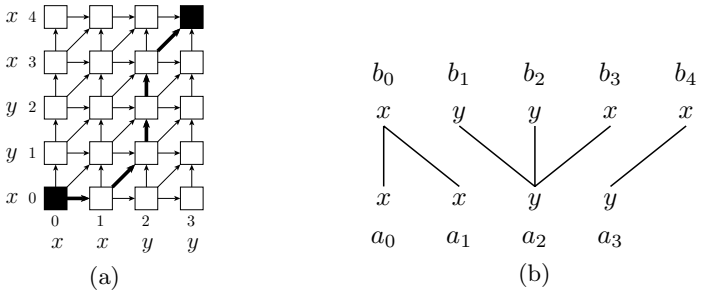


Fig. 1. (a) Warping graph for $xyyxx$ and $xxyy$. The optimal alignment is shown with thicker arrows and is $(0,0)$, $(1,0)$, $(2,1)$, $(2,2)$, $(2,3)$, $(3,4)$. (b) Aligned symbols.

$\gamma : \Sigma \times \Sigma \to \mathbb{R}^{\geq 0}$. The γ function can be defined as the euclidean distance of 2D points, the difference of curvatures, etc. The weight of an alignment is defined as $\sum_{0 \leq \ell < k} \gamma(a_{i_\ell}, b_{j_\ell})$.

An alignment between A and B is optimal if its weight is minimum among that of all possible alignments. The *Dynamic Time Warping* (DTW) *dissimilarity* measure between A and B will be denoted with $D(A,B)$ and is defined as the weight of the optimal alignment between both sequences. The DTW dissimilarity can be computed as $D(A,B) = d(m-1, n-1)$, where d is this recurrence:

$$d(i,j) = \begin{cases} \gamma(a_0, b_0), & \text{if } i = j = 0; \\ d(i-1,j) + \gamma(a_i, b_0), & \text{if } i > 0 \text{ and } j = 0; \\ d(i,j-1) + \gamma(a_0, b_j), & \text{if } i = 0 \text{ and } j > 0; \\ \min \left\{ \begin{array}{l} d(i-1,j-1), \\ d(i-1,j), \\ d(i,j-1) \end{array} \right\} + \gamma(a_i, b_j), & \text{if } i > 0 \text{ and } j > 0. \end{cases} \tag{1}$$

This equation formulates the $D(A,B)$ computation problem as a shortest path problem in the so-called *warping graph*, an array of nodes (i,j), where $0 \leq i < m$ and $0 \leq j < n$, connected by horizontal, vertical and diagonal arcs (see Fig. 1). All arcs incident on the same node (i,j) are weighted with $\gamma(x_i, y_j)$. Each path from $(0,0)$ to $(m-1, n-1)$ defines an alignment between A and B containing a pair (i,j) for each traversed node (i,j). The weight of a path (and of its corresponding alignment) is the sum of the weights of its arcs. Since the warping graph is acyclic, the optimal path can be computed by Dynamic Programming in $O(mn)$ time.

Aligned pairs of symbols in DTW can be assimilated to substitutions in ED, but DTW allows for one-to-many correspondences. This makes DTW appropriate to model "elastic distortions" of strings describing shapes or time series. On the other hand, DTW alignments have no insertions or deletions and seem preferable to ED when these operations do not naturally arise. There are alternative definitions of the DTW (different arcs in the warping graph or weighting

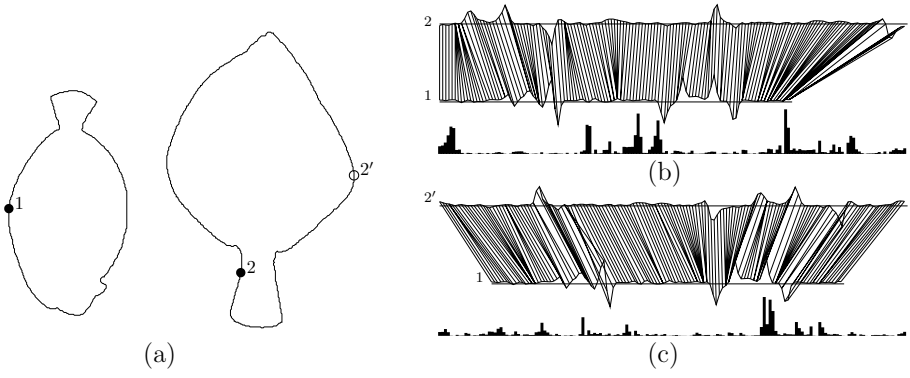


Fig. 2. (a) Two fish shapes. The black dots indicate starting points in their (counter-clockwise) contour coding as strings of curvature values. (b) Optimal alignment of the curvatures starting at black dots. The absolute value of the distance between aligned points is shown under the alignment. The DTW dissimilarity is the sum of these values. (c) A more significant alignment is possible if the second shape string starts at $2'$.

functions that affect differently diagonal arcs). For the sake of clarity, we will consider only DTW similarity as defined by the recurrence (1).

There is a fundamental flaw when trying to compare cyclic strings with DTW: the DTW is sensitive to the election of the starting point. Fig 2 illustrates this problem for cyclic strings of curvature values representing contours.

3 The WARP System

Bartolini *et al.* have presented in [1] a shape retrieval system that compares (cyclic) strings of 2D points describing contours with DTW: the so-called WARP system. DTW cannot be directly applied to sequences of 2D points, since it is sensitive to changes in position, scale, angle, and starting point (i.e., the chosen representative for the cyclic string). The WARP system proposes a normalization of the sequences of 2D points which is invariant to all these factors. This invariant representation is computed by properly manipulating the Fourier descriptors of the sequence (considered as a function of complex numbers), which is reconstructed from its new Fourier descriptors by an inverse Fourier transform.

Let $A = a_0 a_1 \dots a_{m-1}$ be a sequence of complex numbers where each element denotes a point in the complex plane (thus defining a 2D point). The Discrete Fourier Transform of A is a sequence of complex values

$$a'_i = \sum_{0 \le k < m} a_k e^{-j2\pi ki/m}, \qquad i \in \{-m/2, \dots, -1, 0, 1, \dots, m/2 - 1\},$$

where $j = \sqrt{-1}$. Fourier descriptors model the signal as a composition of revolving ellipses. The main ellipse is centered at the contour centroid, a'_0. The

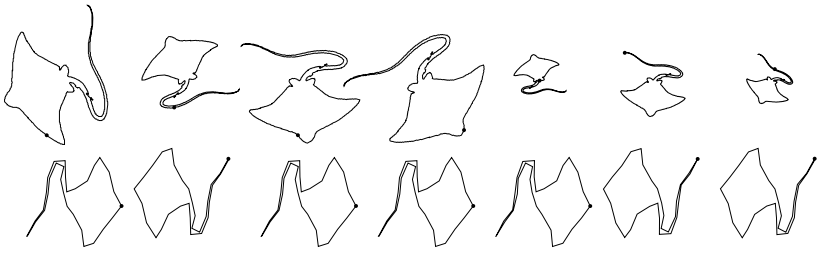


Fig. 3. Upper row: the same shape with different scale, rotation angle and starting point (black dot). Lower row: corresponding normalized version. There is an ambiguity of π radians in rotation and it affects the election of the starting point.

a'_0 descriptor can be set to 0 in order to provide invariance to translation. The value of a'_1 is the length of the main axis of the basic ellipse; therefore, dividing all the descriptors by r_1 provides invariance to scale. Let us consider the polar representation the descriptors: $a'_i = r_i e^{j\theta_i}$. Invariance to rotation can be obtained by substracting $(\theta_{-1} + \theta_1)/2$ (the orientation of the basic ellipse) to all θ_i. Invariance with respect to the starting point can be achieved by adding $i(\theta_{-1} - \theta_1)/2$ to all θ_i. Noise in the contour can be reduced by taking only low frequency components. The WARP system considers only the 32 lower frequency components before computing the inverse Fourier transform. One added benefit of considering a few low frequency Fourier descriptors is the length reduction of the string.

It should be noted that using the orientation of the basic ellipse leads to an ambiguity of π radians and, therefore, rotation invariance is obtained only modulo π radians [3]. Fig. 3 illustrates this effect when normalizing the same shape at different scales, with different rotations, and with different starting points. This ambiguity can be solved, for example, by dividing the normalized shape in two regions (left and right of the vertical axis passing through the centroid), measuring mass and/or energy on each side and deciding that the right (or left) side should be the one of greatest value. However, the WARP system does not consider this problem. In any case, let us consider that rotation invariance has been really achieved. The basic idea of the WARP is that, after normalization, all shapes have a normalized version with a "standard starting point" and thus, are amenable to comparison by means of the DTW dissimilarity measure. But this is a flawed reasoning: invariance is only achieved for different translations, scalings, rotations, and starting point *of the same shape*. Different shapes (even similar ones) may differ substantially in their rotation and starting point. Fig. 4 shows two perceptually very similar figures (in fact, the second and third ones have been obtained from the first one by slightly compressing the horizontal axis) whose normalized version are significantly different in orientation and starting point. These problems appear frequently on shapes whose basic ellipse is nearly a circle. Invariance to rotation and starting point election should be provided by a different method.

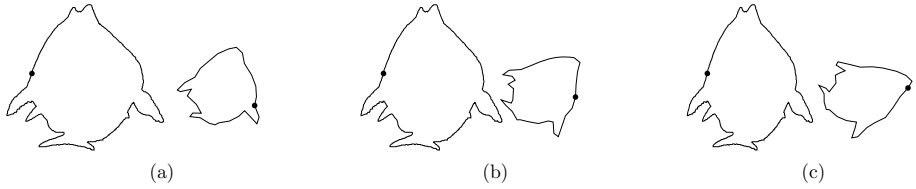


Fig. 4. (a) Original shape and its normalized version. (b) The same shape compressed in the X axis and its normalized version, which has a different rotation and starting point than the normalized version of the original shape. (c) The same shape a bit more compressed and its normalized version, which is different to the previous versions.

4 The Cyclic Dynamic Time Warping Dissimilarity

A *cyclic alignment* between $[A]$ and $[B]$ is a sequence of pairs (i_0, j_0), (i_1, j_1), $\ldots$, (i_{k-1}, j_{k-1}) such that, for $0 \leq \ell < k$, (a) $0 \leq i_\ell < m$ and $0 \leq j_\ell < n$; (b) $0 \leq i_{(\ell+1) \bmod m} - i_\ell \leq 1$ and $0 \leq j_{(\ell+1) \bmod n} - j_\ell \leq 1$; and (c) $(i_\ell, j_\ell) \neq (i_{(\ell+1) \bmod m}, j_{(\ell+1) \bmod n})$. The *weight of a cyclic alignment* (i_0, j_0), (i_1, j_1), $\ldots$, (i_{k-1}, j_{k-1}) is defined as $\sum_{0 \leq \ell < k} \gamma(a_{i_\ell}, b_{j_\ell})$.

Maes showed in [4] that, the Cyclic Edit Distance (CED), $\hat{D}([A], [B])$ can be defined in terms of the conventional Edit Distance (ED), $\hat{D}(A, B)$ as $\hat{D}([A], [B]) = \min_{0 \leq j < m} \min_{0 \leq k < n} \hat{D}(\sigma^j(A), B)$. Therefore, it can be computed in $O(m^2 n^2)$. He realized that $\hat{D}([A], [B]) = \min_{0 \leq j < m} \hat{D}(\sigma^j(A), B)$: only one of the two strings must be cyclically shifted in order to compute the CED, thus reducing time complexity to $O(m^2 n)$ time. This observation led to the design of a Divide-and-Conquer algorithm based on a non-crossing property of optimal edit paths on an extended edit graph that computes the CED in $O(mn \log n)$ time.

Let us try to adapt the derivation of Maes Algorithm for CED to the computation of the Cylic Dynamic Time Warping (CDTW) dissimilarity. First, we are going to show that the optimal cyclic alignment can be computed in terms of optimal alignments between non-cyclic strings.

Lemma 1. *If $m > 1$, $n > 1$, and (i_0, j_0), (i_1, j_1), $\ldots$, (i_{k-1}, j_{k-1}) is an optimal alignment between A and B, there is at least one ℓ such that $i_\ell \neq i_{(\ell+1) \bmod m}$ and $j_\ell \neq j_{(\ell+1) \bmod n}$.*

Proof: Any alignment including (i_ℓ, j_ℓ), $(i_\ell + 1, j_\ell)$, and $(i_\ell + 1, j_\ell + 1)$ can be "improved" by removing $(i_\ell + 1, j_\ell)$, since $\gamma(a_{i_\ell + 1}, b_{j_\ell}) \geq 0$. Analogously, any alignment including (i_ℓ, j_ℓ), $(i_\ell, j_\ell + 1)$, and $(i_\ell + 1, j_\ell + 1)$ can be "improved" by removing $(i_\ell, j_\ell + 1)$. □

Lemma 2. *The CDTW dissimilarity between $[A]$ and $[B]$, $D([A], [B])$, can be computed as $D([A], [B]) = \min_{0 \leq k < m} \min_{0 \leq \ell < n} D(\sigma^k(A), \sigma^\ell(B))$.*

Proof: The demonstration is trivial when $m = 1$ or $n = 1$. Let us consider that $m, n > 1$ and let (i_0, j_0), (i_1, j_1), $\ldots$, (i_{k-1}, j_{k-1}) be an optimal alignment. Let

ℓ be an index such that $i_\ell \neq i_{(\ell+1) \bmod m}$ and $j_\ell \neq j_{(\ell+1) \bmod n}$ (Lemma 1). The weight of this cyclic alignment is $D(\sigma^{(i_\ell+1) \bmod m}(A), \sigma^{(j_\ell+1) \bmod n}(B))$, which is considered by the double minimization. □

According to Lemma 2, the value of $D([A],[B])$ can be trivially computed in $O(m^2n^2)$ by solving mn recurrences like (1). Is it possible to perform cyclic shifts on only one of the strings? No: in general, $D([A],[B])$ is not $\min_{0\leq k<m} D(\sigma^k(A),B)$ or $\min_{0\leq k<n} D(A,\sigma^k(B))$, as the following counter-example shows: let $\Sigma=\{x,y\}$ and let $\gamma(\cdot,\cdot)$ be 0 if both arguments are equal, and 1 in other case; the value of $D([xyx],[yxy])$ is 0, since $D(xxy,xyy)=0$, but $D(xyx,yxy)=2$, $D(yxx,yxy)=1$, $D(xxy,yxy)=1$ $D(xyx,xyy)=1$, and $D(xyx,yyx)=1$. *Therefore, an equivalent of Maes' algorithm for CED cannot be directly applied to CDTW dissimilarity computation*, contrarily to what is supposed in [10].

Theorem 1. *The CDTW dissimilarity between cyclic strings* $[A]$ *and* $[B]$ *can be computed as* $D([A],[B]) = \min_{0\leq k<m}\big(\min(D(\sigma^k(A),B), D(\sigma^k(A)a_k,B))\big)$.

Proof: Each alignment induces a segmentation on A and a segmentation on B. All the symbols in a segment are aligned with the same symbol of the other cyclic sequence (Lemma 1). There is a problem when $b_{n-p-1}b_{n-p}\ldots b_{n-1}$ and $b_0b_1\ldots b_q$, for some p, $q\geq 0$, should belong to the same segment of B. In that case, the optimal path cannot be obtained by simply shifting A, since b_{n-1} must be aligned with the last symbol of $\sigma^k(A)$ and b_0 must be aligned with its first symbol, i.e., they cannot belong to the same segment. The string $\sigma^k(A)a_k$, formed by appending to $\sigma^k(A)$ its first symbol, permits to align $b_{n-p}b_{n-p+1}\ldots b_n$ and $b_1b_2\ldots b_q$ with the first symbol of $\sigma^k(A)$, since a_k also appears at the end of $\sigma^k(A)a_k$. □

Fortunately, for each value of k, $D(\sigma^k(A),B)$ can be obtained as a subproduct of the computation of $D(\sigma^k(A)a_k,B)$. The first warping graph of Fig. 6 shows that the warping graph underlying $D(\sigma^0(A),B)$ is a subgraph of the warping graph of $D(\sigma^0(A)a_0,B)$.

The value of $D(\sigma^k(A)a_k,B)$, for each k, can be obtained by computing a shortest path in an *extended warping graph* similar to the extended edit graph defined by Maes [4] (see Fig. 5 (a)). Since the non-crossing property of edit paths also holds for alignment paths (see Fig. 5 (b)), the Divide-and-Conquer approach proposed by Maes can be applied to CDTW. The reader is addressed to [4] to obtain a complete description of the Divide-and-Conquer procedure. It should be taken into account that, unlike in Maes' algorithm, the optimal path starting at $(k,0)$ can finish either at node $(k+m-1,n-1)$ or $(k+m,n-1)$. The running time of the algorithm is $O(mn\log n)$: each recursive step divides the search space in two balanced halves and all recursive operations at the same recursion require total $O(mn)$ time. Fig. 6 depicts this procedure.

5 Experiments

In order to assess the CDTW performance in classification and shape retrieval tasks, we have used the SQUID demo database, publicly available at the web-

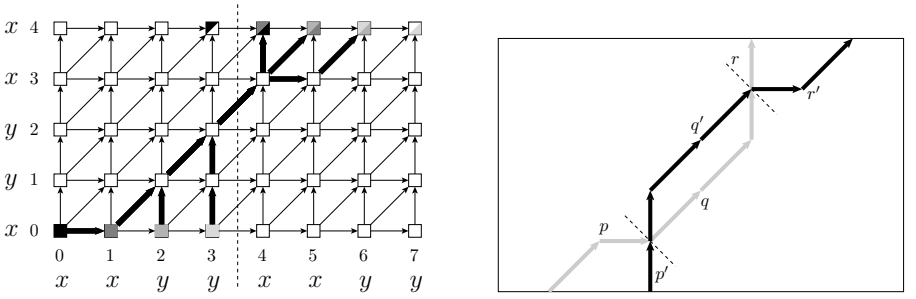


Fig. 5. (a) Extended warping graph for $A = xxyy$ and $B = xyyxx$. The optimal alignment for $[A]$ and $[B]$ is the optimal path starting from some colored node in the lower row and ending at node with the same color in the upper row. The optimal path departing at $(0,0)$ is $(0,0)$, $(1,0)$, $(2,1)$, $(3,2)$, $(4,3)$, $(4,4)$; departing at $(1,0)$: $(1,0)$, $(2,1)$, $(3,2)$, $(4,3)$, $(4,4)$; departing at $(2,0)$: $(2,0)$, $(2,1)$, $(3,2)$, $(4,3)$, $(5,4)$; and departing at $(3,0)$: $(3,0)$, $(3,1)$, $(3,2)$, $(4,3)$, $(5,3)$, $(6,4)$. (b) Optimal crossing paths can be avoided: if the weight of the subpath q is greater than the weight of the subpath q', the black path can be improved by traversing q' instead of q.

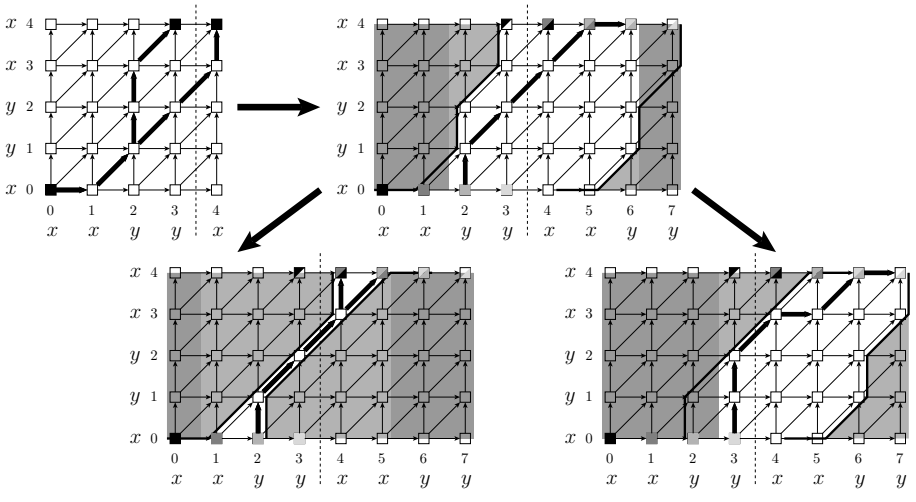


Fig. 6. Divide-and-Conquer procedure to compute the CDTW dissimilarity between $[A = xxyy]$ and $[B = xyyxx]$. First, the optimal alignment between A and B and between Aa_0 and B is computed. The optimal path underlying the alignment between $\sigma^0(A)$ and B (which is a displaced version of the optimal path between $\sigma^4(A)$ and B) is used as a frontier in the extended graph: only the white region must be explored to compute the optimal alignment between $\sigma^2(A)$ and B and between $\sigma^2(A)a_2$ and B. The optimal path between $\sigma^1(A)$ and B and $\sigma^1(A)a_1$ and B cannot cross the optimal path between $\sigma^0(A)$ and B and between $\sigma^2(A)$ and B. The optimal path between $\sigma^3(A)$ and B and $\sigma^3(A)a_3$ and B cannot cross the optimal path between $\sigma^2(A)$ and B and between $\sigma^4(A)$ and B.

site `http://www.ee.surrey.ak.uk/Research/VSSP/iimagedb/demo.html/`. It contains 1100 contours of fishes. Bartolini *et al.* manually labeled some of the contours as belonging to one of the following categories: seahorses (5 images), seamoths (6), sharks (58), soles (52), tonguefishes (19), crustaceans (4), eels (26), u-eels (25), pipefishes (16), and rays (41). Images not belonging to any category were assigned to a default class (848). In [1], Bartolini *et al.* present a shape retrieval experiment with the WARP system on this database. It should be noted that most shape fishes in the database have a basic ellipse with very different axis magnitudes, thus minimizing the non-invariance problems pointed out in Sect. 3.

In order to perform the same shape retrieval experiment with CDTW, each shape has been smoothed with a gaussian kernel (stdev=10) in order to reduce discretization noise and the curvature of each contour has been computed at each point t as $\kappa(t) = (\dot{X}(t)\ddot{Y}(t) - \ddot{X}(t)\dot{Y}(t))/(X(t)^2 + Y(t)^2)^{3/2}$. The length of the string was reduced by considered the average curvature at each four contiguous points. The curvature is invariant with respect to translation and rotation. The γ function has been defined as $\gamma(\kappa_A(t), \kappa_B(t')) = \sqrt{|\kappa_A(t) - \kappa_B(t')|}$.

Fig. 7 shows the precision/recall graph on the whole dataset for both the WARP system (extracted from Fig. 6 (a) of [1]) and the CDTW dissimilarity shape retrieval system. It can be seen that the CDTW system outperforms the WARP system.

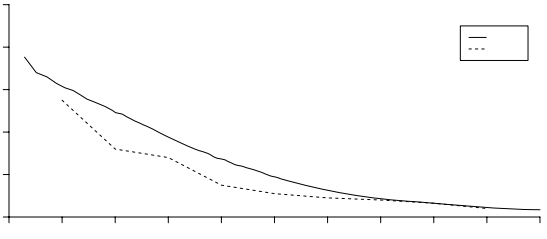


Fig. 7. Precision P (number of relevant shapes among the retrieved ones, in %) as a function of recall R (% of relevant shapes recovered w.r.t. all relevant shapes in the database)

6 Conclusions

We have studied the use of DTW dissimilarity measure to compare strings defining the contour of shapes. We have considered the WARP system, which has been recently presented in [1], and pointed out some of its deficiencies: mainly, its difficulty in providing rotation and starting point invariance for perceptually similar shapes.

The Cyclic Dynamic Time Warping dissimilarity has been defined and an algorithm to compute it in $O(mn \log n)$ has been presented. The method is based on the Cyclic Edit Distance algorithm proposed by Maes. We have shown that Maes'

algorithm cannot be directly applied to cyclic DTW dissimilarities computation by just replacing the edit operations by alignments of symbols: two conventional DTW dissimilarities must be computed for each cyclic shift of one string. Fortunately, one of these dissimilarities can be obtained as a subproduct of the other.

The Cyclic Dynamic Time Warping dissimilarity has been applied to a shape classification task with a publicly available database and it has shown to outperform the WARP system.

Acknowledgments

The authors wish to thank S. Abbasi, F. Mokhtarian, and J. Kittler for making the SQUID database publicly available. They also thank I. Bartolini, P. Ciaccia, and M. Patella for providing their labeling of the SQUID database.

References

1. I. Bartolini, P. Ciaccia, and M. Patella. WARP: Accurate Retrieval of Shapes Using Phase of Fourier Descriptors and Time Warping Distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(1):142–147, 2005.
2. H. Bunke and H. Bühler. Applications of Approximate String Matching to 2D Shape Recognition. *Pattern Recognition*, 26(12):1797–1812, 1993.
3. A. Folkers and H. Samet. Content-based Image Retrieval Using Fourier Descriptors on a Logo Database. In *Proc of the 16th Int Conf on Pattern Recognition*, pages 521–524, 2002.
4. M. Maes. On a Cyclic String-to-String Correction Problem. *Information Processing Letters*, 35:73–78, 1990.
5. M. Maes. Polygonal Shape Recognition using String Matching Techniques. *Pattern Recognition*, 24(5):433–440, 1991.
6. A. Marzal and S. Barrachina. Speeding up the computation of the edit distance for cyclic strings. *Int. Conference on Pattern Recognition*, pages 271–280, 2000.
7. R. A. Mollineda, E. Vidal, and F. Casacuberta. *Efficient Techniques for a very Accurate Measurement of Dissimilarities between Cyclic Patterns*, volume 1876, pages 121–126. Springer, 2000.
8. G. Peris and A. Marzal. Fast Cyclic Edit Distance Computation with Weighted Edit Costs in Classification. *Proc. Int. Conf. on Pattern Recognition*, 4, 2002.
9. D. Sankoff and J. Kruskal, editors. *Time warps, string edits, and macromolecules: the theory and practice of sequence comparison.* Addison-Wesley, Reading, MA, 1983.
10. T. B. Sebastian, P. N. Klein, and B. B. Kimia. On aligning curves. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(1):116–125, 2003.
11. R.A. Wagner and M.J. Fischer. The String-to-String Correction Problem. *Journal of ACM*, 21(1):168–173, 1974.
12. D. Zhang and G. Lu. A Comparative Study of Fourier Descriptors for Shape Representation and Retrieval. In *5th Asian Conf. on Computer Vision*, Jan 2002.

Diagnosing Errors in DbC Programs Using Constraint Programming

R. Ceballos, R. M. Gasca, C. Del Valle, and D. Borrego

Departamento de Lenguajes y Sistemas Informáticos,
Universidad de Sevilla (Spain)
{ceballos, gasca, carmelo, borrego@lsi.us.es}

Abstract. Model-Based Diagnosis allows to determine why a correctly designed system does not work as it was expected. In this paper, we propose a methodology for software diagnosis which is based on the combination of Design by Contract, Model-Based Diagnosis and Constraint Programming. The contracts are specified by assertions embedded in the source code. These assertions and an abstraction of the source code are transformed into constraints, in order to obtain the model of the system. Afterwards, a goal function is created for detecting which assertions or source code statements are incorrect. The application of this methodology is automatic and is based on Constraint Programming techniques. The originality of this work stems from the transformation of contracts and source code into constraints, in order to determine which assertions and source code statements are not consistent with the specification.

1 Introduction

In recent decades, the Model-Based Diagnosis community has dedicated big efforts to the development of a methodology for the diagnosis of industrial systems based on integrated sensors which are supposed to work correctly. The diagnosis aim is to detect and to isolate the reason of an unexpected behaviour; in other words, to identify the parts which fail in a system. In order to explain a wrong behaviour, the diagnosis process uses a set of observations and a model of the system. In this work, a methodology for diagnosing software is proposed, that is, for isolating errors in programs.

During the recent decades, the number of techniques for automatically testing and debugging software has increased substantially. The testing techniques allow to detect if there are errors in a software development, but it is difficult to isolate the errors if only testing techniques are used. The debugging techniques allow to isolate the errors of a program in an interactive way. Our objective is the application of the Model-Based Diagnosis techniques to a program, in order to isolate the errors of a program.

In order to obtain the errors of a program, we have considered the deep integration between the different areas derived from Software Engineering (Design by Contract and Testing techniques) and Artificial Intelligence (Model-Based Diagnosis and Constraint Programming). Our software diagnosis methodology

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 200–210, 2006.

has two different steps: first, it is necessary to capture the specification of the correct behaviour of a program (using DbC, testing or expert information); and second, it is necessary to isolate the error (using Model-Based Diagnosis and Constraint Programming techniques).

The main idea is to transform the contracts and source code into an abstract model based on constraints, in a Max-CSP (Maximal Constraint Satisfaction Problem). This model enables the detection of errors in contracts and/or in a source code. A Max-CSP is a framework for modelling and solving real problems as a set of constraints among variables, and a goal function for satisfying the maximum number of constraints.

Design by Contract (DbC) was proposed in [1]. DbC improves the software quality. A previous paper [2] proposed two measures in order to validate the benefits of using DbC: robustness and diagnosability. The robustness is the degree which the software is able to recover from internal faults that would otherwise have provoked a failure. Diagnosability expresses the effort required in the localization of a fault as well as the preciseness allowed by a test strategy on a given system. The results show that robustness and diagnosability improve rapidly with only a few contracts, and for improving the diagnosability, the quantity of the contracts is less important than their quality.

There are different techniques to automate the testing and debugging of a program, such as, slicing techniques [3], model-based debugging[4], model checking [5] or delta debugging [6]. Our proposal is different from these techniques, because we use DbC to obtain a more precise location of the errors in a program. The DbC specification is also used in other techniques for the verification of the component-based programs. The objective of our methodology is not only the verification of the software; our goal is to obtain the isolation of errors in a program, what is a diagnosis methodology. This paper is an improvement and an extension of our own previous work [7].

The remainder of the paper is organized as follows. Section 2 shows the diagnosis framework. Section 3, 4 and 5 explain the framework modules. Finally, conclusions are drawn and future work is outlined.

2 Diagnosis Framework

Figure 1 represents the complete diagnosis process. The process is based on three modules. The Abstract Model Generation (AMG) module receives the source code and contracts of a program and obtains the abstract model. The abstract model is a set of constraints.

The Error Detection (ED) module detects the errors of an executed program. The following definitions specify the kind of errors that this module can detect.

Definition 1. An *infeasible assertion* is a non-viable assertion due to conflicts with previous assertions, fields or variables values. The set of assertions of a contract is verified when a program is executed. An infeasible assertion is a

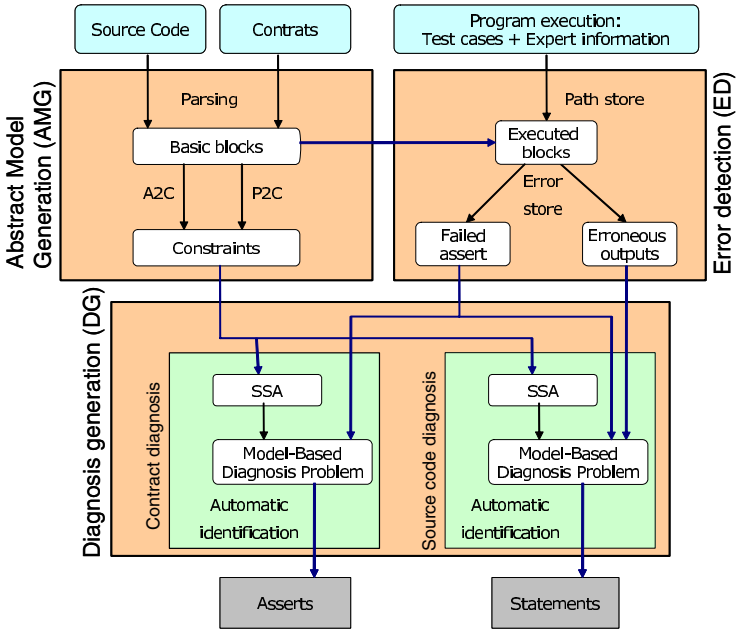


Fig. 1. Diagnosis framework

wrongly designed assertion that cannot be satisfied, and it stops the program execution when it is not necessary.

Definition 2. A *bug* is a statement or a set of statements that does not allow us to obtain the correct result. This paper does not consider the errors detected in compilation time (such as syntax errors), nor dynamic errors (such as exceptions, memory access violations, infinite loops, etc). The bugs under consideration are minor variations of the correct program, such as errors in assignment statements, or errors in the conditions of looping statements or conditional statements.

The ED module receives the set of basic blocks obtained in the AMG module. In our approach, when a program is executed, the information of the executed basic blocks is stored by the ED module. This information will be used for detecting the errors and for diagnosing the system.

The Diagnosis Generation (DG) module obtains the diagnosis for a failed assert or erroneous outputs. The diagnosis of a program will be a set of infeasible assertion and/or bugs. The following subsections describe the modules of the diagnosis framework.

3 Abstract Model Generation (AMG)

In Model-Based diagnosis approaches, a model of the system and a set of observations enable detecting and isolating the reasons of an erroneous behaviour

of a system. The system model simulates the components and their connections by using constraints where the variables represent the inputs and outputs of components.

In an Object-Oriented (OO) program the methods of different objects are linked to obtain an specified behaviour. Each method of an object can be considered as a component which generates a result depending on the input values, the field values, and the statements of the method (which can be considered as subcomponents). The pre-treatment of the source code and program contracts enables to obtain a model of a program. The following subsections show the process for generating this model.

3.1 Determining Basic Blocks

Every OO program is a set of classes. A class is made of methods, fields, assertions, etc. In order to automate the diagnosis of a program, it is necessary to divide the system into subsystems. Each class is transformed into a set of basic blocks (BB) by using a parsing program. These basic blocks can be: blocks of invariants, blocks of static class fields, blocks of object attributes, and blocks of object or class methods.

- Block of invariants (IB): It includes the set of invariants of a class.
- Block of static fields of a class (SB): It includes the set of static field declarations and static code blocks of a class.
- Block of object fields (OB): It includes the set of object field declarations of a class.
- Block of class method or object method (MB): For each constructor or method of a class, a block is obtained. This block is the set of all the statements and assertions (such as preconditions, postconditions or loop invariants) which are included in the method. If the method is static, the block is named a class method block; otherwise it is named an object method block.

Each program class can be transformed into a set of basic blocks (BB_s) equivalent to the $Class_i$. Each method-block is a set of sub-blocks such as conditional blocks (conditional statements) or loop blocks (loops). In our methodology the assignments are defined as the indivisible blocks.

3.2 Program Transformation

The abstract model is a set of constraints which represents the behaviour of the contracts (assertions) and the source code (statements) of a program. Our approach uses the function *A2C* (assertion to constraints) for transforming an assertion into constraints. The transformation of the source code into constraints appeared in a previous work [7]. The process is based on the transformation of statements of each basic block. Our approach uses the function *P2C* (program to constraints) for transforming a source code into constraints. The main ideas of the transformation are:

- Indivisible blocks:
  Assignments: {*Ident := Exp*}
  The assignment statement is transformed into the following equality constraint: {Ident = Exp}. If the assignment statement is not a part of the minimal diagnosis, then the equality between the assigned variable and the assigned expression must be satisfied.
  Method calls and return statements: For each method call, the constraints defined in the precondition and postcondition of the method are added. The method calls allow to link basic blocks and to obtain the order of the blocks in the program execution.
- Conditional blocks: {*if (cond)* {If_{Block}} *else* {$Else_{Block}$}}
  There are two possible paths in a conditional statement depending on the inputs of the condition. The constraints of a conditional statement include the condition and the inner statements of the selected path (only one of the two possible paths is executed).
- Loop blocks: {*while (cond)* {$Block_{Loop}$}}
  In a loop, the number of iterations depends on the inputs. Each iteration is transformed into a conditional statement. The structure of a loop is simulated as a set of nested conditional statements. In [7] a method is proposed to reduce the model to less than n iterations, but if the invariant of the loop exists, the diagnosis process is more precise.

4 Error Detection (ED)

This module detects the errors of an execution of a program. This module can detect failed assertions or erroneous outputs. A failed assertion is an assertion which stops the program due to the conflicts with previous assertions, fields or variables values. The erroneous outputs are detected when outputs are different of the correct results. These correct results must be specified by contracts, test cases or expert information. The Error Detection (ED) module receives the set of basic blocks obtained by the AMG module. At this point, it is important to introduce one concept from the imperative programming.

Definition 3. A *Path* is the sequence of statements that are executed. The executed path of a program depends on the inputs. Conditional statements, loop statements and method calls enable the incorporation of more or less statements to a program execution.

In a Object-Oriented Imperative Language, a path of a program is a sequence of basic blocks obtained by linking constructors and method calls. In our approach, the diagnosis of the program is based on a model generated by linking the constraints obtained from the basic blocks of the executed path.

The erroneous outputs are detected by using DbC specification, test cases and expert information. If the DbC is too weak, an expert can decide which should be the correct inputs and outputs. Testing techniques enable the selection of the

most significant observations in order to detect errors in programs. In [8], the objectives and complications of good Testing are set out.

Definition 4. A *Test case* (TC) is a set of inputs (class fields, parameters or variables), execution preconditions, and expected outcomes, which are developed for a particular objective, such as to exercise a particular program path or to verify the compliance with a specific requirement.

When a program is executed by using a test case, the information about the executed basic blocks is stored, such as the executed path, values of the variables, and the results of the assertion evaluations. This information is necessary for the following Diagnosis Generation module.

5 Diagnosis Generation (DG)

The DG module receives the errors detected by the ED module. These errors can be failed assertions or erroneous outputs. If the error is a failed assertion, the cause of the errors can be infeasible assertions and/or bugs. If the error is an erroneous output, the cause of the problem must be due to bugs. The following subsection shows the transformation of an abstract model into a diagnosis problem in order to detect infeasible assertions and/or bugs.

5.1 Diagnosis Problem

A diagnosis is a hypothesis about which changes are necessary in a program to obtain a correct behavior. The definition of diagnosis, in Model Based Diagnosis (MDB), is built up from the notion of the abnormal predicate [9]: $AB(c)$ is a boolean variable which holds when a component c of the system is abnormal. For example, an adder component is abnormal if the output of the adder is not the sum of its inputs. A diagnosis specifies whether each component of a system is abnormal or not. In order to clarify the diagnosis process, some definitions must be established.

Definition 5. *System model* (SM) is a tuple {PD(PC), TC} where: PC are the program components, that is, the finite set of statements and asserts of a program; PD is the program (statements and asserts) description, that is, the set of constraints obtained of the PC; and TC is a test case.

Definition 6. *Diagnosis*: Let $\mathcal{D} \subseteq$ PC, $\mathcal{D}$ is a diagnosis if PD' $\cup$ TC is satisfiable, where PD' = PD(PC $-$ $\mathcal{D}$).

Definition 7. *Minimal Diagnosis* is a diagnosis $\mathcal{D}$ that for no proper subset $\mathcal{D}$' of $\mathcal{D}$, $\mathcal{D}$' is a diagnosis. The minimal diagnosis implies to modify the smallest number of program statements or assertions.

The goal is to identify the minimal diagnosis consistent with a test case. The constraints of the PD of a program will be obtained as it was explained in

Table 1. Program model of the modified Toy problem

TC	PC	PD
Inputs : {a = 3, b = 2, c = 2, d = 3, e= 3}	S1 : int x = a * c	(AB(S1) ∨ (x == a * c))∧
Outputs : {f = 12, g = 12}	S2 : int y = b * d	(AB(S2) ∨ (y == b * d))∧
Test Code: S1 .. S5	S3 : int z = c + e	(AB(S3) ∨ (z == c + e))∧
	S4 : int f = x + y	(AB(S4) ∨ (f == x + y))∧
	S5 : int g = y + z	(AB(S5) ∨ (g == y + z))∧
	Post : f = a * c + b * d	(f == a * c + b * d)∧
	∧ g = b * d + c * e	(g == b * d + c * e)

section 3. When a program is executed, the order of the assertions and statements is very important. It is necessary to maintain this order in the abstract model (based on constraints). Hence, the program under analysis is transformed into a static single assignment (SSA) form. In SSA form, only one assignment is made to each variable in the whole program. For example, the code *x=a*c; ...x=x+3;... {Post:x =...}* is changed to *x1=a*c; ...x2=x1+3;... {Post:x2 =...}*. More details about SSA form are shown in [10].

Table 1 shows the PD of the toy program derived from the standard toy problem used in the diagnosis community [9]. The program does not reach the correct output because the third statement is an adder instead of a multiplier. In order to obtain the minimal diagnosis a Maximal Constraint Satisfaction Problem (Max-CSP) is generated. A Max-CSP is a CSP with a goal function to maximize. The objective is to find an assignment of the AB variables that satisfies the maximum number of the PD constraints: Goal Function = Max(N $AB(\mathrm{i}) : AB(\mathrm{i}) = \mathit{false}$). The diagnosis process by using a Max-CSP was shown in a previous work [11]. For example, by using a Max-CSP, the minimal diagnoses for the toy program would be: {{S3}, {S5}, {S1, S2}, {S2, S4}}.

5.2 Diagnosing Contracts

The assertions can be checked by using test cases or not.

- **Diagnosis of assertions without using test cases:** Two kinds of checks are proposed at this point:
  - Checking the invariants of a class: The invariants must always be satisfied. Hence, a Max-CSP is generated by using the invariants of each class in order to check if all the invariants of a class can be satisfied together.
  - Checking the assertions of the methods: The precondition and postcondition of a method must be feasible with the invariants of a class. In order to detect conflicts between the precondition or the postcondition with the invariants, a Max-CSP is generated by using the constraints associated with the assertions.

  The solutions of these Max-CSP problems enable the verification of the feasibility of assertions.

```
/**
 *  @inv getBalance()>= 0
 *  @inv getInterest >= 0
 */
public interface Account {

    /**
     *  @pre income > 0
     *  @post getBalance() >= 0
     */
    public void deposit (double income);

    /**
     *  @pre withdrawal > 0
     *  @post getBalance() ==
     *      getBalance()@pre - withdrawal
     */
    public void withdraw (double withdrawal);

    public double getBalance ();

}
```

```
public class AccountImp implements Account {

    private double interest;

    private double balance;

    public AccountImp()  {
       this.balance = 0;
    }

    public void deposit (double income) {
       this.balance = this.balance - income;
    }

    public void withdraw (double withdrawal) {
       this.balance = this.balance - withdrawal;
    }

    public double getBalance() {
       return this.balance;
    }
}
```

Fig. 2. Interface *Account* and class *AccountImp* source code

Table 2. Diagnosis of the method *Withdraw* by using a test case

TC	Inputs:	{balance@pre = 0, withdrawal = 100}
	Outputs:	{balance = 0}
	Test code:	Method Withdraw
PD	Inv.	(AB(Inv) ∨ (balance@pre >= 0)) ∧
	Pre.	(AB(Pre) ∨ (withdrawal > 0)) ∧
	Post.	(AB(Post) ∨ (balance = balance@pre - withdrawal)) ∧
	Inv.	(AB(Inv) ∨ (balance >= 0))

– **Diagnosis of assertions by using test cases:** It is possible to obtain more information about the viability of the method assertions by applying test cases to the sequence {invariants + precondition + postcondition + invariants } in each method.

Example. In order to clarify the methodology, the class *AccountImp* is used. This class implements the interface *Account* that simulates a bank account. It is possible to deposit money and to withdraw money. Figure 2 shows the source code. The method *deposit* has a bug, because it *decreases* the account balance.

Table 2 shows the PD for the method *withdraw*. In this example there are four constraints: the invariants before and after the method, the precondition, and the postcondition. The test case specifies that the initial and final balance of the account must be 0, when a positive amount is withdrawn.

The invariant specifies that the balance must be equal or greater than zero when the method finishes, but if this invariant is satisfied, it implies that the precondition and the postcondition could not be satisfied together. The postcondition implies that *balance* = *balance@pre* - *withdrawal*, that is, $0-withdrawal > 0$, and this is impossible if the *withdrawal* is positive. The error stays in the precondition, since this precondition is not strong enough to stop the program execution when the withdrawal is not equal or greater than the balance.

5.3 Diagnosing Source Code (with Assertions)

As we are looking for the minimal diagnosis, this work proposes to use a Max-CSP in order to maximize the number of satisfied constraints of the PD. The constraint obtained by the assertions must be satisfied, because these constraints give us information about the correct behaviour. But the constraints obtained from the source code can be satisfied or not. The result of the diagnosis process depends on the outputs obtained by using the test case and the final state of the program:

- State 1: If the program ended up with a failed assertion, and did not reach the end as specified in the test case, then the error can be a strict assertion (the assertion is very restrictive) or one or more bugs before the assertion. In order to determine the cause of the problem, the program should be executed again without the assertion, in order to check if the program can finish without the assertion.
- State 2: If the program ends, but the result is not the one specified by the test case, then the error can be a bug, or an assertion which is not enough restrictive (this enables executing statements which obtain an incorrect result). If the error is a bug, the resolution of the Max-CSP provides the minimal diagnosis that includes the bug. If the error is due to a weak assertion, then a deeper study of the assertions is necessary. At present, we are researching this point.

Example. In order to clarify the methodology, an example is proposed in Table 3. This is an account with an initial balance of 300 units. Two sequential operations are applied (a withdrawal of 300 units and a deposit of the same quantity). The final balance should be 300 units. The constraints solver determines that the error is caused by the statement included in the method *deposit*. If the method is examined closely, it can be seen that there is a subtraction instead of an addition. The postcondition of this method was too weak, and did

Table 3. Class *AccountImp* checking by using a test case

TC	Inputs:	{balance@pre = 300, withdrawal = 300, income = 300}
	Outputs:	{balance = 300}
	Test code:	S1: account.withdraw(withdrawal) S2: account.deposit(income)
PD	Inv.	balance0 >= 0 ∧
	Pre.	withdrawal > 0 ∧
	Code	(AB(S1) ∨ (balance1 = balance0 - withdrawal)) ∧
	Post.	balance1 = balance0 - withdrawal ∧
	Inv.	balance1 >= 0 ∧
	Pre.	income > 0 ∧
	Code	(AB(S2) ∨ (balance2 = balance1 - income)) ∧
	Post.	balance2 >= 0 ∧
	Inv.	balance2 >= 0

not permit to detect the error. The statement included in the method *withdraw* influences the final result of the balance; however, it is not a possible bug because of the postcondition, since it is strong enough to guarantee the balance value at the end of the method. We can conclude that if the contracts are strong enough the diagnosis will be better.

6 Conclusion and Future Work

This paper is an improvement of a previous work [7] in order to automate the diagnosis of DbC software. This approach incorporates a more precise way to diagnose software since more DbC characteristics are incorporated. As shown by the studied examples, the stronger the contracts are, the better the diagnosis is, because the framework has more information of the expected behaviour.

A more complex diagnosis process is being developed in order to obtain a more precise minimal diagnosis. We are extending this methodology to include all the characteristics of an Object-Oriented language, such as inheritance, exceptions and concurrence. Another important direction is the application of the methodology to more complex and real examples where the transformation of the system into constraints is less straightforward.

Acknowledgements

This work has been funded by the Spanish Ministry of Science and Technology (DPI2003-07146-C02-01) and the European Regional Development Fund (ERDF/ FEDER).

References

1. Meyer, B.: Applying design by contract. IEEE Computer **25** (1992) 40–51
2. Traon, Y.L., Ouabdesselam, F., Robach, C., Baudry, B.: From diagnosis to diagnosability: Axiomatization, measurement and application. The Journal of Systems and Software **1** (2003) 31–50
3. Weiser, M.: Program slicing. IEEE Transactions on Software Engineering **4** (1984) 352–357
4. Mateis, C., Stumptner, M., Wieland, D., Wotawa., F.: Model-based debugging of java programs. In: AADEBUG. (2000)
5. Groce, A., Visser, W.: Model checking java programs using structural heuristics. In: ISSTA '02: Proceedings of the 2002 ACM SIGSOFT international symposium on Software testing and analysis, Roma, Italy (2002) 12–21
6. Cleve, H., Zeller, A.: Locating causes of program failures. In: 27th International Conference on Software Engineering (ICSE 2005), St. Louis, Missouri (2005)
7. Ceballos, R., Gasca, R.M., Valle, C.D., Rosa, F.D.L.: A constraint programming approach for software diagnosis. In: AADEBUG Fith International Symposium on Automated and Analysis-Driven Debugging, Ghent, Belgium (2003) 187–196
8. Binder, R.: Testing Object-Oriented Systems : Models, Patterms, and Tools. Object Technology Series. Addison Wesley (2000)

9. de Kleer, J., Mackworth, A., Reiter, R.: Characterizing diagnoses and systems. Artificial Intelligence **2-3** (1992) 197–222
10. Alpern, B., Wegman, M., Zadeck, F.K.: Detecting equality of variables in programs. In: Proceedings of the 15th ACM SIGPLAN-SIGACT symposium on Principles of programming languages, San Diego, California (1988) 1–11
11. Ceballos, R., del Valle, C., Gómez-López, M.T., Gasca, R.M.: CSP aplicados a la diagnosis basada en modelos. Revista Iberoamericana de Inteligencia Artificial **20** (2003) 137–150

Early Fault Classification in Dynamic Systems Using Case-Based Reasoning

Aníbal Bregón[1], M. Aránzazu Simón[1], Juan José Rodríguez[2], Carlos Alonso[1], Belarmino Pulido[1], and Isaac Moro[1]

[1] Intelligent Systems Group (GSI), Department of Computer Science, E.T.S.I. Informática, Univ. of Valladolid, Campus Miguel Delibes s/n, 47011 Valladolid, Spain
anibal@infor.uva.es, {arancha, calonso, belar, isaac}@infor.uva.es
[2] Lenguajes y Sistemas Informáticos, Univ. of Burgos, Burgos, Spain
jjrodriguez@ubu.es

Abstract. In this paper we introduce a system for early classification of several fault modes in a continuous process. Early fault classification is basic in supervision and diagnosis systems, since a fault could arise at any time, and the system must identify the fault as soon as possible. We present a computational framework to deal with the problem of early fault classification using Case-Based Reasoning. This work illustrates different techniques for case retrieval and reuse that have been applied at different times of fault evolution. The technique has been tested for a set of fourteen fault classes simulated in a laboratory plant.

1 Introduction

In this article we present an exploratory work about using Case Based Reasoning (CBR) methodology for fault classification. In previous works, we have tested different Artificial Intelligence techniques (Knowledge-based systems and machine learning) to improve model-based diagnosis [2,17]. Now we are studying CBR suitability for that purpose.

Case Based Reasoning is an Artificial Intelligence methodology for solving problems where a model of the domain knowledge is not available, and which exhibits basic machine learning capabilities. In the plant operation context, several works have been developed using CBR for different tasks [6]: Situation Assessment [3,7,8], Planning, Diagnosis, Troubleshooting, Maintenance, and Quality Management. To perform fault diagnosis, in dynamic systems a CBR system must be able to classify series with different length, due to the fact that different faults have to be identified as soon as possible, and the symptom dynamics are different.

In this work, we apply CBR methodology to the problem of fault classification in a laboratory plant. Our dataset is made up of time series corresponding with historic values of the set of observed variables of the plant. The CBR system works in parallel with a model-based diagnosis system which performs fault detection and localization [15]. Once a fault is detected, the CBR system provides a hint on the most probable fault candidate.

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 211–220, 2006.

We have considered fourteen classes of faults. Our CBR system offers several options for facing some steps of the CBR cycle. Different retrieving methods have been implemented and compared.

The organization of this article is as follows. First, we will describe the CBR methodology, the retrieving algorithm and the similarity measures. Later on, we briefly introduce the laboratory plant and its classes of faults. Afterwards, we show and discuss the results. Finally, we draw some conclusions and propose some future work.

2 The Case-Based Reasoning System

Case-Based Reasoning is a way to solve problems by remembering past similar situations and reusing the information and knowledge about these situations [10,11]. CBR uses the information stored in a case base to infer the solution for new problems.

CBR proposes a four-step cycle, described in [1] as: **Retrieve, Reuse, Revise, Retain.**

2.1 Case-Based Reasoning for Early Classification

CBR has been used for diagnosis tasks in different domains ([12] and [13] provide several examples). More precisely we want to do diagnosis as classification of time series (as in [7]). In previous works we introduced a CBR system which achieves good classification success percentages for "post-mortem" fault classification [5]. In our system we relay upon a model-based diagnosis system for fault detection and localization, we are just concerned with an early diagnosis problem. As opposed to "post-mortem" diagnosis, early diagnosis of time series data means fault classification with incomplete data. If symptoms exhibit different dynamics, they will manifest at different times. Hence, faults can only be completely identified when the whole series are available. Consequently, with incomplete data, different faults can be consistent with current observations. In this context, CBR must provide a set of feasible faults, just after fault detection, according to available observations.

On using CBR for early classification we get some advantages compared with other classifiers. A CBR system is able to incorporate new cases in the system at any time, without recalculating all of the classification rules. Another important advantage is that we can use different length series without training the system for each length.

2.2 The Case-Based Reasoning Architecture

As we want to implement a CBR system to perform fault classification in dynamic systems, our proposal is to adapt the CBR stages to deal with temporal information. The developed system implements several techniques for the retrieving and the reusing steps of the CBR cycle.

The first task in the CBR cycle is to **retrieve** one or more similar cases from the case library where former experiences are stored. Hence, it is necessary to have a retrieving algorithm and a similarity measure that will be used to bring back a set of similar cases. In this work, we have chosen the K-Nearest Neighbors algorithm as retrieving algorithm and three different similarity measures:

- **Euclidean distance**
- **Manhattan distance**
- **Dynamic Time Warping (DTW)** [8,9]**:** is a technique that allows to obtain a dissimilarity measure between two sequences with different lengths which are not exactly aligned in the time axis. In the considered application the time series are multivariate. In order to use DTW with this kind of data we consider a one-dimensional approach for calculating the similarity between two series. We apply DTW for each variable. The similarity between two multivariate series is the average of the similarities for each variable. In order to work out the distance between two variables we use three different kinds of metrics: **Linear**, **Quadratic** and **Valley**.

The reasons why we have selected these measures are:

- **Euclidean distance:** We decided to use it because it is the most common distance measure in numerical domains, and it is used in most of the CBR systems.
- **Manhattan distance** can obtain good results with very low computational cost.
- **Dynamic Time Warping:** DTW is a robust dissimilarity measure for time series. We expect results using DTW to be better than those obtained using the Euclidean and Manhattan distances.

The **reuse** [1,18] method used in this work is based on the K-Nearest Neighbors. We can choose the number of neighbors to be used and select the solution for the current case voting among the solutions for each retrieved neighbor.

2.3 Case Representation

All the cases stored in the case base are made up of several time series describing faults in a laboratory plant. In our CBR System, a case is composed by the temporal data series from sensors and the class of the fault that will be the solution our system tries to adapt. Additional details about the cases and the fault classes will be provided once we introduce the case study in Section 3.

3 Plant Description

The laboratory plant shown in figure 1 resembles common features of industrial continuous processes. It is made up of four tanks $\{T_1, \ldots, T_4\}$, five pumps $\{P_1, \ldots, P_5\}$, and two PID controllers acting on pumps P_1, P_5 to keep the level

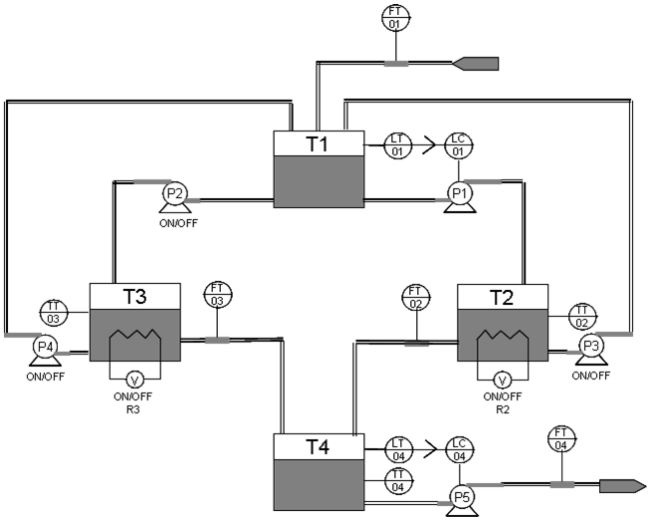


Fig. 1. The diagram of the plant

of $\{T_1, T_4\}$ close to the specified set point. To control temperature on tanks $\{T_2, T_3\}$ we use two resistors $\{R_2, R_3\}$, respectively.

In this plant we have eleven different measurements: levels of tanks T_1 and T_4 –$\{LT01, LT04\}$–, the value of the PID controllers on pumps $\{P_1, P_5\}$ –$\{LC01, LC04\}$–, in-flow on tank T_1 –$\{FT01\}$–, outflow on tanks $\{T_2, T_3, T_4\}$ –$\{FT02, FT03, FT04\}$–, and temperatures on tanks $\{T_2, T_3, T_4\}$ –$\{TT02, TT03, TT04\}$–. Action on pumps $\{P_2, P_3, P_4\}$, and resistors –$\{R_2, R_3\}$– are also known.

This plant can work on different situations. We have defined three working situations which are commanded through four different operation protocols. In the operation protocol used in this article resistor R_3 is switch off, while resistor R_2 is on. Also, pumps $\{P_3, P_4\}$ are switch off; hence, just flow $FT01$ is incoming to tank T_1.

3.1 Fault Modes

We consider the fault classes described in table 1. Summarizing we have fourteen different faults for the current protocol:

- Tank leakages: 5 classes of faults. We have considered only 4 tanks but in tank $T1$ we can distinguish between small and big leakages.
- Pipe blockages: 5 classes of faults.
- Pump failures: 3 classes of faults.
- Resistor failures: Only 1 class of fault.

Figure 2 shows evolution of one example for each class. Each row shows a case of a different class of fault. Each column shows one of the observed variables. All charts corresponding with the same type of magnitude (for example, temperatures) are show in the same scale.

Table 1. Classes of faults considered

Class	Component	Description
FM01	T1	Small leakage in tank T1
FM02	T1	Big leakage in tank T1
FM03	T1	Pipe blockage T1 (left outflow)
FM04	T1	Pipe blockage T1 (right outflow)
FM05	T3	Leakage in tank T3
FM06	T3	Pipe blockage T3 (right outflow)
FM07	T2	Leakage in tank T2
FM08	T2	Pipe blockage T2 (left outflow)
FM09	T4	Leakage in tank T4
FM10	T4	Pipe blockage T4 (right outflow)
FM11	P1	Pump failure
FM12	P2	Pump failure
FM13	P5	Pump failure
FM14	R2	Resistor failure in tank T2

4 Experimental Design and Results

The study was made on a data-set, made up of several examples obtained from simulations of the different classes of faults that could arise in the plant. We have modeled each fault class with a parameter in the $[0, 1]$ range. We performed twenty simulations for each class of fault. Each simulation lasted 900 seconds. We randomly generate the fault magnitude, and its origin, in the interval $[180, 300]$. We also have assumed that the system is in stationary state before the fault appears.

The number of samples per fault class was selected based on a trade-off between three factors:

- experience with other machine learning techniques used for classification, which demanded a high number of class instances to improve successful classification rates,
- the cost of obtaining a big number of simulations, with random fault parameters and initial conditions,

Hence, twenty simulations was a compromise between the desired number of instances for classification techniques and the cost of the simulations.

The data sampling was one data per second. However, due to the slow dynamics in the plant, we can select one data every three seconds without losing discrimination capacity. Since we just have eleven measures, then each simulation will provide eleven series of three hundred numeric elements. Moreover, decreasing the sampling we reduce the processing time for DTW (which is quadratic).

To perform early classification we use variable size windows. The size of each window will be increased as time elapses. Therefore, accuracy increases with time. Hence, our CBR system will be able to classify time series data with different lengths.

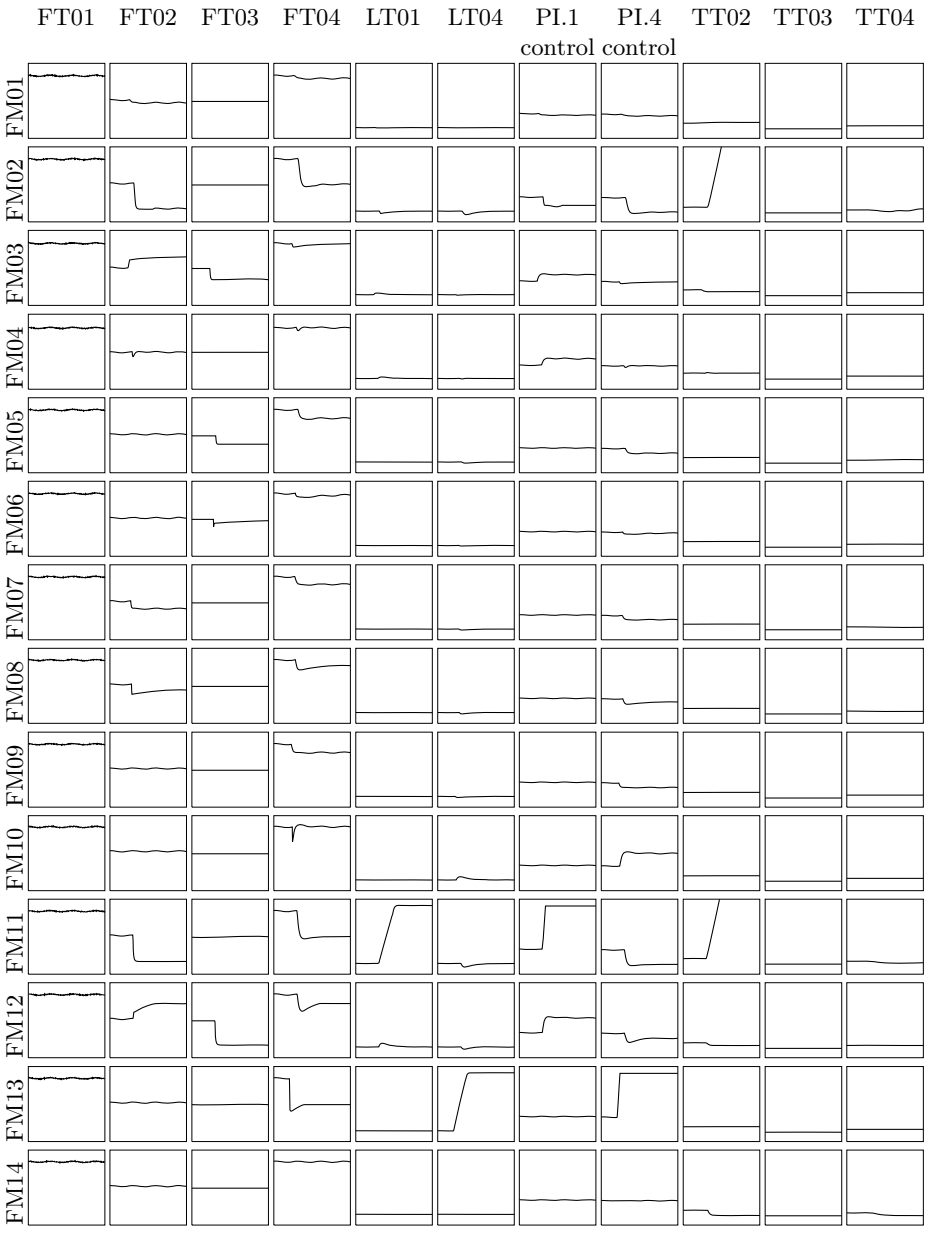


Fig. 2. Examples of classes of faults. Each row shows a case of a fault mode. Each column shows one of the variables.

As the plant we are dealing with has eleven different observed variables, we use a window for each variable. Then, the length of each window will be increased until we reach the maximum time allowed to identify a fault.

We have conducted several experiments using stratified cross-validation to estimate the accuracy of the classification method. We have partitioned the experimental data-set into ten equally sized subsets. Each subset was tested with ten different lengths (from ten percent to hundred percent of the series length).

Tables 2, 3, and 4 show the classification success achieved using one, three, and five nearest neighbors for the reusing task. Increasing the number of neighbors, the accuracy systematically decreases. An explanation for such behavior is that we have twenty examples of each class. This is a small number compared with the number of classes (fourteen), and the dimensionality of the data.

Table 2. Classification success achieved using ten different window lengths, DTW (lineal, quadratic, and valley metrics), Euclidean and Manhattan distances for the retrieving task, and one neighbor for the reusing task

	Window length									
Distance	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
Lineal	32.9	33.2	51.4	84.3	87.1	88.6	88.9	89.2	89.2	89.2
Quadratic	34.6	34.3	56.1	87.9	91.1	90.7	90.4	90.7	91.1	91.4
Valley	34.6	34.3	56.1	87.9	91.1	90.7	90.4	90.7	91.1	91.4
Euclidean	33.6	34.3	52.1	83.6	86.1	87.1	87.5	88.6	88.6	88.9
Manhattan	34.3	33.6	49.3	81.4	85.4	87.5	87.5	87.9	88.6	88.2

Table 3. Classification success achieved using ten different window lengths, DTW (lineal, quadratic, and valley metrics), Euclidean and Manhattan distances for the retrieving task, and three neighbors for the reusing task

	Window length									
Distance	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
Lineal	31.1	30.4	48.2	82.1	83.6	85.4	84.3	85	86.1	86.4
Quadratic	31.8	30.7	57.9	84.3	87.1	84.6	86.1	87.1	87.1	88.6
Valley	31.8	30.7	57.9	84.3	87.1	84.6	86.1	87.1	87.1	88.6
Euclidean	32.9	31.4	51.4	80	84.3	84.3	85	84.6	85	86.1
Manhattan	32.9	31.4	44.3	80.7	83.6	85.4	85.4	84.6	85	85.4

Table 4. Classification success achieved using ten different window lengths, DTW (lineal, quadratic, and valley metrics), Euclidean and Manhattan distances for the retrieving task, and five neighbors for the reusing task

	Window length									
Distance	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
Lineal	21.4	20.7	46.4	79.3	81.4	83.6	83.2	83.9	84.3	85.4
Quadratic	23.2	24.3	53.2	83.2	83.6	84.6	84.3	84.3	85.0	85.0
Valley	23.2	24.3	53.2	83.2	83.6	84.6	84.3	84.3	85.0	85.0
Euclidean	18.9	20	48.6	78.9	82.1	83.6	84.6	85	85.4	85.7
Manhattan	19.6	19.6	42.1	76.4	81.4	83.9	83.2	82.9	83.9	84.6

As expected, the results obtained using DTW are better than those obtained using the Euclidean or Manhattan distances. DTW is a much more robust dissimilarity measure for time series, allowing similar shapes to match even if they are out of phase in the time axis [9]. Euclidean distance is much more affected for small changes in the time axis.

Regarding the classification success, the most important increase occurs on passing through thirty percent to fifty percent of the series length. As we stated before, we randomly generate the fault magnitude in the interval $[180, 300]$. The simulation lasted 900 seconds. Therefore, the interval $[180, 300]$ corresponds with $[20\%, 33\%]$ of the series length. Fault detection using Consistency-Based Diagnosis, as described in [15,16], calls the CBR system after fault detection and localization. In our Consistency-based Diagnosis system, instantaneous fault detection is not required, due to the slow dynamics. Fault detection tests are performed every 45 seconds; hence, there will be a delay in calling the CBR system. So, we consider that 87.9% of classification success for 40% of the whole data series is a rather promising result.

5 Conclusions and Future Work

This article describes an exploratory work of a CBR system for early fault diagnosis in continuous processes. We have considered a laboratory plant with eleven observed variables and fourteen different classes of faults. The system uses the K-Nearest Neighbors algorithm for retrieving cases. It can be used with several metrics, such as Manhattan distance, Euclidean distance and Dynamic Time Warping. The system has been validated using a dataset obtained from simulations of fault classes in a laboratory plant. The CBR system stores some cases corresponding with faults that could arise in the plant. The accuracy of the classifiers using the different metrics with ten different series length (from ten percent to hundred percent of the series length) has been compared.

The proposed method seems to be feasible based on the experimental results so far. In fact, the system is able to provide an accurate fault classification with a small percentage of the series length.

The developed CBR system is able to work with time series, which is a problem not covered in most CBR systems. Comparing with other approaches, for instance, Colomer et al. [7] that proposed a CBR system for situation assessment of dynamic systems based on feature extraction, our system exhibits several advantages. It is able to perform early fault classification against "post-mortem" classification, which is the only one supported in [7]. Additionally, we do not need a previous preprocessing task, and our approach provides a better success classification rate.

We have also compared the result obtained using case-based reasoning with other approaches: decision trees and Support Vector Machines [4], or Boosting with literals [14], obtaining better result for the earliest classification times.

As a future work, we plan to increase the size of the case base. Our initial guess is that a bigger size will improve the result on the K-Nearest Neighbors algorithms for more than one neighbor.

Acknowledgments: This work has been partially funded by Spanish Ministry of Education and Culture, through grant DPI2005-08498, and Junta Castilla y León VA088A05.

References

1. A. Aamodt and E. Plaza. Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. *AI Communications. IOS Press, Vol. 7: 1,*, pages 39–59, 1994.
2. C. Alonso, J.J. Rodriguez, and B. Pulido. Enhancing consistency-based diagnosis with machine-learning techniques. In *14th International Workshop on Principles of Diagnosis, DX03*, Washington, D.C. USA, 2003.
3. D. M. Brann, D. A. Thurman, and C. M. Mitchell. Case-Based Reasoning as a Methodology for Accumulating Human Expertise for Discrete System Control. In *Proceedings of the IEEE Int. Conf. on SMC, Vancouver, B. C., Canada*, pages 4219–4223, 1995.
4. A. Bregón, B. Pulido, M.A. Simón, I. Moro, O. Prieto, J.J. Rodríguez, and C. Alonso. Focusing fault localization in model-based diagnosis with case-based reasoning. In *17th International Workshop on Principles of Diagnosis, DX06*, Peñaranda de Duero, Spain, 2006.
5. A. Bregón, M. A. Simón, J. J. Rodríguez, C. Alonso, B. Pulido, and I. Moro. Un sistema de razonamiento basado en casos para la clasificación de fallos en sistemas dinámicos. In *Primer Congreso Español de Informática, Granada, España*, 2005.
6. J. Britanik and M. Marefat. Case-Based Manufacturing Process Planning with integrated support for knowledge sharing. In *IEEE Int. Symp. on Assembly Task Planning*, pages 107–112, 1995.
7. J. Colomer, J. Melendez, and F. Gamero. A qualitative case-based approach for situation assessment in dynamic systems. application in a two tank system. In *Proceedings of the IFAC-Safeprocess 2003*, Washington, USA, 2003.
8. J. Colomer, J. Melendez, and F. I. Gamero. Qualitative representation of process trends for situation assessment based on cases. In *15th Triennial World Congress of the International Federation of Automatic Control, Barcelona, Spain*, 2002.
9. E. Keogh and C. A. Ratanamahatana. Exact indexing of dynamic time warping. *Knowledge and Information Systems*, 7(3):358–386, 2005.
10. J. Kolodner. *Case-Based Reasoning*. Morgan Kaufmann Publishers, 1993.
11. D. B. Leake. CBR in Context: The present and Future. *Case-Based Reasoning: Experiences, Lessons, and Future Directions. Menlo Park: AAAI Press.*, 1996.
12. M. Lenz, M. Manago, and E. Auriol. Diagnosis and decision support. In *Case-Based Reasoning Technology*, pages 51–90, 1998.
13. C. Price. *Computer-based diagnostic systems*. Springer Verlag, New York, 1999.
14. O. Prieto and A. Bregón. A comparative of two machine-learning techniques to focus the diagnosis task(to be published, draft available upon request). In *Third European Starting AI Researcher Symposium (STAIRS'06), Riva del Garda, Italy*, 2006.

15. B. Pulido, C. Alonso, and F. Acebes. Lessons learned from diagnosing dynamic systems using possible conflicts and quantitative models. In *Engineering of Intelligent Systems. Fourteenth International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems (IEA/AIE-2001)*, volume 2070 of *Lecture Notes in Artificial Intelligence*, pages 135–144, Budapest, Hungary, 2001.
16. B. Pulido and C. Alonso González. Possible conflicts: a compilation technique for consistency-based diagnosis. *"IEEE Trans. on Systems, Man, and Cybernetics. Part B: Cybernetics"*, 34(5):2192–2206, 2004.
17. B. Pulido, J.J. Rodriguez Diez, C. Alonso González, O. Prieto, E. Gelso, and F. Acebes. Diagnosis of continuous dynamic systems: integrating consistency-based diagnosis with machine-learning techniques. In *XVI IFAC World Congress, 2005*, Prague, Zcheck Republic, 2005.
18. I. Watson. *Applying Case-Based Reasoning: Techniques for Enterprise Systems.* Morgan Kaufmann Publishers, 1997.

Face Description for Perceptual User Interfaces

M. Castrillón-Santana, J. Lorenzo-Navarro, D. Hernández-Sosa, and J. Isern-González

IUSIANI
Edificio Central del Parque Científico-Tecnológico
Campus Universitario de Tafira
Universidad de Las Palmas de Gran Canaria
35017 Las Palmas - Spain
mcastrillon@mcastrillon.ulpgc.es

Abstract. We investigate mechanisms which can endow the computer with the ability of describing a human face by means of computer vision techniques. This is a necessary requirement in order to develop HCI approaches which make the user feel himself/herself perceived. This paper describes our experiences considering gender, race and the presence of moustache and glasses. This is accomplished comparing, on a set of 6000 facial images, two different face representation approaches: Principal Components Analysis (PCA) and Gabor filters. The results achieved using a Support Vector Machine (SVM) based classifier are promising and particularly better for the second representation approach.

1 Introduction

Nowadays human faces are everywhere. Not only we are exposed to facial patterns due to the fact of living in community, but also faces are present on magazine covers, commercials, news, ads, etc. Faces provide a channel which fills a main part of the non verbal communication held during human encounters [1]. They convey both dynamic information and signals of great interest for social interaction such as gender, age and more. The interpretation of those signals is a basic ability to be included in any Vision Based Interface [27] which makes use of Computer Vision technology to perceive the user in a Human Computer Interaction (HCI) context.

Many facial analysis papers have particularly focused on the face recognition and verification problems. A well known corpus used to evaluate recognition techniques is the FERET database [22] and more recently the Face Recognition Vendor Test. Verification approaches have their own framework, the BANCA protocol [2]. However, other facial descriptors which are particularly useful to describe unknown individuals, or to realize changes in human appearance during social interaction, have not excited the interest of the researchers similarly. Certainly, gender classification and facial expression recognition are exceptions [19,21]; but other descriptors such as race, glasses, moustaches, beards, hair color, hair style, eyes color, etc., have not been widely considered.

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 221–230, 2006.

Recent developments suggest that these descriptors can be of interest for automatic face processing. Indeed, local context is taking more importance in the literature for detection and recognition [26]. Some authors have evidenced that the local context is used differently by individuals, e.g. people born and living in Europe would pay more attention to hair and its color while people born and living in Japan would not consider the hair as an identification cue [6,24].

In this paper our main effort is to provide results in the direction of face description by means of additional semantic labels. These labels or descriptors can be used during HCI in order to endow the computer with abilities that can make the user feel himself/herself perceived. For this aim, we compare two well known state of the art approaches for face representation: 1) Principal Components Analysis (PCA), and 2) Gabor filters. Both representations are used by a Support Vector Machine (SVM) based classifier to provide an automatic suggestion.

An introduction to the techniques employed for face representation is given in Section 2. Section 3 presents and discusses the experimental results achieved. Some conclusions are summarized in Section 4.

2 Facial Description

In this work, we have paid attention only to inner facial features, trying to cover a small subset of the semantic facial descriptors which can be extracted from a single face image: gender, race, and the presence of moustache and glasses.

Different recent works have tackled the problem of gender recognition. A recent approach based on perfectly aligned images outperforms humans in low resolution images [19]. In [18] a Gabor wavelet representation on selected points is used with good results in gender and race classification. In relation with the others descriptors, there are different references [15,29] which try to detect the presence of glasses in a face, but we have none tackling the presence of moustache.

2.1 Face Representation

Principal Components Analysis (PCA). PCA decomposition is a well known technique used to reduce data redundancy. This representation schema chooses the dimension reduction that maximizes the scatter of the projected samples. PCA has been used extensively for face representation since the work described in [17], due to the fact that it provides a reduced representation without a significant lost of information. As seen in Figure 1, once an image, I, is projected, different coefficients represent the image in this space of reduced dimensionality, $I_{PCA} = \{v_1, v_2, v_3, ..., v_n\}$. The original image can be recovered by means of a linear combination, defined by these coefficients, of the different eigenvectors plus the average image.

Gabor Filters. The linear receptive field (RF) for simple cell responses in the primary visual cortex (V1) can be modeled by two-dimensional Gabor filters [16].

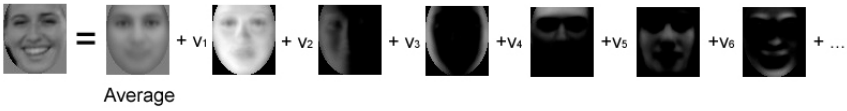


Fig. 1. PCA decomposition of an image into the average image and the linear combination of different eigenvectors, known as eigenfaces

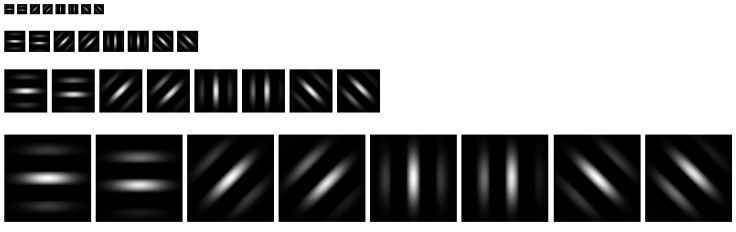


Fig. 2. The bank of filters applied sized 11×11, 23×23, 47×47, 95×95

Most of these cells are combined in pairs, one cell of each pair has even symmetry and the other one has odd symmetry [23]. These considerations allow for a definition of a biologically motivated filter [10]:

$$p_k(x) = \frac{k^2}{\sigma^2} \exp(-\frac{k^2}{2\sigma^2}x^2)(\exp(ikx) - \exp(-\frac{\sigma^2}{2})) \quad (1)$$

where k defines the frequency, orientation and location of the filter. These filters have been used in recent years as independent components to represent natural images [20].

The convolution of an image with these filters provides –for each pixel– a vector whose dimension depends on the number of orientations and scales used. In this paper we used 4 different orientations and 4 different scales, as shown in Figure 2. Thus, for each pixel 32 values are obtained (4 scales, 4 orientations and 2 symmetries). Therefore, the convolution yields a representation of higher dimensionality than the original image.

Because of the resulting high dimensionality, different authors have considered ways to reduce the number of points, orientations, or scales used. In [7] a genetic algorithm accomplishes this reduction for texture classification. For faces, a weighted grid is employed for recognition configuring a representation that is more robust to pose changes [12]. For facial expression recognition the selection of the Gabor filters giving all the possibilities in terms of position, orientation and scale, is performed by an Adaboost approach followed by a SVM-based classifier. This approach yields a system that can provide real-time classification performance [3]. A similar approach for face recognition is described in [30] but in this case, it uses the intra-face and extra-face difference spaces for classification instead of a SVM-based classifier.

In our work we have considered applying the Gabor filters to a smaller number of image points thereby decreasing redundancy and providing faster classification. In order to have some preliminary results, our selection approach is quite simple in contrast to the Adaboost approach used in [3] and [30]. This simple approach performs different scans as described in Figure 3. On each scan the previously stored Gabor points (if any) are combined with each pixel. The location of the pixel that provides the best performance is then added to the list of points only if it improves the previous best rate.

```
change ← true
max_performance ← 0
Resets Gabor points list
while change do
   change ← false
   for each pixel do
      Compute Gabor representation using the Gabor points list and the current pixel location
      Get new_performance for the set
      if new_performance > max_performance then
         change ← true
         max_performance ← new_performance
         Stores current pixel location in a temporary location
      end if
   end for
   if change then
      Adds best pixel location to the Gabor list
   end if
end while
```

Fig. 3. Gabor points selection algorithm

3 Experiments

3.1 Datasets and Libraries

The dataset contains 6000 face images taken randomly from internet and selected samples from facial databases such as the BIOID [11]. They have been annotated by hand to get their eye positions and labelled according to the different semantic descriptors considered: female/male, clear/dark, glasses/no glasses, moustache/no moustache. These images have been normalized according to eye positions obtaining 59×65 pixels images. Table 1 summarizes the composition of the test and training sets used for the experimental setup.

Every face analyzed is transformed to both face representation spaces described above. For each representation and descriptor a classifier is computed based on the widely used and powerful Support Vector Machine (SVM) approach [28].

Different libraries have been used for these experiments. The OpenCV [14] library provides tools for PCA computation and projection. The Gabor filters have been computed adapting for OpenCV the David Bolme implementation [5]. Finally the SVM classifier was implemented making use of the available LIBSVM [9].

Table 1. Training and test sets containing 59×65 pixels images. We have tried to build balanced (in number) training sets. For some descriptors one class has not so many samples, for that reason the training set is reduced and therefore the test set has much more samples of the typical class in the dataset: clear skin, no glasses, no moustache.

	Training set		Test set	
Descriptor	Female	Male	Female	Male
Gender	1223	1523	835	2246
	Training set		Test set	
Descriptor	Clear	Dark	Clear	Dark
Race	574	316	4811	306
	Training set		Test set	
Descriptor	No	Yes	No	Yes
Glasses presence	912	692	4042	356
Moustache presence	710	480	4389	426

3.2 PCA + SVM

The PCA space was computed using 4000 samples of the face dataset requiring 12 hours in a PIV 2.2 Ghz. The paper described in [8] analyzed the performance of different classifiers based on a PCA+SVM approach modifying the number of eigenfaces used for classification. The authors concluded that 70 coefficients provide a good trade-off between correct recognition rate and training processing time, see Figure 4.

That said, it can be considered the fact, as already referred by different authors [4], that some eigenfeatures selected have no interest for the problem analyzed. Thus, the first eigenfaces contain generally information related with illumination that is not useful here, or for example some eigenfeatures contain information that may not be discriminant for the glasses presence problem. Other authors have considered a more precise selection of them, for example in [25] the authors do not just take the first n eigenfeatures but select them by means of a genetic algorithm. Instead of this, we have considered the use of a Gabor filters based representation whose results are presented in the next section.

3.3 Gabor Filters + SVM

The Gabor location selection approach described above, see Figure 3, was applied to each training set in Table 1 to choose the best configuration based on the Gabor banks. For any of the problems after one scan, the overall hit resulted in at least 99% for the training set. Due to this fact, in the experiments presented here we have selected new points to add based on the performance of the test set (however no image from the test set was used to train the classifier) in order to achieve a longer performance evolution. The final performance and locations achieved for the different problems are depicted in Figures 5 and 6 respectively.

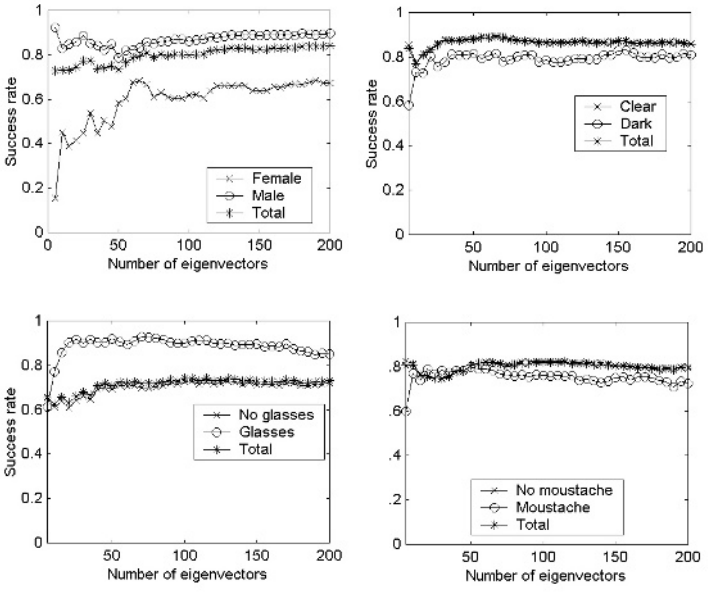


Fig. 4. PCA+SVM performance. Top left) gender, top right) race, bottom left) glasses presence, bottom right) moustache presence.

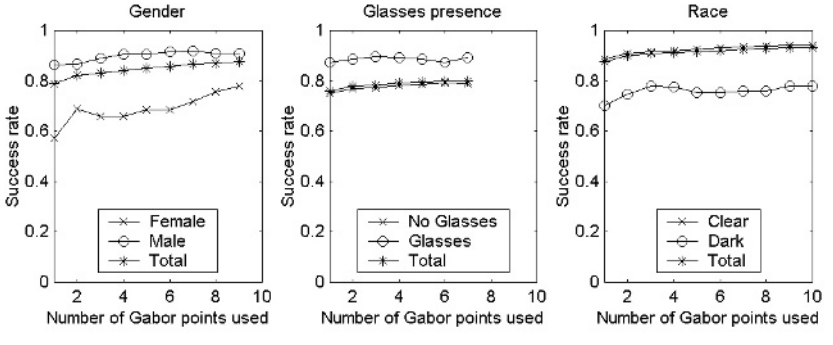


Fig. 5. Gabor+SVM. Iterative improvement for the test set for gender, glasses presence and race.

As suggested in Figure 6, the selected points are located in image regions related to the feature being analyzed. In the case of gender, it seems to be a concentration close to the mouth and eyes, which fits with psychophysical results achieved in [13]. The points to check the moustache and glasses presence are selected close to the possible location of those elements in the face (even when all the image pixels were considered). However, we have not yet a clear explanation for the Gabor points selected for race classification.

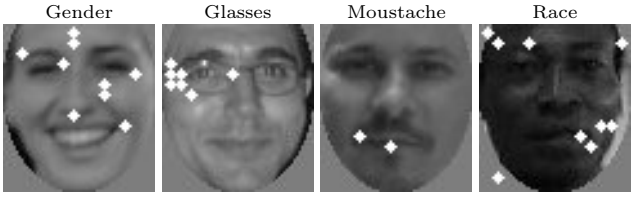


Fig. 6. Gabor points location selected by the approach for the different descriptors

3.4 Discussion

Table 2 presents the best results achieved for each problem using both approaches, i.e. PCA+SVM and Gabor-based+SVM. The overall performance is clearly better using the second approach without showing a significant increase in computational cost. This result is due to the fact that only some selected points are used to compute the Gabor features.

However, the first approach makes possible a shorter training stage as the new classifier setup can be built very fast as only a single SVM classifier training is necessary (if the PCA space is considered fixed). On the other hand, the compilation of the Gabor points list which provides the best performance for a given set, requires an undefined number of scans on the image and the computation of multiple SVM classifiers during the selection process. This fact is a disadvantage of this approach if we consider the possibility of an interactive system which were able to learn incrementally, and therefore would need to retrain online.

Table 2. Summary comparing the best classifier for each approach. The processing time required for classification is indicated in milliseconds.

	PCA + SVM			Gabor filters		
Descriptor	Recog. rate	N. eigenvalues	Proc. time	Recog. rate	N. filters	Proc. time
Gender	82.9%	160	5	87.6%	9	7
Race	89.1%	65	1	93.2%	10	5
Glasses presence	72%	70	1	79.8%	7	2
Moustache presence	81.8%	65	1	85.8%	2	1

4 Conclusions and Future Work

A dataset of still face images has been used to analyze the possibility of automatic suggestion of semantic descriptors given a human face image. In some cases these descriptors have not been considered previously by facial analysis literature. Two face representation approaches have been compared, getting better results for

the one which is based on the biologically motivated Gabor filters. The resulting classifiers perform reliably in both cases for real time operation.

Future work should also extend the facial descriptors domain and take into account the possibility of providing a weighted output. For example the race descriptor should not have a binary output. Indeed a degree could be provided and/or some other classes could be added such as Asian, Indian, Hispanic, etc. Additionally a comparison must be performed with other approaches for Gabor banks location selection such as Adaboost and genetic algorithms. This can also be applied to the different eigenfeatures in the first approach, or to the number of Gabor filters, orientations, and scales in the second. A combination of both approaches can also be analyzed.

The face dataset must also be increased due to the fact that currently it does not contain a large number of samples for some of the descriptor classes. We think that a reason to this is that in our main source, i.e. internet, it is easier to gather images corresponding to young, caucassian and good looking (!) people. Therefore, the collection of new samples for non trendy features requires a longer search. Bigger databases would likely be needed to provide better performance, particularly for those descriptors which present a clear border among the different classes, e.g. glasses presence.

Acknowledgments

Work partially funded by research projects Univ. of Las Palmas de Gran Canaria UNI2003/06, UNI2004/10 and UNI2004/25, Canary Islands Autonomous Government PI2003/160 and PI2003/165 and the Spanish Ministry of Education and Science and FEDER funds (TIN2004-07087).

References

1. M. Argyle. *Bodily communication*. Methuen, 2nd edition, 1988.
2. E Bailly-Bailliere, S Bengio, F Bimbot, M Hamouz, J Kittler, J Mariethoz, J Matas, K Messer, V Popovici, F Poree, B Ruiz, and J-P Thiran. The banca database and evaluation protocol. In J Kittler and M Nixon, editors, *Proc. Audio- and Video-Based Biometric Person Authentication*, pages 625–638, Berlin, June 2003. Springer.
3. M.S. Bartlett, Gwen Littlewort, Ian Fasel, and Javier R. Movellan. Real time face detection and facial expression recognition: Development and applications to human computer interaction. In *Computer Vision and Pattern Recognition*, 2003.
4. P.N. Belhumeur, J.P. Hespanha, and D.J. Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *IEEE Trans. on PAMI*, 19(7):711–720, 1997.
5. David Bolme. Elastic bunch graph matching. Master's thesis, Colorado State University, Computer Science Department, June 2003 2003.
6. Vicki Bruce and Andy Young. *The eye of the beholder*. Oxford University Press, 1998.

7. N. W. Campbell and B. T. Thomas. Automatic selection of gabor filters for pixel classification. In *Sixth International Conference on Image Processing and its Applications*, pages 761–765, July 1997.
8. M. Castrillón Santana, J. Lorenzo Navarro, D. Hernández Sosa, and Y. Rodríguez-Domínguez. An analysis of facial description in static images and video streams. In *2nd Iberian Conference on Pattern Recognition and Image Analysis*, Estoril, Portugal, June 2005.
9. Chi-Chung Chang and Chih-Jen Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.
10. J.G. Daugman. Complete discrete 2-d gabor transforms by neural networks for image analysis and compression. *IEEE Trans. on Acoustics, Speech, and Signal Processing*, 36(7), July 1988.
11. Robert W. Frischholz and Ulrich Dieckmann. Bioid: A multimodal biometric identification system. *IEEE Computer*, 33(2), February 2000.
12. B. Gokberk, L. Akarun, and E. Alpaydýn. Feature selection for pose invariant face recognition. In *International Conference on Pattern Recognition*, Barcelona (Spain), 2002.
13. F. Gosselin and P. G. Schyns. Bubbles: a technique to reveal the use of information in recognition tasks. *Vision Research*, pages 2261–2271, 2001.
14. Intel. Intel Open Source Computer Vision Library, b4.0. www.intel.com/research/mrl/research/opencv, August 2004.
15. Zhong Jing and Robert Mariani. Glasses detection and extraction by deformable contour. In *International Conference on Pattern Recognition*, 2000.
16. J.P. Jones and L.A. Palmer. An evaluation of the two-dimensional gabor filter model of simple receptive fields in cat striate cortex. *Journal of Neurophisiology*, 58(6):1233–1258, 1987.
17. Y. Kirby and L. Sirovich. Application of the karhunen-loève procedure for the characterization of human faces. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 12(1), July 1990.
18. Michael J. Lyons, Julien Budyneck, and Shigery Akamatsu. Automatic classification of single facial images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(12):1357–1362, December 1999.
19. Baback Moghaddam and Ming-Hsuan Yang. Learning gender with support faces. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(5):707–711, 2002.
20. Bruno A. Olshausen and David J. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381:607–609, 1996.
21. Maja Pantic and Leon J.M. Rothkrantz. Automatic analysis of facial expressions: The state of the art. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(12):1424–1445, December 2000.
22. P. Jonathon Phillips, Hyeonjoon Moon, Syed A. Rizvi, and Patrick J. Rauss. The feret evaluation methodology for face recognition algorithms. TR 6264, NISTIR, January 1999.
23. Daniel A. Pollen and Steven F. Ronner. Phase relationship between adjacent simple cells in the visual cortex. *Science*, 212:1409–1411, June 1981.
24. Pawan Sinha and T. Poggio Torralba. I think i know that face... *Nature*, 384(6608):384–404, 1996.
25. Zehang Sun, George Bebis, Xiaojing Yuan, and Sushil J. Louis. Genetic feature subset selection for gender classification: A comparison study. In *Sixth IEEE Workshop on Applications of Computer Vision*, December 2002.

26. Antonio Torralba. Contextual modulation of target saliency. *Advances in Neural Information Processing Systems*, 2001.
27. M. Turk. Computer vision in the interface. *Communications of the ACM*, 47(1):61–67, January 2004.
28. V. Vapnik. *The nature of statistical learning theory*. Springer, New York, 1995.
29. Bo Wu, Haizhou Ai, and Ran Liu. Glasses detection by boosting simple wavelet features. In *17th Int. Conf. on Pattern Recognition, Cambridge, UK*, pages 292–295, August 2004.
30. Peng Yang, Shiguang Shan, Wen Gao, Stan Z. Li, and Dong Zhang. Face recognition using ada-boosted gabor features. In *Proc. of the 6th International Conference on Automatic Face and Gesture Recognition*, 2004.

Genetic Algorithms Hybridized with Greedy Algorithms and Local Search over the Spaces of Active and Semi-active Schedules*

Miguel A. González, María Sierra, Camino R. Vela, and Ramiro Varela

University of Oviedo. Department of Computing. Artificial Intelligence Center.
Campus of Viesques, 33271 Gijón, Spain
http://www.aic.uniovi.es/Tc

Abstract. The Job Shop Scheduling is a paradigm of Constraint Satisfaction Problems that has interested to researchers over the last years. In this work we propose a Genetic Algorithm hybridized with a local search method that searches over the space of semi-active schedules and a heuristic seeding method that generates active schedules stochastically. We report results from an experimental study over a small set of selected problem instances of common use, and also over a set of big problem instances that clarify the influence of each method in the Genetic Algorithm performance.

1 Introduction

The Job Shop Scheduling (JSS) is a Constraint Satisfaction Problem (CSP) that has interested to many researchers over the last years. Consequently we can found in the literature a great number of approaches based on different metaheuristics, constraint satisfaction techniques or operational research algorithms [6]. In this paper we apply three complementary techniques to the JSS problem: a Genetic Algorithm (GA) with decoding in the set of parameterized active schedules, a local search method to refine each chromosome that searches over the space of semi-active schedules, and also a randomized greedy algorithm guided by an admissible heuristic that generates active schedules to seed the initial population of the GA. We report results from an experimental study over two sets of problem instances of medium and large size. These results show that the local search is always very efficient and that the heuristic seeding is only efficient in the short term for medium size instances and that it is comparable to local search for large instances. The rest of the paper is organized as follows. In section 2 we formulate the JSS problem. In section 3 we describe the search spaces of active and semi-active schedules. Then the proposed approaches are described in the next three sections: the GA in section 4, the local search in section 5 and the seeding method in section 6. Section 7 reports the experimental study. And finally in section 8 we summarize the main conclusions.

* This work has been supported by project FEDER-MCYT TIC2003-04153 and by FICYT under grant BP04-021.

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 231–240, 2006.

2 Problem Formulation

The JSS problem requires scheduling a set of N jobs $\{J_1, \ldots, J_N\}$ on a set of M physical resources or machines $\{R_1, \ldots, R_M\}$. Each job J_i consists of a set of tasks or operations$\{\theta_{i1}, \ldots, \theta_{iM}\}$ to be sequentially scheduled. Each task θ_{il} has a single resource requirement, a fixed duration $du\theta_{il}$ and a start time $st\theta_{il}$ whose value should be determined. The problem has three constraints: precedence constraints, capacity constraints and non-interruption constraints. Precedence constraints defined by the sequential routings of the tasks within a job translate into linear inequalities of the type: $st\theta_{il} + du\theta_{il} \leq st\theta_{i(l+1)}$ (i.e. θ_{il} before $\theta_{i(l+1)}$). Capacity constraints that restrict the use of each resource to only one task at a time translate into disjunctive constraints of the form: $st\theta_{il}+du\theta_{il} \leq st\theta_{jk} \vee st\theta_{jk} + du\theta_{jk} \leq st\theta_{il}$. Non-interruption constraints mean that once an operation has started it cannot be interrupted for its whole processing time. The objective is to come up with a feasible schedule such that the completion time, i.e. the makespan, is minimized.

A problem instance is usually represented by means of the constraint graph, see Figure 1, where each node represents an operation and its resource requirement, with the exceptions of the dummy nodes *start* and *end*; bold arcs represent

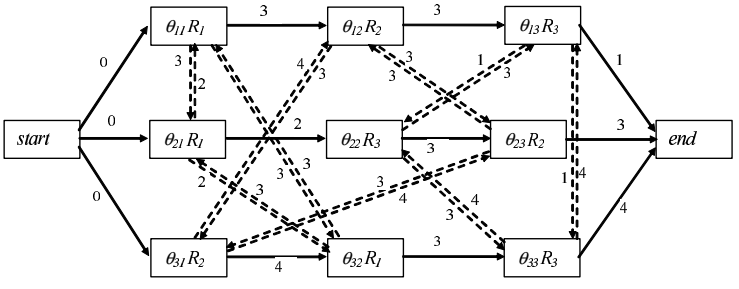


Fig. 1. A problem instance with 3 jobs and 3 machines

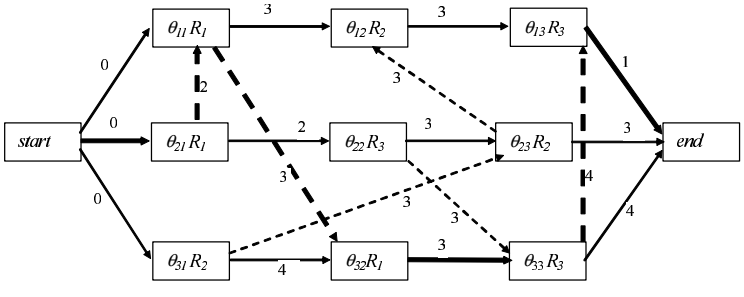


Fig. 2. A feasible schedule to a problem with 3 jobs and 3 machines. The bold face arcs show a critical path whose length, i.e. the makespan, is 12.

precedence constraints and dotted arcs represent capacity constraints. Every arc is labeled by the duration of the operation at the outcoming node. This way a problem solution might be represented by an acyclic subgraph of the constraint graph showing job routings on the machines. Figure 2 shows a feasible solution to the problem instance of Figure 1. The makespan is the cost of the longest or critical path between the dummy nodes *start* and *end*. A maximal subset of consecutive operations on this path requiring the same machine is a critical block.

3 Semi-active, Active and Non-delay Schedules

For scheduling problems there are three search spaces that are of interest: the spaces of semi-active, active and non-delay schedules.

Definition 1. *A schedule is* ***semi-active*** *if for an operation to start earlier, the relative ordering of at least two operations must be swapped.*

Definition 2. *A schedule is* ***active*** *if for an operation to start earlier, at least another one must be delayed.*

Definition 3. *A schedule is* ***non-delay*** *if it is never the case that a machine is idle while any operation can start to run on that machine.*

Active schedules are a subset of semi-active schedules, and non-delay schedules are a subset of active ones. The space of active schedules is complete, however the space of non-delay schedules is not. For this reason active schedules seem to be the most interesting choice, even though in some cases semi-active and non-delay schedules are quite interesting too, as we will see later. To build up active schedules it is of common use the well-known $G\&T$ algorithm proposed by Giffler and Thomson in [5]. This algorithm can be improved with a filtering mechanism to limit the amount of time that a machine is idle while a requiring operation is ready to run. This mechanism is controlled by means of a parameter $\delta \in [0, 1]$, in this case the algorithm is termed as hybrid $G\&T$ and searches over the space of the so called parameterized active schedules. The hybrid $G\&T$ algorithm is showed in Algorithm 1, its complexity is $O(N * M * max(N, K))$, K being the complexity of the selection operation at step 8. When $\delta = 1$ the algorithm searches over the whole set of active schedules. As long as parameter δ decreases the search space is constrained to a subset of active schedules so that it might not be complete, and in the extreme $\delta = 0$ the search is restricted to the space of non-delay schedules.

4 A Genetic Algorithm for the JSS Problem

We have used a rather conventional GA with generational replacement. Every generation is constructed from the previous one by first grouping all the chromosomes into pairs in the selection phase. Then crossover is applied to each pair of

Algorithm 1. Hybrid G&T

1. Let $A = \{\theta_{j1}, 1 \leq j < N\}$;
while $A \neq \emptyset$ **do**
2. Let $st\theta_i$ be the earliest starting time of θ_i $\forall\theta_i \in A$;
3. Let $\theta_1 \in A$ such that $st\theta_1 + du\theta_1 \leq st\theta + du\theta, \forall\theta \in A$;
4. Let $R = MR(\theta_1)$; $\{MR(\theta)$ *is the machine required by operation* $\theta\}$
5. Let $B = \{\theta \in A; MR(\theta) = R, st\theta < st\theta_1 + du\theta_1\}$
6. Let $\theta_2 \in B$ such that $st\theta_2 \leq st\theta, \forall\theta \in B$;
{*the earliest starting time of every operation in B,if it is selected next,is a value of the interval* $[st\theta_2, st\theta_1 + du\theta_1[\}$
7. Reduce the set B such that
$B = \{\theta \in B : st\theta \leq st\theta_2 + \delta((st\theta_1 + du\theta_1) - st\theta_2), \delta \in [0, 1]\}$;
{*now the interval is restricted to* $[st\theta_2, st\theta_2 + \delta((st\theta_1 + du\theta_1) - st\theta_2]\}$
8. Select $\theta^* \in B$ **with a given criteria** and schedule it at time $st\theta^*$;
9. Let $A = A\backslash\{\theta^*\} \cup \{SUC(\theta^*)\}$;
{$SUC(\theta)$ *is the next operation to* θ *in its job if any exists*}
end while

chromosomes accordingly to crossover probability P_c and the resulting offsprings are mutated accordingly to mutation probability P_m. Finally a tournament selection is done among each two parents and their two offsprings to take part of the next generation. To codify chromosomes we have chosen permutations with repetition as proposed by Bierwirth in [1]: a chromosome is a permutation of the set of operations, each one being represented by its job number. This way a job number appears within a chromosome as many times as the number of job operations. For example the chromosome *(2 1 1 3 2 3 1 2 3)* actually represents the permutation of operations *(*θ_{21} θ_{11} θ_{12} θ_{31} θ_{22} θ_{32} θ_{13} θ_{23} θ_{33}*)*. This permutation should be understood as expressing tentative partial schedules for each set of operations requiring the same machine. When two of these partial schedules are not compatibles each one to the other, the decoding algorithm is in charge of repairing them. This codification presents a number of interesting characteristics, e.g. it is easy to evaluate with different algorithms, it allows for efficient genetic operators and, in general, it is better than other permutation based schemas as demonstrated in [9]. Crossover and mutation operators are generalized order crossover and order based mutation also described in [1].

In order to evaluate chromosomes we used the Algorithm 1 so that the selected operation in sentence 8 is the one of the set B that is the leftmost in the chromosome sequence. This way the decoding algorithm has a complexity of $O(N^2 * M^2)$. In doing so, the GA searches over the set of (parameterized) active schedules. A decoding algorithm generating semi-active schedules may be implemented with a complexity of $O(N * M)$. However in this case the average quality of the schedules is worse and the GA reaches finally worse solutions when running during the same amount of time. A prototype implementation designed accordingly to the guidelines above and codified in *C++* language can be downloaded from the web site *http://www.aic.uniovi.es/tc*.

5 Local Search

Conventional *GAs* as the one described in previous section often produce moderate results. However meaningful improvements can be obtained by means of hybridization with other methods. One of these techniques is local search [7] [10], in this case the *GA* is called a Memetic Algorithm. Roughly speaking local search is usually implemented by defining a neighborhood of each point in the search space as the set of chromosomes reachable by a given transformation rule. Then a chromosome is replaced in the population by the best neighbor accordingly to the acceptation criterion. The local search completes either after a number of iterations or when no neighbor satisfies the acceptation criterion.

In this paper we consider the local search strategy termed N_3 by D. Mattfeld in [7]. This strategy relies on the concepts of critical path and critical block. For example in the problem instance of Figure 1 there is a critical path defined by the operations sequence $(\theta_{21}\ \theta_{11}\ \theta_{32}\ \theta_{33}\ \theta_{13})$ that contains two critical blocks given by the sub-sequences $(\theta_{21}\ \theta_{11}\ \theta_{32})$ and $(\theta_{33}\ \theta_{13})$. N_3 considers every critical block of a critical path and made a number of moves on the operations of each block. After a move inside a block the feasibility must be tested. Since an exact procedure is computationally prohibitive, the feasibility is estimated by an approximate algorithm proposed by Dell' Amico and Trubian in [4]. This estimation ensures feasibility at the expense of omitting a few feasible solutions. In [7] the transformation rule of N_3 is defined as follows.

Definition 4 (N_3). *Let v and w be successive operations on a critical path, and let PMv and SMw the machine predecessor of v and the machine successor of w respectively. If neither PMv nor SMw belong to the same block as v and w, then the only permutation is (w, v), otherwise every permutation of PMv, v, w and v, w, SMw is considered as neighboring if v and w are reversed also.*

The acceptation criterion is based on a makespan estimation which is done in constant time, as it is also described in [7], instead of calculating the exact makespan of each neighbor. The estimation provides a lower bound of the makespan. The selected neighbor is the one with the lowest makespan estimation whenever this value is lower than the makespan of the current chromosome. The solution graph obtained with N_3 always represents a feasible schedule, but this schedule might be semi-active. Only the last accepted neighbor is translated into a new chromosome (by doing a simply topological sorting of the solution graph) and inserted in the next generation.

When used, the local search is applied to every chromosome in the initial population and also during evolution to each chromosome generated by crossover and mutation. In any case the local search algorithm is applied just after the hybrid $G\&T$ algorithm, hence starting from an active schedule.

6 Heuristic Seeding

In order to build up the initial chromosomes we have consider two possibilities. Firstly random permutations and then chromosomes generated by means of the

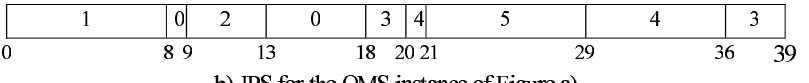
hybrid $G\&T$ algorithm. In the last case, to obtain good chromosomes and an acceptable degree of diversity at the same time we have used a stochastic selection rule in step 8 (see Algorithm 1) based on a heuristic criteria that assigns a selection probability to each possible choice. These schedules are then transformed into chromosomes by topological sorting of the solution graph.

In this work we have used a heuristic criteria based on the so called Jackson's Preemtive Schedule (JPS) described in [2][3]. The JPS is a lower bound of a JSS problem instance obtained by relaxation of the original problem in two steps: first the capacity constraints of every machine but one M' are relaxed to obtain an instance of the One Machine Scheduling (OMS) problem with heads and tails, and then the non-interruption constraint of machine M' is relaxed too. Heads and tails are given in principle by the duration of operations before and after the operation requiring to machine M' in every job respectively, and then adjusted by constraint propagation as operations are scheduled. The JPS is computed as follows. Let us consider a set of jobs J' such that each job θ_i of J' is defined by a head r_i that indicates the earliest starting time of θ_i in M', a processing time $d\theta_i$ on machine M', and a tail q_i establishing a lower bound of the processing time of θ_i after completion on M'. $JPS(J')$ is the best solution of instance J' if J' is relaxed as it is indicated above. The procedure to compute $JPS(J')$ starts by assigning each job θ_i a priority proportional to q_i. Then at any time t from the lowest r_i until every job is processed on machine M', the job with the largest priority among the jobs ready to run on M' is assigned to this machine during the time interval $[t, t+1[$. Figure 3 shows an example of JPS for a problem instance. In [2] J. Carlier and E. Pinson prove that the JPS can be computed with a complexity of $O(n * log_2 n)$, n being the number of jobs in J'.

The JPS yields a lower bound of the makespan of the best solution that can be reached form an intermediate state during the application of the hybrid $G\&T$ algorithm. Hence what we do is to compute the JSP restricted to M' for each state after scheduling one of the operations in set B. Then each operation in B has a selection probability in inverse ratio to the corresponding JPS.

i	0	1	2	3	4	5
r_i	4	0	9	15	20	21
$d\theta_i$	6	8	4	5	8	8
q_i	20	25	30	9	14	16

a) OMS problem instance



b) JPS for the OMS instance of Figure a)

Fig. 3. The Jackson's Preemtive Schedule of a One Machine Sequencing problem with heads and tails. The makespan is 50 given by the completion time of job 4 on M' plus its tail $(36 + 14)$.

7 Experimental Study

For experimental study we have considered two benchmarks. Firstly a set of selected instances of medium size taken from the OR-library that are recognized as hard to solve for a number of researchers, for example D. Mattfeld [7]. We have also considered another set of large instances with sizes 30×30 and 40×40 that were used in [8]. In every case we have considered different combinations of the proposed strategies: heuristic seeding (HS), local search (LS) and parameterized reduction of the search space, i.e. different values of parameter δ. Other GA parameters such as crossover and mutation probabilities have been chosen from our experience. Regarding the number of generations and the population size, we have chosen values so that the running time of different algorithms is similar in order to compare them. In every case we have run the GA 30 times and take the best solution reached, the mean of the best solutions reached in each trial and the running time in seconds.

Tables 1 and 2 summarize the results from the GA over the 12 selected instances with local search and two values of parameter δ. When local search is not used the GA reaches better solutions with $\delta = 0.5$. This happens due to the raw GA is not able to search for any of the good solutions in the whole search space and so it is a better choice restricting the search to a subset of solutions with low average makespan, even though none of these solutions is optimal or near optimal. However when combined with local search, the results are better when $\delta = 1$. In this case the GA is able to guide the search towards near optimal solutions over the whole search space.

We have also conducted similar experiments with semi-active decoding. In this case the results were worse. With the raw GA the mean error in percent was 15.6 with a similar running time. When combined with local search and parameters /100/140/0.7/0.2// the mean error in percent was 3.0; but with double running time due to the local search algorithm has more opportunities of improving when it starts form a semi-active schedule. The rest of the experiments were conducted by decoding with (parameterized) active schedules.

Table 3 shows the results from the best previous versions combined with heuristic seeding. In spite of the heuristic chromosomes are better than the random ones, the results get worse with respect to random seeding, independently of using or not local search. Analogous to the previous experiments, given the size of the search space for these instances it is better to start from random points of the search space than introducing a bias in the initial population with good chromosomes. In the first case the GA is able to evolve by itself towards good solutions, but in the second the well-known phenomena of premature convergence is produced due to a number of initial chromosomes are concentrated in a suboptimal region of the search space.

Finally Table 4 summarizes the results from the set of large problem instances. These are the best results we have obtained with local search and without it for different values of parameter δ. Now in any case the best results are reached with the largest reduction possible ($\delta = 0$) so restricting the search to the space of non-delay schedules. This is due to the size of the search space is extremely high regarding the running time allowed to the GA. Now local search and heuristic

Table 1. Summary of results of the GA for two values of δ. The GA parameters are /Population Size/Number of Generations/Crossover Probability/Mutation Probability/δ/. The GA configuration is /200/340/0.7/0.2/0.5/ for the first 3 instances and /200/300/0.7/0.2/0.5/ for the remaining ones, in order to get similar running time.

Problem Instance		$\delta = 1$			$\delta = 0.5$		
Name(size)	Best Known	Best	Err%	Time(sec)	Best	Err%	Time(sec)
abz7(20×15)	665	706	7.9	25.6	678	3.3	26.8
abz8(20×15)	670	725	10.7	25.6	699	5.8	27.0
abz9(20×15)	686	733	13.6	25.3	708	5.2	26.7
ft10(10×10)	930	951	4.0	5.8	937	2.7	6.0
ft20(20×5)	1165	1184	3.6	7.2	1174	1.7	7.3
la21(15×10)	1046	1077	4.9	10.0	1046	2.9	10.1
la24(15×10)	935	965	5.3	9.7	941	3.5	10.0
la25(15×10)	977	998	4.3	9.8	985	1.8	10.1
la27(20×10)	1235	1285	6.4	15.1	1263	3.2	15.4
la29(20×10)	1153	1228	8.4	15.2	1193	5.0	15.5
la38(15×15)	1196	1265	7.9	14.8	1222	4.3	14.7
la40(15×15)	1222	1258	5.6	14.3	1245	3.4	15.3
Mean Error%			6.9	14.9		3.6	15.4

Table 2. Summary of results with LS and two values of δ. GA configurations are respectively /100/140/0.7/0.2/1.0/ and /100/180/0.7/0.2/0.5/.

Problem Instance		$\delta = 1$			$\delta = 0.5$		
Name(size)	Best Known	Best	Err%	Time(sec)	Best	Err%	Time(sec)
abz7(20×15)	665	676	2.9	25.8	673	2.5	25.3
abz8(20×15)	670	687	4.0	26.8	685	4.0	26.3
abz9(20×15)	686	703	4.7	26.1	702	7.4	25.3
ft10(10×10)	930	930	0.5	4.7	936	1.1	5.8
ft20(20×5)	1165	1165	1.1	6.3	1173	1.1	5.7
la21(15×10)	1046	1055	1.2	9.1	1055	2.1	8.8
la24(15×10)	935	938	1.6	8.4	944	1.4	8.0
la25(15×10)	977	977	0.9	8.9	982	1.3	8.7
la27(20×10)	1235	1253	2.3	15.8	1249	2.2	14.8
la29(20×10)	1153	1187	4.6	14.5	1179	4.2	14.7
la38(15×15)	1196	1215	3.3	15.3	1216	3.7	15.7
la40(15×15)	1222	1229	1.8	15.1	1229	1.5	15.9
Mean Error%			2.4	14.7		2.7	14.2

seeding produce similar results, even though if both are used together the results do not improve with respect to the independent application of each method. Hence for large problem instances heuristic seeding is also an efficient technique, even comparable to local search, due to the GA is only able to visit a very small fraction of the search space so that the bias towards suboptimal solutions produced by heuristic seeding is better that leaving the GA to start from random points with lower average value.

Table 3. Summary of results obtained with the best versions, with LS and without it, when used in combination with heuristic seeding (33% of the total). Average values of initial populations refer to a sample of 100 chromosomes obtained with each method. The GA parameters are /100/140/0.7/0.2/1/ when LS is used, and /200/340/0.7/0.2/0.5/ for the first 3 problems and /200/300/0.7/0.2/0.5/ for the remaining ones, when LS is not used.

Prob.	Best	Mean Init. Pop.		GA with $LS(\delta = 1)$			GA witout $LS(\delta = 0.5)$		
Inst	Known	Rand.	Heur.	Best	Err%	Time(sec)	Best	Err%	Time(sec)
abz7	665	823	777	677	3.1	26.1	681	3.5	26.1
abz8	670	853	797	688	3.8	27.0	692	5.1	25.3
abz9	686	893	852	703	7.2	25.6	705	8.3	25.0
ft10	930	1197	1108	930	0.7	4.5	937	2.9	5.6
ft20	1165	1526	1492	1165	1.3	6.2	1173	1.7	7.4
la21	1046	1326	1254	1055	1.1	8.7	1068	3.7	10.1
la24	935	1193	1093	935	1.7	8.6	955	3.6	10.0
la25	977	1249	1158	977	0.8	8.6	980	1.5	10.1
la27	1235	1555	1505	1254	2.5	16.2	1265	3.2	15.5
la29	1153	1480	1446	1185	4.5	14.9	1193	4.9	15.6
la38	1196	1531	1423	1205	3.2	16.9	1242	5.5	14.9
la40	1222	1512	1421	1233	2.0	14.6	1241	2.7	14.6
Mean Error%					2.7			3.9	

Table 4. Summary of results from the large problem instances. With LS the GA parameters are /100/140/0.7/0.2/0.0/, and without LS are /300/350/0.7/0.2/0.0/.

	GA without LS				GA with LS			
Problem Instances	Rand. Init. Pop.		Heur. IP (33%)		Rand. Init. Pop.		Heur. IP (33%)	
Name(Size) BN	Best	Err%	Best	Err%	Best	Err%	Best	Err%
P1(30×30) 5794	5855	1.9	5813	1.1	5830	1.3	5827	1.2
P2(30×30) 6448	6566	2.8	6513	2.2	6520	1.8	6507	1.9
P3(30×30) 6843	6898	1.9	6877	1.6	6860	1.4	6885	1.4
P4(40×40) 8221	8281	1.4	8254	1.3	8242	1.5	8262	1.7
P5(40×40) 9096	9221	2.2	9167	1.7	9147	1.3	9131	1.5
P6(40×40) 9789	9881	2.6	9893	1.8	9873	1.6	9876	1.7
Mean Error%		2.1		1.6		1.5		1.6

8 Conclusions

In this work we have combined three strategies for solving the JSS problem: a genetic algorithm, a local search procedure and a stochastic greedy algorithm. We have also considered two complete search spaces: semi-active schedules and active ones. In the last case we have used a strategy to constraint the search to a subspace that despite not being complete has a lower average makespan than the whole space. This filtering mechanism is controlled by a parameter $\delta \in [0, 1]$. Then we have reported results from an experimental study over two

sets of problem instances of medium and large sizes. From this study we can give the following conclusions. Decoding in the space of active schedules is always more efficient than decoding in the space of semi-active ones. The local search procedure used, termed N_3 in the literature, is quite efficient for problem instances of any size. Regarding the filtering mechanism of the search space, the conclusion is that it is efficient to make a reduction in direct ratio with the problem size, and that this reduction should be lower when the local search is used. For large instances with sizes 30 or 40 the best choice is a total reduction ($\delta = 0$) even with the local search. The greedy algorithm produces solutions much better in average than random schedules. However for medium size problems the GA reaches better solutions starting from random chromosomes than starting from heuristic ones, but for large problems the effect of heuristic seeding is similar to the effect of the local search. Summarizing, the N_3 local search is always quite efficient for medium and large problem instances, and the filtering and seeding mechanisms may be used to adjust the ratio between the size of the effective search space and the running time and consequently is more and more efficient as long as the problem size increases.

As future work we will extend the proposed combined approach to other scheduling problems such as for example the JSS with setup times [3].

References

1. Bierwirth, C.: A Generalized Permutation Approach to Jobshop Scheduling with Genetic Algorithms. OR Spectrum **17** (1995) 87–92.
2. Carlier, J. and Pinson, E. Adjustment of heads and tails for the job-shop problem. European Journal of Operational Research **78** (1994) 146–161.
3. Cheung, W., Zhou, H. Using Genetic Algorithms and Heuristics for Job Shop Scheduling with Sequence-Dependent Setup Times. Annals of Operational Research **107** (2001) 65–81.
4. Dell Amico, M., Trubian, M. Applying Tabu Search to the Job-shop Scheduling Problem. Annals of Operational Research **41** (1993) 231–252.
5. Giffler, B. Thomson, G. L.: Algorithms for Solving Production Scheduling Problems. Operations Reseach **8** (1960) 487–503. (1960).
6. Jain, A. S. and Meeran, S. Deterministic job-shop scheduling: Past, present and future. European Journal of Operational Research **113** (1999) 390–434.
7. Mattfeld, D. C.: Evolutionary Search and the Job Shop. Investigations on Genetic Algorithms for Production Scheduling. Springer-Verlag, November (1995).
8. Varela, R., Vela, C. R., Puente, J., Gmez A. A knowledge-based evolutionary strategy for scheduling problems with bottlenecks. European Journal of Operational Research **145** (2003) 57–71.
9. Varela, R., Serrano, D., Sierra, M. New Codification Schemas for Scheduling with Genetic Algorithms. LNCS 3562, Springer-Verlag (2005) 11–20.
10. Yamada, T. and R. Nakano. Scheduling by Genetic Local Search with multi-step crossover. Fourth Int. Conf. On Parallel Problem Solving from Nature (PPSN IV), Berlin, Germany, (1996) 960–969.

Hebbian Iterative Method for Unsupervised Clustering with Automatic Detection of the Number of Clusters with Discrete Recurrent Networks

Enrique Mérida-Casermeiro and Domingo López-Rodríguez

Department of Applied Mathematics
University of Málaga, Málaga, Spain
{merida, dlopez}@ctima.uma.es

Abstract. In this paper, two important issues concerning pattern recognition by neural networks are studied: a new model of hebbian learning, as well as the effect of the network capacity when retrieving patterns and performing clustering tasks. Particularly, an explanation of the energy function when the capacity is exceeded: the limitation in pattern storage implies that similar patterns are going to be identified by the network, therefore forming different clusters.

This ability can be translated as an unsupervised learning of pattern clusters, with one major advantage over most clustering algorithms: the number of data classes is automatically learned, as confirmed by the experiments. Two methods to reinforce learning are proposed to improve the quality of the clustering, by enhancing the learning of patterns relationships.

As a related issue, a study on the net capacity, depending on the number of neurons and possible outputs, is presented, and some interesting conclusions are commented.

1 Introduction

In 1949, Hebb [2], introduced a physiological learning method based on the reinforcement of the interconnection strength between neurons. It was explained in the following terms:

> When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased.

This kind of learning method has been widely applied to recurrent networks in order to store and retrieve patterns in terms of their similarity. Models that used this learning rule were the bipolar model (BH) presented by J. J. Hopfield in 1982 [4], representing a powerful neural model for content addressable memory, or its analog version [5], among others. These networks, although successful in

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 241–250, 2006.

solving many combinatorial optimization problems, present two main problems when used as content-addressable memory: their low capacity and the apparition of spurious patterns.

The capacity parameter α is usually defined as the quotient between the maximum number of patterns to load into the network and the number of used neurons that obtains an acceptable error probability (usually $p_{\text{error}} = 0.05$ or 0.01). It has been shown that this constant is approximately $\alpha = 0.15$ for BH.

This value means that, in order to load K patterns, more than $\frac{K}{\alpha}$ neurons will be needed to achieve an error probability equal to p_{error}. Or equivalently, if the net is formed by N neurons, the maximum number of patterns that can be loaded in the net (with that error constraint) is $K < \alpha N$.

Recently, in [8], a multivalued recurrent network is used to avoid the undesirable apparition of spurious patterns by using the so-called *augmented patterns* (the method presented in that work is also valid for BH). Patterns are correctly retrieved if sufficiently separated.

The main idea of this work holds that when patterns are very close each other, or if the net capacity is exceeded, then local minima corresponding to similar patterns tend to be combined, forming one unique local minimum. So, although considered as a limitation of the net as associative memory, this fact can explain the way in which the human brain form concepts: several patterns, all of them similar to a common typical representative, are associated (as in the vector quantization), and form a group in which particular features are not distinguishable.

Obviously, enough samples are needed to generalize and not to distinguish their particular features. If there exist few samples from some class, they will still be retrieved by the net individually, that is, as an associative memory.

2 MREM Model with Semi-parallel Dynamics

Let $\mathcal{H}$ be a recurrent neural network formed by N neurons, where the state of each neuron i is defined by its output V_i, $i \in \mathcal{I} = \{1, 2, \ldots, N\}$ taking values in any finite set $\mathcal{M} = \{m_1, m_2, \ldots, m_L\}$. This set does not need to be numerical.

The state of the network, at time t, is given by a N-dimensional vector, $\boldsymbol{V}(t) = (V_1(t), V_2(t), \ldots, V_N(t)) \in \mathcal{M}^N$. Associated to every state vector, an energy function, characterizing the behavior of the net, is defined:

$$E(\boldsymbol{V}) = -\frac{1}{2} \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{I}} w_{ij} f(V_i, V_j) + \sum_{i \in \mathcal{I}} \theta_i(V_i) \tag{1}$$

where $w_{i,j}$ is the weight of the connection from the j-th neuron to the i-th neuron, $f : \mathcal{M} \times \mathcal{M} \to \mathbb{R}$ can be considered as a measure of similarity between the outputs of two neurons, usually verifying the similarity conditions mentioned in [8,10]:

1. For all $x \in \mathcal{M}$, $f(x, x) = c \in \mathbb{R}$.
2. f is a symmetric function: for every $x, y \in \mathcal{M}$, $f(x, y) = f(y, x)$.
3. If $x \neq y$, then $f(x, y) \leq c$.

and $\theta_i : \mathcal{M} \to \mathbb{R}$ are the threshold functions. Since thresholds will not be used for content addressable memory, henceforth we will consider θ_i be the zero function for all $i \in \mathcal{I}$.

The introduction of this similarity function provides, to the network, of a wide range of possibilities to represent different problems [6,7,8,9,10,11]. So, it leads to a better representation than other multivalued models, as SOAR and MAREN [1,12], since in those models most of the information enclosed in the multivalued representation is lost by the use of the signum function that only produces values in $\{-1, 0, 1\}$.

In every instant, the net evolves to reach a state of lower energy than the current one.

In this work, we have considered discrete time and semi-parallel dynamics, where only one neuron is updated at time t. The next state of the net will be the one that achieves the greatest descent of the energy function by changing only one neuron output.

Let us consider a total order in $\mathcal{M}$. The potential increment when a-th neuron changes its output from V_a to $l \in \mathcal{M}$ at time t, is $U_{a,l}(t) = -\Delta E$:

$$U_{a,l}(t) = \frac{1}{2} \sum_{i \in \mathcal{I}} [w_{a,i} f(l, V_i(t)) + w_{i,a} f(V_i(t), l) - (w_{a,i} f(V_a(t), V_i(t)) + \\ + w_{i,a} f(V_i(t), V_a(t)))] - \frac{1}{2} w_{aa} [f(l, l) - f(V_a(t), V_a(t))] \quad (2)$$

If f verifies the similarity conditions, then the *reduced potential increment* is obtained:

$$U^*_{a,l}(t) = -\frac{1}{2} \sum_{i \in \mathcal{I}} [(w_{a,i} + w_{i,a}) \cdot (f(V_i(t), l) - f(V_i(t), V_a(t)))] \quad (3)$$

We use the following updating rule for the neuron outputs:

$$V_a(t+1) = \begin{cases} l, & \text{if } U_{a,l}(t) \geq U_{b,k}(t) \forall k \in \mathcal{M} \text{ and } \forall b \in \mathcal{I} \\ V_a(t), & \text{otherwise} \end{cases} \quad (4)$$

This means that each neuron computes in parallel the value of a L-dimensional vector of potentials, related to the energy decrement produced if the neuron state is changed. The only neuron changing its current state is the one producing the maximum decrease of energy.

It has been proved that the MREM model with this dynamics always converges to a minimal state [7]. This result is particularly important when dealing with combinatorial optimization problems, where the application of MREM has been very fruitful [6,7,8,9,10,11].

3 MREM as Auto-Associative Memory

Now, let $X = \{\boldsymbol{X}_k : k \in \mathcal{K}\}$ be a set of patterns to be loaded into the neural network. Then, in order to store a pattern, $\boldsymbol{X} = (x_i)_{i=1,2,\ldots,N}$, components of the W matrix must be modified in order to make $\boldsymbol{X}$ the state of the network with minimal energy.

As pointed out in [8,10], since energy function is defined as in Eq. (1), we calculate $\frac{\partial E}{\partial w_{ij}} = -\frac{1}{2} f(V_i, V_j)$ and we modify the components of matrix W in order to reduce the energy of state $\boldsymbol{V} = \boldsymbol{X}$ by the rule $\Delta w_{i,j} = -\alpha \frac{\partial E}{\partial w_{i,j}} = \frac{\alpha}{2} f(x_i, x_j)$ for some $\alpha > 0$.

Particularly, for $\alpha = 2$ (every $\alpha > 0$ produces functionally equivalent networks, see [8,10]), it results:

$$\Delta w_{i,j} = f(x_i, x_j) \tag{5}$$

and considering that, at first, $W = 0$, that is, all the states of the network have the same energy and adding over all the patterns, the next expression is obtained:

$$w_{i,j} = \sum_{k \in \mathcal{K}} f(x_{ki}, x_{kj}) \tag{6}$$

Equation (6) is a generalization of *Hebb's postulate of learning*, because the weight w_{ij} between neurons is increased in correspondence with their similarity.

It must be pointed out that, when bipolar neurons and the product function are used, $f(x, y) = xy$, the well-known learning rule of patterns in the Hopfield's network is obtained.

In order to recover a loaded pattern, the network is initialized with the known part of that pattern. The network dynamics will converge to a stable state (due to the decreasing of the energy function), that is, a minimum of the energy function, and it will be the answer of the network. Usually this stable state is next to the initial one.

But, as explained in [8,10] in terms of similarity between vectors associated to different states, when the bipolar Hopfield network loads the pattern $\boldsymbol{X}$, by Eq. (5) with $f(V_i, V_j) = V_i V_j$, the energy of other states is also reduced and the opposite state is also loaded. States with local minimum energy and no associated with input patterns are called *spurious states* and loading spurious states is usually considered an undesirable effect.

4 Spurious Patterns

In [8,10], a technique to avoid the apparition of spurious states when loading pattern $\boldsymbol{X} = (x_i)$ in BH is presented.

Definition 1. *Given a state $\boldsymbol{V}$ of the net, and a similarity function f, its associated matrix ($G_{\boldsymbol{V}}$) is defined as a $N \times N$ matrix whose elements are $G_{i,j} = f(V_i, V_j)$.*

In addition, it can be defined its associated vector ($\boldsymbol{G_V}$) as the vector with N^2 components obtained by expanding matrix $G_{\boldsymbol{V}} = (G_{i,j})$ into vector form, $\boldsymbol{G_V} = (G_k)$, verifying $G_{j+N(i-1)} = G_{i,j}$.

Definition 2. *Suppose that $\boldsymbol{X} = (x_1, x_2, \ldots, x_N)$ is a pattern to be loaded in the net, and $\mathcal{M} = \{m_1, m_2, \ldots, m_L\}$. The* **augmented pattern** *associated to $\boldsymbol{X}$ is the vector $\hat{\boldsymbol{X}} = (x_1, x_2, \ldots, x_N, m_1, m_2, \ldots, m_L) \in \mathcal{M}^{N+L}$ whose components are $\hat{x_i} = x_i$ if $i \leq N$ and $\hat{x_i} = m_{i-N}$ if $i > N$.*

It must be pointed out that, in order to load augmented patterns into the net, only N neurons are necessary, since last L components are clamped to a fixed value. It is only necessary to consider the weights $w_{i,j}$ associated to these components.

Theorem 1. *The function $\Psi : \hat{\boldsymbol{X}} \to \boldsymbol{G}_{\hat{\boldsymbol{X}}}$, that associates every augmented pattern to its associated vector, is injective.*

Proof. Please refer to [8].

So, instead of loading a pattern $\boldsymbol{X}$ with $x_i \in \mathcal{M} = \{m_1, m_2, \ldots, m_L\}$, its associated augmented pattern $\hat{\boldsymbol{X}}$ can be loaded into the net. The state $\hat{\boldsymbol{X}}$ will be the only one maximizing the energy decrease, since the term $W_{\hat{\boldsymbol{X}}} = (f(\hat{x}_i, \hat{x}_j))_{i,j}$ is added to the previously computed weight matrix W, as indicated by Eq. (6).

5 Some Remarks on the Capacity of the Net

In [8], Mérida et al. find an expression for the capacity parameter α for MREM model in terms of the number of neurons N and the number of possible states for each neuron, L, for the case in which N is big enough to apply the Limit Central Theorem ($N \geq 30$):

$$\alpha(N, L) \approx \frac{1}{N} + \frac{\left(\frac{A^2}{z_\alpha{}^2} - B\right)}{NC} \tag{7}$$

where $A = N + 3 + \frac{(N-1)(4-L)}{L}$, $B = \frac{8(N-1)(L-2)}{L^2}$, $C = \frac{8N}{L}$ and z_α is obtained by imposing the condition that the maximum allowed error probability in retrieving patterns is p_{error}. For $p_{\text{error}} = 0.01$, we get $z_\alpha \approx 2.326$.

Some facts can be extracted from the above expression.

For a fixed number of neurons, capacity is not bounded above: Suppose N fixed. Equation (7) can be rewritten in the following form:

$$\alpha(N, L) \approx \frac{1}{N^2 z_\alpha^2} \left[2L + 4(N-1) + z_\alpha^2 + \frac{2(N-1)(N-1+z_\alpha^2)}{L} \right]$$

If we make L tend to ∞, we get $\lim_{L\to\infty} \alpha(N, L) = \infty$, since the coefficient of L in this expression is positive.

What actually happens is that $\alpha(N, \cdot)$, as a function of L, has a minimum at the point $L_0(N) = \sqrt{(N-1)(N-1+z_\alpha^2)} \approx N + 1$ for $z_\alpha = 2.326$. It is a decreasing function for $L < L_0(N)$ and increasing for $L \geq L_0(N)$.

One consequence of this result is that, for appropriate choice of N and L, the capacity of the net can be $\alpha(N, L) > 1$.

This fact can be interpreted as a adequate representation of the multivalued information, because, to represent the same patterns as MREM with N and L fixed, BH needs NL binary neurons and therefore the maximum number of stored patterns may be greater than N. So it is not a strange thing that the capacity

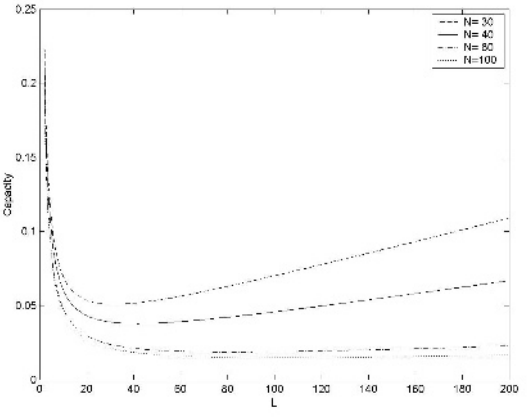


Fig. 1. Representation of the capacity α of the network versus the number of possible states of the neurons, L, for different values of N, the number of neurons

can reach values greater than 1, if the patterns are multivalued, MREM needs much less neurons to represent the pattern than BH.

For a fixed number of possible outputs, capacity is bounded below by a positive constant: Suppose L is fixed. Equation (7) can be rewritten as follows:

$$\alpha(N,L) \approx \frac{1}{z_\alpha^2 L}\left[2 + \frac{4(L-1)+2z_\alpha^2}{N} + \frac{2(L-1)^2+(L-2)z_\alpha^2}{N^2}\right]$$

It can be easily seen that this expression represents a function whose value decreases as N grows. So, a net with more neurons than other, and the same possible states, will present less capacity than the second one. Thus, a minimum positive capacity can be computed for each possible value of L, verifying $\alpha_{\min}(L) = \lim_{N\to\infty} \alpha(N,L) = \frac{2}{z_\alpha^2 L} > 0$.

$\alpha_{\min}(L)$ coincides with the asymptotic capacity for the net with L possible neuron outputs. For example, if $L = 2$ (as in BH), an asymptotic capacity of $\alpha_{\min}(2) = 0.1847$ is obtained, exactly the capacity for BH provided in other works [3].

All these facts can be observed in Fig. 1, representing the value of the parameter $\alpha(N,L)$ for $N \in \{30, 40, 80, 100\}$ and $L \in \{2, 3, \ldots, 200\}$.

6 When Capacity is Exceeded

This work tries to explain what may happen psychologically in the human brain. When a reduced number of patterns has to be memorized, the brain is able to remember all of them when necessary. Similarly, when the capacity of the net is not exceeded, the net is able to retrieve exactly the same patterns that were loaded into it. But when the brain receives a great amount of data to be recognized or classified, it distinguishes between some groups of data (in an unsupervised

way). This kind of behavior is also simulated by neural networks, as we will show next, proving the power (and adequation) of the model herein presented.

Then, learning rules as Hebb's (or the more general given by Eq. (5)), where connection between neurons is reinforced by the similarity of their expected outputs, may produce classifiers that discover some knowledge from the input patterns, like the actual number of groups in which the data is divided. Then, an unsupervised clustering of the input pattern space is automatically performed.

If a pattern, say $\boldsymbol{X}$, is to be loaded in the net, by applying Eq. (5), a local minimum of the energy function E is created at $\boldsymbol{V} = \boldsymbol{X}$. If another pattern $\boldsymbol{X'}$ is apart from $\boldsymbol{X}$, its load will create another local minimum. But, if $\boldsymbol{X}$ and $\boldsymbol{X'}$ are close each other, these two local minima created by the learning rule will be merged, forming one local minima instead.

Then, if a group of patterns is loaded into the net (overflowing its capacity), and all of them are close each other, only one local minimum will be formed, and at the moment of retrieving these data, the unique pattern to be retrieved will be associated to the state of minimum energy. So, patterns can be classified by the stable state of the net which they converge to.

This technique has also the advantage of time. Its execution time only depends on the number of patterns, and it is totally independent of the number of classes in which the data is divided into.

7 Learning Reinforcement

Equation (5) for the learning rule, generalization of Hebb's one, shows that the only thing taking part in updating weight matrix is the pattern to be loaded into the net at that time. So, it represents a very 'local' information, and does not take account of the possible relationships that pattern could have with the already stored ones. So, it is convenient to introduce an additional mechanism in the learning phase, such that the information concerning to relationships between patterns is incorporated in the update of the weight matrix. In what follows, we will consider that the similarity function is $f(x, y) = 2\delta_{x,y} - 1$, that is, its value is 1 if $x = y$ and -1 otherwise.

Learning Reinforcement Method: Suppose that we have the (augmented) pattern $\boldsymbol{X}_1$ stored in the net. So, we have the weight matrix $W = (w_{i,j})$. If pattern $\boldsymbol{X}_2$ is to be loaded into the network, by applying Eq. (5), components of matrix ΔW are obtained.

If $w_{i,j}$ and $\Delta W_{i,j}$ have positive signum (both values equal 1), it means that $X_{1i} = X_{1j}$ and $X_{2i} = X_{2j}$, indicating the relationship between X_{1i}, X_{1j}, X_{2i} and X_{2j}. If both are negative valued, something similar happens, but with inequalities instead of equalities.

So, the fact of $w_{i,j}$ and $\Delta W_{i,j}$ having the same signum is a clue of a relationship that is repeated between components i and j of patterns $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$. In order to reinforce the learning of this relationship, we propose a novel technique, presenting also another kind of desirable behavior: The model proposed before, given by Eq. (6), is totally independent of the order in which patterns

are presented to the net. This fact does not actually happen in the human brain, since every new information is analyzed and compared to data and concepts previously learned and stored.

So, to simulate this kind of learning, a method named **LR** is presented:

Let us multiply by a constant, $\beta > 1$, the components of matrices W and ΔW where the equality of signum is verified, i.e., the components verifying $w_{i,j} \cdot \Delta W_{i,j} > 0$. Hence the weight matrix learned by the network is, after loading pattern $\boldsymbol{X}_2$:

$$w'_{i,j} = \begin{cases} w_{i,j} + \Delta W_{i,j} & \text{if } w_{i,j} \cdot \Delta W_{i,j} < 0 \\ \beta[w_{i,j} + \Delta W_{i,j}] & \text{if } w_{i,j} \cdot \Delta W_{i,j} > 0 \end{cases} \tag{8}$$

Similarly, if there are some patterns $\{\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_R\}$ already stored in the network, in terms of matrix W, and pattern $\boldsymbol{X}_{R+1}$ is to be loaded, matrix ΔW (corresponding to $\boldsymbol{X}_{R+1}$) is computed and then the new learning rule given by Eq. (8) is applied.

It must be noted that this method satisfy Hebb's postulate of learning quoted in Sec. 1.

This learning reinforcement technique, **LR**, has the advantage that it is also possible to learn patterns one by one or by blocks, by analyzing at a time a whole set of patterns, and comparing the resulting ΔW to the already stored in the net. Then, for instance, if $\{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_R\}$ has already been loaded into the net in terms of matrix W, we can load a whole set $\{\boldsymbol{Y}_1, \ldots, \boldsymbol{Y}_M\}$ by computing $\Delta W = (\sum_{k=1}^{M} f(y_{ki}, y_{kj}))_{i,j}$ and then applying Eq. (8).

Iterative Learning Reinforcement Method: LR can be improved in many ways. In this section, an iterative approach, **ILR**, to enhance the solution given by **LR** is presented.

Suppose that, by using Eq. (8) of **LR**, matrix W_X related to pattern set $X = \{\boldsymbol{X}_k : k \in \mathcal{K}\}$ has been learned and denote by $\varphi_{W_X}(\boldsymbol{X}_k)$ the stable state reached by the network (with weight matrix W_X) when beginning from the initial state given by $\boldsymbol{V} = \boldsymbol{X}_k$.

Then, the cardinal of $\{\varphi_{W_X}(\boldsymbol{X}_k) : k \in \mathcal{K}\}$ is (no multiplicities) the number of classes that **LR** finds, n_{a}.

$\varphi_{W_X}(X) := \{\varphi_{W_X}(\boldsymbol{X}_k) : k \in \mathcal{K}\}$ can be considered (with all multiplicities included) as a new pattern set, formed by a noiseless version of patterns in X. So, if applying a second time **LR** to $\varphi_{W_X}(X)$, by using Eq. (8) to build a new matrix $W_{\varphi_{W_X}(X)}$, better results are expected than in the first iteration, since the algorithm is working with a more refined pattern set.

This whole process can be repeated iteratively until a given stop criterion is satisfied. For example, when two consecutive classifications assign each pattern to the same cluster.

8 Simulations

In order to show the ability of **LR** and **ILR** to perform a clustering task as mentioned in Sec. 6, several simulations have been made whose purpose is the clustering of discrete data.

Table 1. Average clustering results on 10 runs of these algorithms, where n_a indicates the number of obtained clusters, p_{cc} is the correct classification percentage (that is, the percentage of simulations in which $n_a = n$) and Err. is the average error percentage

	$n=3$						$n=4$						$n=5$					
	LR			**ILR**			**LR**			**ILR**			**LR**			**ILR**		
K	n_a	p_{cc}	Err.	n_a	p_{cc}	Err.	n_a	p_{cc}	Err.	n_a	p_{cc}	Err.	n_a	p_{cc}	Err.	n_a	p_{cc}	Err.
75	3.1	90	0.53	3.1	90	0.53	4.7	60	3.46	4.1	90	0.26	6.0	40	7.33	5.2	80	2.00
150	3.3	70	0.33	3.1	90	0.13	4.8	70	1.06	4.4	80	0.73	7.5	20	4.00	6.2	30	2.80
300	3.7	70	0.60	3.3	80	0.43	5.5	30	3.06	4.6	60	4.80	7.7	0	3.16	5.8	50	0.50
450	3.5	70	0.37	3.1	90	0.17	4.9	60	0.55	4.4	80	0.31	7.2	10	4.93	6.0	40	4.44
600	3.2	80	0.25	3.2	80	0.25	5.4	50	0.61	4.6	50	0.31	8.4	30	3.03	7.0	50	2.88
750	3.3	80	0.06	3.0	100	0.00	5.4	30	3.49	4.6	60	3.12	10.6	10	9.88	8.1	10	5.56
900	3.2	90	3.23	3.0	100	0.00	5.5	20	0.56	4.8	40	0.42	8.3	10	3.30	6.3	40	2.76
Av.	3.3	78	0.76	3.1	90	0.21	5.2	46	1.82	4.5	66	1.42	7.9	17	5.09	6.3	43	2.99

Several datasets have been created, each of them formed by K 50-dimensional patterns randomly generated around n centroids, whose components were integers in the interval $[1, 10]$. That is, the n centroids were first generated and input patterns were formed from them by introducing some random noise modifying one component of the centroid with probability 0.018. So, the Hamming distance between input patterns and the corresponding centroids is a binomial distribution $B(50, 0.018)$. Patterns are equally distributed among the n clusters. It must be noted that patterns may have Hamming distance even 5 or 6 from their respective centroid, and new clusters can be formed by this kind of patterns.

So, a network with $N = 50$ neurons taking value in the set $\mathcal{M} = \{1, \ldots, 10\}$ has been considered. The parameter of learning reinforcement has been chosen $\beta = 1.5$. It has been observed that similar results are obtained for a wide range of values of β.

The results obtained in our experiments are shown in Table 1. It can be observed not only the low classification error (from 0% to 9.88% on average), but in addition these new techniques get the exact, or very approximate, number of groups in which the pattern set is actually divided in almost every simulation. In fact, whenever the number n_a of discovered clusters equals n, an error percentage of 0% is obtained, retrieving in those cases the initial centroids. It can also be verified that **ILR** clearly outperforms **LR** in most cases, getting a more accurate classification and improving the estimation of the number of clusters, as seen in the last row of the table.

9 Conclusions

In this work, we have explained that the limitation in capacity of storing patterns in a recurrent network has not to be considered as determinant, but it can be used for the unsupervised classification of discrete patterns.

The neural model MREM, based on multivalued neurons, has been developed, as a generalization of the discrete Hopfield model and as an auto-associative memory, improving some of the undesirable aspects of the original Hopfield model: some methods and results for the net not to store spurious patterns in the learning phase have been shown, reducing so the number of local minima of the energy function not associated to an input pattern.

By applying a slight modification to Hebb's learning rule, a mechanism to reinforce the learning of the relationships between different patterns has been introduced to the first part of the process, incorporating knowledge corresponding to several patterns simultaneously. This mechanism can be repeated iteratively to enhance the relationship learning procedure.

In addition, simulations confirm the idea expressed in this work, getting optimal results of classification in many cases.

This work presents a new open research line, from the fact that some new modifications of the learning rule, reinforcing other aspects of the relationships among patterns, may be developed.

References

1. M. H. Erdem and Y. Ozturk, *A New family of Multivalued Networks*, Neural Networks **9,6**, 979-989, 1996.
2. D. O. Hebb, *The Organization of Behavior*, New York: Wiley, 1949.
3. J. Hertz, A. Krogh and R. G. Palmer, *Introduction to the theory of neural computation*, Lecture Notes Volume I. Addison Wesley, 1991.
4. J.J. Hopfield, *Neural networks and physical systems with emergent collective computational abilities*, Proc. of National Academy of Sciences USA, **79**, 2254-2558, 1982.
5. J. J. Hopfield, *Neurons with graded response have collective computational properties like those of two-state neurons*, Proceedings of the National Academy of Sciences USA, **81**, 3088-3092, 1984.
6. D. López-Rodríguez and E. Mérida-Casermeiro, *Matrix Bandwidth Minimization: A Neural Approach*, ICCMSE, **1**, 324-327, 2004.
7. E. Mérida Casermeiro, *Red Neuronal recurrente multivaluada para el reconocimiento de patrones y la optimización combinatoria*, Ph. D. dissertation (in spanish). Univ. Málaga, España, 2000.
8. E. Mérida Casermeiro and J. Muñoz-Pérez, *MREM: An associative autonomous recurrent network*, Journal of Intelligent and Fuzzy Systems 12(3-4), 163-173, 2002.
9. E. Mérida Casermeiro, J. Muñoz Pérez and R. Benítez Rochel, *A recurrent multivalued neural network for the N-queens problem*, Lecture Notes in Computer Science **2084**, 522-529, 2001.
10. E. Mérida Casermeiro, J. Muñoz-Pérez and M. A. García-Bernal, *An Associative Multivalued Recurrent Network*, IBERAMIA 2002, 509-518, 2002.
11. E. Mérida-Casermeiro and D. López-Rodríguez, *Multivalued Neural Network for Graph MaxCut Problem*, ICCMSE, **1**, 375-378, 2004.
12. Y. Ozturk and H. Abut, *System of associative relationships (SOAR)*, Proceedings of ASILOMAR, 1997.

Heuristic Perimeter Search: First Results

Carlos Linares López

Planning and Learning Group
Computer Science Department
Universidad Carlos III de Madrid
Avda. de la Universidad, 30
28911 Leganés - Madrid (Spain)
carlos.linares@uc3m.es

Abstract. Since its conception, the perimeter idea has been understood as a mean for boosting single-agent search algorithms when solving different problems with the same target node, t. However, various results emphasize that the most remarkable contribution of perimeter search is that it is an efficient way for improving the original heuristic estimations. Henceforth, a natural question arises: whether it is feasible or not to increase even more the capabilities for improving $h(\cdot)$ when using a perimeter-like approach. As it will be shown, the so-called "*heuristic perimeter*" idea can be widely considered as an alternative to the classical perimeter and as a baseline for the research in this area.

1 Introduction

Heuristic functions, denoted as $h(\cdot)$ are the most prominent component of any heuristic single-agent search algorithm. They result from insight of the search domain and provide guidance to the search algorithm by qualifying the descendants of any node as more or less promising. Moreover, it has been proved that in presence of a perfect heuristic function (i. e., if $h(n,m) = k^*(n,m)$ where $k^*(n,m)$ is the optimal cost of the path between nodes n and m), A* and related single-agent search algorithms expand nodes only on the optimal path $\pi_{n,m}$, whereas a *blind* heuristic function (i. e., $h(n,m) = k, \forall n,m$ where k is a constant) results in a brute-force search [1]. Therefore, many researchers have tried to capture the importance of heuristic functions and how they affect the performance of single-agent search algorithms, tipically in terms of running time and/or number of node expansions. Most noticeable, Korf, Reid and Edelkamp have shown how to predict accurately the performance of a single-agent search algorithm using a mathematical characterization of the heuristic function employed [2].

As a result, a great effort has been invested (and is still being invested) in devising automatic procedures for improving the accuracy of any heuristic function, $h(\cdot)$: criticizing relaxed models of the original heuristic function [1,3,4]; automatic learning methods of new and more accurate heuristic values [5,6] and dynamic improvement of heuristic functions [7,8].

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 251–260, 2006.

On the other hand, the perimeter idea simulatenously and independently devised by Manzini [9] and Nelson and Dillenburg [10], resulted in a faster search algorithm. It consists of two stages

First, creation of a perimeter (denoted as $\mathcal{P}_{p_d}$) around the target node t which contains all the nodes whose cost for reaching t is less or equal to a predefined value p_d, known as the perimeter depth, so that at least one child have a cost strictly bigger than p_d. This perimeter is usually created using a brute-force search algorithm such as depth-first search.

Second, search from the start state s using a single-agent search algorithm (such as IDA* [11]) guided by the heuristic function h_{p_d}, defined as:

$$h_{p_d} = \min_{m \in \mathcal{P}_{p_d}} \{h(n,m) + k^*(m,t)\} \tag{1}$$

that forces the forward search to approach the perimeter nodes instead of the target node t from any node n outside the perimeter. In the preceding formula, $k^*(m,t)$ will be lower than or equal to the perimeter depth p_d, and $h(n,m)$ is the heuristic distance from the node n to the perimeter node m.

Though it was conceived as a way for boosting single-agent search algorithms when solving different problems with the same target node, t (so that the perimeter $\mathcal{P}_{p_d}$ can be reused over and over again), it has been recently shown that perimeter single-agent search algorithms like BIDA* are faster than their unidirectional counterparts even if they generate the perimeter everytime they start solving a new problem [12]. A careful analysis of the behaviour of the perimeter search algorithms revealed that the most significant contribution of the perimeter idea is that it improves the values of the original heuristic function, resulting in a more informed heuristic function known as $h_{p_d}(\cdot)$ (i. e., $h_{p_d}(n,m) \geq h(n,m) \forall n,m$), at the cost of incrementing the number of heuristic evaluations [13]. Thus, the perimeter idea can be also seen as a natural mean for improving the heuristic estimations of $h(\cdot)$.

So far, a natural question arises: whether it is possible or not to improve the perimeter, $\mathcal{P}_{p_d}$, generated by the classical approach resulting in an even better informed heuristic function, $h_{p_d}(\cdot)$. Section 2 introduces a new algorithm, hBIDA*, which employs a new sort of perimeters known as *heuristic perimeters*; Section 3 discusses some empirical results and section 4 ends up with some relevant conclusions.

2 hBIDA*

Because the original perimeter idea makes use of a brute-force search algorithm to create the perimeter $\mathcal{P}_{p_d}$, it seems reasonable to employ the heuristic function $h(\cdot)$ for generating an *informed* perimeter set, $\mathcal{P}_{p_d,\Delta}$, which shall be biased towards the start state, s.

Definition 1. *A node m belongs to the heuristic perimeter $\mathcal{P}_{p_d,\Delta}$ if and only if $k^*(m,t)+\zeta(m) \leq \Delta$ and $\exists m_j \in SCS(m)$ such that $k^*(m_j,t)+\zeta(m_j) > \Delta$ where $SCS(m)$ are the successors of node m and $\zeta(m)$ is defined as follows:*

Table 1. Pseudocode of the heuristic perimeter generation

```
    HeuristicPerimeter (m, g(m), P_{p_d,Δ})
(1) IF (g(m) + ζ(m) ≤ Δ)
(2)     GENERATE all k children of m : m_j|_{j=1}^{k}
(3)     FOR (j from 1 to k)
(4)         IF (k*(m_j, t) + ζ(m_j) > Δ)
(5)             add m to P_{p_d,Δ}
(6)             RETURN
(7)     FOR (j from 1 to k)
(8)         HeuristicPerimeter (m_j, g(m_j), P_{p_d,Δ})
```

$$\zeta(m) = \begin{cases} -p_d, & k^*(m,t) \leq p_d \\ +\delta(m) = h(s,m) - h_0, & k^*(m,t) > p_d \end{cases} \tag{2}$$

where $h_0 = h(s,t)$. □

Table 1 shows the pseudocode of the procedure for setting up the heuristic perimeter for any combination of p_d and Δ. A key distinctive feature of hBIDA* and pure bidirectional search is that perimeter nodes $m_j \in \mathcal{P}_{p_d,\Delta}$ are not the terminal nodes of a search tree generated by a single-agent search algorithm like IDA*, even if $p_d = 0$. Indeed, the backward search or perimeter creation returns at step (6), adding node m to $\mathcal{P}_{p_d,\Delta}$ in step (5) as soon as a child m_j has been observed to exceed the thresold Δ in step (4). However, IDA* would have gone deeper through the descendants of m for which $k^*(m_j,t) + \zeta(m_j) \leq \Delta$. To see this, consider that outside the classical perimeter, $\mathcal{P}_{p_d}$, $k^*(m_j,t) > p_d$ so that $\zeta(m_j) = \delta(m_j) = h(s,m_j) - h_0$. Thus, the evaluation performed in (2) yields $k^*(m_j,t) + h(s,m_j) - h_0 \leq \Delta$, that is, $k^*(m_j,t) + h(s,m_j) \leq h_0 + \Delta$ where $h_0 + \Delta$ is the thresold employed by the IDA* in every iteration.

Figure 1 shows the differences between the classical perimeter and the heuristic perimeter. In this example, the following assumptions have been done for the sake of clarity: all edges from a state n to any of its successors, n_i, have a cost equal to 1, i. e., $c(n,n_i) = 1$; it has been also assumed that the only allowed changes of the heuristic function when estimating the distance to the same node, m, from a node n and any of its successors, n_i, are -1 and +1, i. .e, $|h(n,m) - h(n_i,m)| = 1$.

In both cases, a perimeter depth, p_d, equal to 2 units has been considered around the target node t. However, the heuristic perimeter with $\Delta = 2$ augmentes the classical perimeter, $\mathcal{P}_{p_d}$ by adding other nodes that are more likely to lay on the optimal path from s to t as suggested by the heuristic function $h(\cdot)$. The figure shows the values $\delta(m) = h(s,m) - h_0$ for all nodes generated during the creation of the perimeter. Let us consider, for example, node A. As it can be seen, $k^*(A,t) + \zeta(A) = 5 + \delta(A) = (\text{since } k^*(A,t) > p_d) = 5 + (-3) = \Delta = 2$. On the other hand, for any of its successors, $k^*(m,t) + \zeta(m) = 6 + \delta(m) = 6 + (-2) = 4 > \Delta$. Therefore, A shall be part of the heuristic perimeter denoted

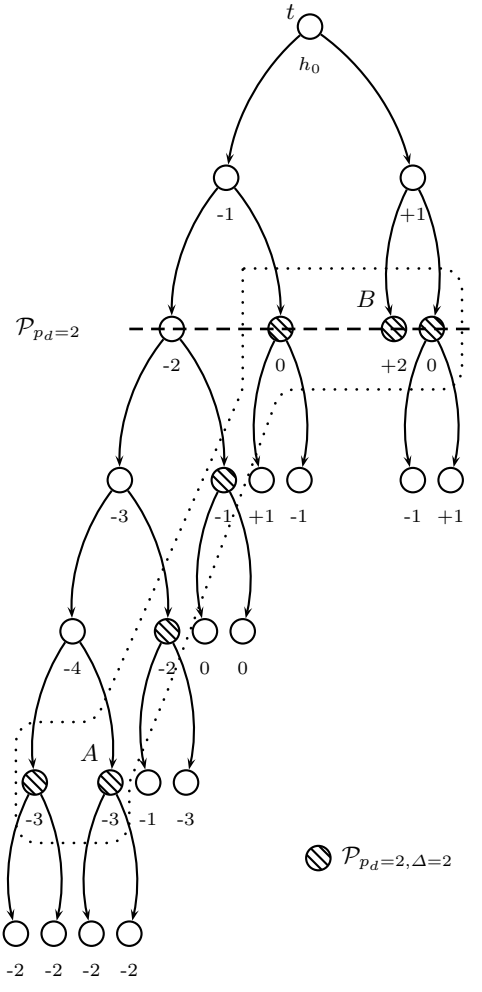


Fig. 1. Heuristic perimeter $\mathcal{P}_{p_d=2,\Delta=2}$ and Classical Perimeter $\mathcal{P}_{p_d=2}$

as $\mathcal{P}_{p_d=2,\Delta=2}$ or simply $\mathcal{P}_{2,2}$ for short. Nevertheless, any classical parameter P_{p_d} can be also derived with the Heuristicperimeter procedure depicted in table 1 as stated in the following result[1].

Lemma 1. *Classical perimeters, $\mathcal{P}_d$, are strictly equal to heuristic perimeters generated with a perimeter depth, $c_* + d$ and $\Delta = -c_*$, i. e., $\mathcal{P}_{p_d=c_*+d,\Delta=-c_*} = \mathcal{P}_d$ where*

$$c_* = min\{c(n, n_i), \forall n, n_i\}$$

if $h(\cdot)$ is a monotonic function, i. e., if:

$$h(n, m) \leq c(n, n_i) + h(n_i, m) \qquad \square$$

[1] Due to space restrictions, the proofs of lemmas and theorems have been suppressed.

Once the heuristic perimeter has been generated, hBIDA* starts a forward search from the start state, s, using the IDA* search algorithm guided by the heuristic function shown in (1) and noted as $h_{p_d,\Delta}$. As it can be seen in figure 1, the perimeter heuristic function h_{p_d} is not guaranteed to improve the original heuristic estimation between s and t, $h(s,t) = h_0$. As a matter of fact, the leftmost node of the perimeter $\mathcal{P}_2$ provides the minimum value over all $m_j \in \mathcal{P}_2$: $h(s,m_j)+k^*(m_j,t) = h_0-2+2 = h_0$. However, the heuristic estimation provided by $h_{2,2}$ is h_0+2 since for all nodes $m_j \in \mathcal{P}_{2,2}$ but node B, $h(s,m_j)+k^*(m_j,t) = h(s,t) + \Delta = h_0 + 2$. On the other hand, node B has the following heuristic estimation: $h(s,B) + k^*(B,t) = h(s,t) + \delta(B) + k^*(B) = h_0 + 2 + 2 = h_0 + 4$. This result can be further generalized as stated in the following theorem.

Theorem 1. *$h_{p_d,\Delta}(m_j,t) \geq h(s,t) + \Delta - c(m,m_j), \forall m_j \in \mathcal{P}_{p_d,\Delta}$. In other words, $h_{p_d,\Delta}(\cdot)$ is guaranteed to improve the heuristic estimation of $h(\cdot)$ by, at least, $\Delta - c(m,m_j)$ units, if $h(\cdot)$ is a monotonic function, for any $\Delta \geq 0$.*

This result needs some further discussion. The rationale behind it is that $h_{p_d,\Delta}(\cdot)$ is always more informed than $h(\cdot)$ for *some* $\Delta > 0$. However, this is not true for *any* Δ value. Let us consider again the assumptions of figure 1: $c(n,n_i) = 1 \forall n_i \in \mathrm{SCS}(n)$ and $h(\cdot)$ is a monotonic function such that the differences $|h(n,m) - h(n_i,m)| = c(n,n_i) = 1 \forall n_i \in \mathrm{SCS}(n)$, which is the case of the domains tested in section 3. In this case, a node m generated at depth d during the backward search has a value $k^*(m,t) + \delta(m)$ strictly even. Therefore, odd values of Δ has to be corrected by substracting $c(m,m_j)$ to get the minimum margin by which $h_{p_d,\Delta}(\cdot)$ is strictly more informed than $h(\cdot)$. On the other hand, $h_{p_d}(\cdot)$ is not guaranteed at all to improve the original heuristic estimation, $h_0 = h(s,t)$, and the maximum improvement it can achieve is $2p_d$ in our domain. So with $\Delta = 2(1+p_d)$, $h_{p_d,\Delta}(\cdot)$ is guaranteed to be strictly more informed than $h_{p_d}(\cdot)$ —it is worth noting that 2 units have been added to $2p_d$ so that the resulting Δ value is still even.

Because hBIDA* operates in the same fashion as BIDA*, the latter can be seen as an instance of the former since the classical perimeters computed by BIDA* can be obtained also by hBIDA* as shown in lemma 1. From this consideration, the following conclusion can be drawn.

Lemma 2. *$hBIDA^*_{p_d=c_*+d,\Delta=-c_*} = BIDA^*_d$*

This result is especially relevant since it states that the same reduction in the active perimeter set observed by Manzini [9] happens in hBIDA*.

Definition 2. *The active perimeter set $\mathcal{P}_{p_d,\Delta}(n)$ of a node n generated in the forward search is defined as follows:*

$$\mathcal{P}_{p_d,\Delta}(n) = \{m_j \in \mathcal{P}_{p_d,\Delta} | h(n,m_j) + k^*(m_j,t) \leq \eta\}$$

where η is the thresold used in the current iteration of the IDA.*

The following theorem proves that the reduction in the active perimeter set also happens in hBIDA*.

Theorem 2. *$\mathcal{P}_{p_d,\Delta}(n) \supseteq \mathcal{P}_{p_d,\Delta}(n_i)$ where n is a node generated in the forward search and $n_i \in \mathrm{SCS}(n)$ and $h(\cdot)$ is a monotonic function.*

This is a significant result, since for the computation of $h_{p_d,\Delta}(n,t)$ it is strictly necessary to perform $|\mathcal{P}_{p_d,\Delta}(n)|$ heuristic evaluations with the original heuristic function, $h(\cdot)$. Fortunately, monotonic heuristic functions guarantee that this number decreases with the depth of the forward search for both the BIDA* and hBIDA*.

3 Results

Two different domains have been selected for testing the performance of hBIDA*: the Korf's test suite for the 15-Puzzle and mazes.

3.1 15-Puzzle

The IDA* search algorithm was the first algorithm solving the whole Korf's test suite which consists of 100 instances of different complexity [11]. The BIDA* is also known to be able to solve all cases faster than IDA* even if the perimeter is generated for every case [12].

A specialized implementation of hBIDA* for the 15-Puzzle using the Manhanttan Distance has been programmed and compared with the performance of $\mathrm{hBIDA}^*_{3,-1}$ which is strictly equivalent to BIDA^*_2 (according to lemma 1 since $c(n,n_i) = 1 \forall n, n_i$ so that $c_* = 1$), the fastest perimeter search algorithm up-to-date. Using precompiled tables, like those devised by Korf, makes that expanding nodes and computing their heuristic value takes a negligible amount of time.

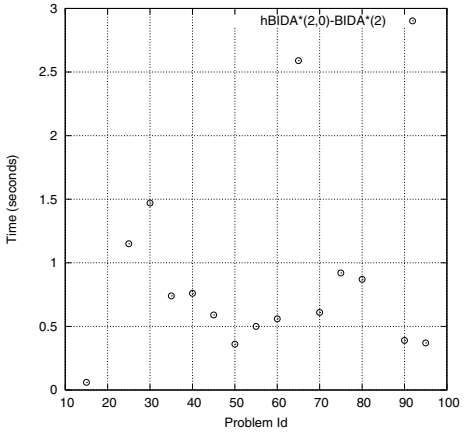


Fig. 2. $\mathrm{hBIDA}^*_{2,0}$ and BIDA^*_2 in the 15-Puzzle

Hence, the heuristic perimeter has been generated with a small perimeter depth, namely $p_d = 2$ units and $\Delta = 0$ thus, avoiding the generation of large perimeter sets.

Surprisingly, both algorithms took exactly the same amount of time for solving the whole test suite, 861 seconds. However, it is not true that both algorithms took the same amount of time for solving each case. Figure 2 shows the difference of running time when BIDA^*_2 ran faster than $\text{hBIDA}^*_{2,0}$ —problem ids are increasingly sorted by the number of nodes generated by IDA*. As it can be seen in the figure, there is a general trend and the margin between both search algorithms when BIDA^*_2 runs faster than $\text{hBIDA}^*_{2,0}$ decrements with the complexity of the instance to solve. Moreover, $\text{hBIDA}^*_{2,0}$, not being worst than BIDA^*_2 for solving all cases, is more efficient when solving the most complex instances. Indeed, $\text{hBIDA}^*_{2,0}$ solved the 10 most difficult instances in 504 seconds, 8 seconds less than the BIDA^*_2, which took 512 seconds.

3.2 Mazes

Finding shortest paths in a maze is known to be a hard problem for IDA* since the difference between the original heuristic estimate, $h_0 = h(s,t)$ and the optimal cost, C^*, tends to be very large. As a matter of fact, IDA* performs $1+(C^*-h_0)/2$ iterations if the cost of every arc, $c(n,n_i)$ is always equal to 1 and the heuristic function $h(\cdot)$ is the sum of the difference of both coordinates x and y (the so-called Manhattan distance) so that $|h(n,t)-h(n_i,t)| = 1$. Henceforth, the nodes closer to the source square, s are re-visited too many times, thus, slowing down the performace of IDA*.

Some experiments have been conducted over two different sizes: small, 100×100 and medium size, 500×500. In both cases, mazes are built randomly:

Since it is guaranteed there is just one path from any source square, s to any other location of the maze, there will be a number of paths of varying length from s. Using a very high perimeter depth, p_d will reduce the number of perimeter nodes, m_j, since many of these paths will have a lenght lower than p_d. This is the main reason why the perimeter approach is expected to be far faster than IDA* in this domain.

Figure 3 shows the running time of both $\text{BIDA}^*_{p_d}$ and hBIDA^*_{0,p_d} for various perimeter depths, $0 \leq p_d \leq 650$, in steps of 10 units, for solving a small test suite with ten cases whose optimal cost ranges from 833 to 1633 in a maze randomly selected with 100×100 squares. As it can be seen, both algorithms have a very similar performance but hBIDA* clearly outperforms BIDA* in most cases. The fastest version of both algorithms are BIDA^*_{630} and $\text{hBIDA}^*_{0,600}$. The former took 0.28 seconds for solving the whole test and the latter 0.24 seconds. It might seem these figures are not very representative, but it should be taken into account that IDA* took 2.34 seconds for solving all cases. In other words, the heuristic improvement due to the classical perimeter idea speeds up the IDA* algorithm by a factor of eight. Employing the heuristic perimeter idea introduced herein, the improvement is almost one order of magnitude.

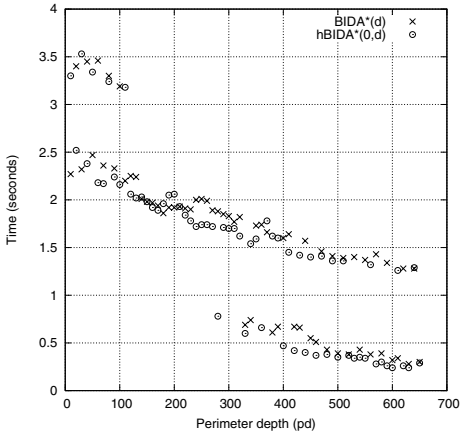


Fig. 3. $\text{BIDA}^*_{p_d}$ vs. hBIDA^*_{0,p_d} in a 100×100 maze

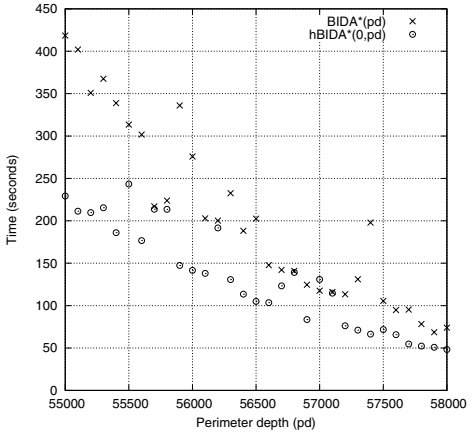


Fig. 4. $\text{BIDA}^*_{p_d}$ vs. hBIDA^*_{0,p_d} in a 500×500 maze

Figure 4 shows the running time of both $\text{BIDA}^*_{p_d}$ and hBIDA^*_{0,p_d} for various perimeter depths, $55,000 \leq p_d \leq 58,000$ in steps of 100 units for solving 30 cases whose solution length is over 60,000 squares. In all cases, but one (with $p_d = 57,000$), hBIDA* turned out to be the fastest algorithm. Of all, $\text{hBIDA}^*_{0,58,000}$ was the fastest one and regarding BIDA* the best performance was achieved with $\text{BIDA}^*_{57,900}$. Whereas the latter took 73.97 seconds for solving the whole test suite, the former took only 48.16 seconds, i. e., $\text{hBIDA}^*_{58,000}$ was roughly a 35% faster than $\text{BIDA}^*_{57,900}$. In any case, improving the heuristic function using a perimeter-like approach (either classical or heuristic based) resulted in

an important speed up since IDA* spent 21,319 seconds for solving the whole test suite. In other words, whilst BIDA* has been able to run almost 300 times faster than IDA*, hBIDA* was about 450 times faster than its unidirectional counterpart. This achievement is due solely to the improvement of the original heuristic estimation, $h(\cdot)$. The differences in the runtime between both perimeter search algorithms are attributed to the differences between $h_{p_d}(\cdot)$ and $h_{p_d,\Delta}(\cdot)$.

4 Conclusions

Perimeter search algorithms have been studied within the scope of this paper as a natural mean for improving the accuracy of the heuristic function, $h(\cdot)$, instead as a mean for re-using off-line computation results in order to speed up the performance of subsequent executions when heading towards the same goal state, t. In previous work by other researchers, it has been shown that the improvement achieved with the classical perimeter idea is by far enough for boosting the unidirectional search algorithm IDA* when solving the Korf's test suite in the 15-Puzzle. Therefore, this paper dealt with the following problem: whereas there is still room for improving $h_{p_d}(\cdot)$ even more or not. A new perimeter approach has been introduced resulting in a heuristic procedure.

From a general point of view, our discussion has not been restricted to a few domains. Indeed, the only requirement set over $h(\cdot)$ is that it shall be a monotonic function, what is usually the case. It is worth noting that the same requirement was also established by the classical perimeter idea. Hence, the heuristic perimeters introduced herein do not impose additional constraints or requirements over the original heuristic function, $h(\cdot)$ or the applicability domain.

Nevertheless, our first experiments have been conducted employing the Manhattan distance over two different domains which is known to be likely to be improved as mentioned before. As a matter of fact, our results support this conjecture.

Regarding the relative improvement of the classical perimeter idea when being compared with the new perimeter approach, it is worth stressing the importance of the following conclusions:

- BIDA$^*_{p_d}$ has been shown to be a sub-class of hBIDA$^*_{p_d,\Delta}$.
- Whilst the classical perimeter heuristic function, h_{p_d}, is not guaranteed at all to be more informed than $h(\cdot)$, it has been shown it is possible to build heuristic perimeters that result in a more informed heuristic function $h_{p_d,\Delta}(\cdot)$.
- Moreover, with the suitable values for p_d and Δ, it can be also guaranteed that $h_{p_d,\Delta}(\cdot)$ will be strictly more informed than $h_{p_d}(\cdot)$.
- Certainly, the improvement of the original heuristic function, $h(\cdot)$, happens at the cost of increasing the number of heuristic evaluations at every node generated by the forward search. It has been shown, however, that hBIDA$^*_{p_d,\Delta}$ do also make use of the so-called active perimeter sets which are guaranteed to decrease with the depth of the forward search.

– hBIDA$^*_{p_d,\Delta}$ has been shown to be more effective than its specialized version, BIDA$^*_{p_d}$, especially when solving the hardest instances.

Acknowledgments

This paper benefited from interactions with Sara Linares Ventosa.

References

1. Pearl, J.: Heuristics. Addison-Wesley, Reading MA (1984)
2. Korf, R.E., Reid, M., Edelkamp, S.: Time complexity of iterative-deepening-A*. Artificial Intelligence **129**(1–2) (2001) 199–218
3. Hansson, O., Mayer, A., Yung, M.: Criticizing solutions to relaxed models yields powerful admissible heuristics. Information Sciences **63** (1992) 207–222
4. Korf, R.E., Taylor, L.A.: Finding optimal solutions to the twenty-four puzzle. In: Proceedings AAAI-96. (1996) 1202–1207
5. Culberson, J.C., Schaeffer, J.: Searching with pattern databases. In: Advances in Artificial Intelligence. Springer-Verlag (1996) 402–416
6. Korf, R.E., Felner, A.: Disjoint pattern database heuristics. Artificial Intelligence **134** (2002) 9–22
7. Auer, A., Kaindl, H.: A case study of revisiting best-first vs. depth-first search. In: Proceedings ECAI-04, Valencia (Spain) (2004) 141–145
8. Kaindl, H., Kainz, G.: Bidirectional heuristic search reconsidered. Journal of Artificial Intelligence Research **7** (1997) 283–317
9. Manzini, G.: BIDA*: an improved perimeter search algorithm. Artificial Intelligence **75** (1995) 347–360
10. Dillenburg, J.F., Nelson, P.C.: Perimeter search. Artificial Intelligence **65** (1994) 165–178
11. Korf, R.E.: Depth-first iterative-deepening: An optimal admissible tree search. Artificial Intelligence **27** (1985) 97–109
12. Linares, C., Junghanns, A.: Perimeter Search Performance. In: Computers and Games: Third International Conference, CG 2002. Volume 2883 of Lecture Notes in Computer Science. Springer-Verlag (2002) 345–359
13. Linares, C.: On the Heuristic Performance of Perimeter Search Algorithms. In: Selected Papers from the 10th Conference of the Spanish Association for Artificial Intelligence (CAEPIA-03). Volume 3040 of Lecture Notes in Computer Science. Springer-Verlag (2004) 445–456

Image Disorder Characterization Based on Rate Distortion

Claudia Iancu[1], Inge Gavat[1], and Mihai Datcu[2]

[1] "Politehnica" University Bucharest,313 Splaiul Independetei 060029, Romania
i_claudia@lpsv.pub.ro, igavat@alpha.imag.pub.ro
[2] German Aerospace Center DLR , Oberpfaffenhofen, Germany
mihai.datcu@dlr.de

Abstract. Rate distortion theory is one of the areas of information transmission theory with important applications in multimodal signal processing, as for example image processing, information bottleneck and steganalysis. This article present an image characterization method based on rate distortion analysis in the feature space. This space is coded using clustering as vector quantization (k-means). Since image information usually cannot be coded by single clusters, because there are image regions corresponding to groups of clusters, the rate and distortion are specifically defined. The rate distortion curve is analyzed, extracting specific features for implementing a database image classification system.

1 Introduction

This article presents a method based on rate distortion theory for characterizing the image "complexity", in order to be used in the analysis of the remote sensing data by FP6 European Commission project "Technology to Support Sustainable Humanitarian Crisis Management", STREAM. The project's purpose is to develop a viable method to process satellite images, in order to find mined fields. Because of a certain relation between "complexity and "disorder", and considering that certain anomalies are caused by "disorder", the proposed method can be used to detect anomalies.

Anomaly detection using Earth Observation images refers to the detection of irregularity in the scene that appears unlikely according to a probabilistic or physical model of the scene, for example ammunition elements in a scene which is dominated by vegetation and soil. The image segmentation or classification into a set of structures or objects is a fundamental aspect for understanding the nature of the observed scene. Most of the work in the field is concentrating to solve the grouping of coherent or regular patterns in images. However, the existing methods do not cope with the heterogeneity of typical high resolution images of scenes subject of hazard. Thus, the present work will focus on the elaboration of models to assess anomalies and/or disorder in images.

Through a hierarchical image content characterization, an image retrieval system can be viewed as a composed communication channel [1],[2]. The image coding is effectuated by extracting its primitive features, i.e. color or spectral

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 261–268, 2006.

information, texture or geometrical parameters. According to information theory principles, which say that data processing cannot increase information, each level in the hierarchical scheme is associated with a certain loss of information. The accuracy of communication, e.g. accessing a target image or a category of images as exploration results, depends on the assumed levels of image "coding".

One interesting approach for image classification, applied with good results, uses Kolmogorov interpretation of entropy as a measure of image "disorder" [3].

For classification and coding we will use rate distortion theory as one of the most successful ideas of information transmission theory. Classification and coding is done in view of an optimal digital representation of the signal, leading to a certain "compression" degree. Through compression, the processed signal is "distorted", rate distortion theory ensuring a balance between an acceptable distortion (error) and a satisfactory compression rate.

Two widely used methods for image processing, based on the results of rate distortion theory, are steganalysis [4] and information bottleneck [5].

We will apply rate distortion theory in an attempt to classify images after their information content.

2 Image Characterization Method

This paper proposes an image classification method, ranking images based on a certain "complexity" related to the information content and expressed through a cluster number K in the feature space. This ranking has the following processing scheme:

- image primitive feature extraction (color and texture)
- parametric quantization of the image primitive feature space (vector quantization, e.g. clustering by k-means for variable number of clusters)
- rate distortion analysis of the cluster space
- rate distortion based estimation of the optimal number of clusters.

The cluster space obtained for certain image primitive features is considered a set of codewords, an abstract vocabulary for each feature. Therefore the whole vocabulary of signal classes is decomposed into many vocabularies corresponding to each primitive feature. The image objects or structures, with meaning (i.e. semantics), are combination of different features using the combination of the extracted clusters.

The rate distortion function is defined as the inferior bound of the mutual information for a given distortion measure. Since the mutual information is the information that symbols transmitted convey with symbols received, we characterize the image in terms of its rate distortion for the feature space [6].

In addition, considering the link between the Kolmogorov complexity and the mutual information, and knowing the Kolmogorov complexity describes randomness, in our case relative to the cluster space, as an image "code", we also use a method to describe image "disorder".

3 Primitive Image Features Extraction

To describe the image content, a set of quasi-complete characterization shall be considered. In the present work are used the spectral (color) and textural descriptors.

3.1 Spectral Features

To form our dataset we split a co-registered image (2752 x 883 pixels) of Oberpfaffenhofen, Germany, taken with a Daedalus sensor, having the following specifications: 12 spectral bands in visible and infrared, 1 m resolution, integrated geometric distortion correction, in 16 images, 285 x 285. Thus, the spectral characterization is done by a 12 dimensional vector for each image pixel.

3.2 Texture Features

To extract texture primitive image features is used the autobinomial Gibbs Random Field as data model [7] [8].

The Gibbs autobinomial Random Field family of stochastic models assume that the statistics of the grey level of a pixel in the image depends only on the grey levels of the pixels belonging to a finite dimensions neighbourhood. The probability of the grey level of the pixel x_s is:

$$p(x_s \mid \partial x_s, \theta) = \frac{1}{Z_s} \exp(-H(x_s \mid \partial x_s, \theta)) \tag{1}$$

where Z_s is a normalization factor given by the sum over all the possible states for the pixel x_s. Assumptions have to be made for the functional form of the energy function H. In this study an autobinomial model has been used, in which the energy function is:

$$H(x_s \mid \partial x_s, \theta) = -\ln(\frac{G}{x_s}) - x_s \eta \tag{2}$$

$$\eta = a + \sum_{i,j} b_{ij} \frac{x_{ij} + x_{ij}^{'}}{G}$$

where each b_{ij} parameter describes the interaction between the pixel x_s and the pair $x_{ij}, x_{i,j}^{'}$ (figure 1), the parameter a represents a sort of auto-interaction and G represents the maximum grey value

A fitting of the model on the image is performed, in order to obtain the best fitting parameters; for the estimation a conditional least square estimator was used:

$$\theta_{CLS} = argmin_{\theta}(x_s - \sum_{x_s=0}^{G} x_s p(x_s \mid \partial x_s, \theta))^2 \tag{3}$$

X_{22}	X_{12}	X_{21}'
X_{11}	X_s	X_{11}'
X_{21}	X_{12}'	X_{22}'

Fig. 1. Image pixel neighborhood

The evidence of the model, i. e. the probability of the model given the data, can be calculated by:

$$p(M \mid D) = \frac{p(D \mid M)p(M)}{p(D)} \tag{4}$$

where the probability of the data D can be obtained via the integration:

$$p(D \mid M) = \int p(D \mid \theta, M)p(\theta \mid M)d\theta \tag{5}$$

From the estimated parameters, we derive several features to describe the image content: the norm of the estimated parameters $\mid \hat{\theta} \mid$ as the strength of the texture, the estimate of the variance $\hat{\sigma}^2_M$ and the evidence of the model M [9].

4 A Modified Rate Distortion Function for Image Classification

One can characterize cluster analysis as an attempt to find the best possible representation of a data set using a fixed number of points. This can be thought of as performing compression or quantization of the information contained in the data, and for that some imprecision (distortion) of the original values results. In order to minimize the error, a finite list of representative cluster centers is used, centers placed in the regions of highest density, in other words, where the data are clustered. In this paradigm, each cluster center provides a representation for nearby observations and the distortion gives a measure of the best level of accuracy obtainable using K clusters. The data are well-summarized when the correct number of centers is picked [10].

This is an analogue of the main problem of rate distortion theory. Rate distortion theory is the branch of information theory addressing the evaluation of coding, as accurately and efficiently as possible, the output of a source. Typically the source output consists of a sequence of realisations of a continuous random variable. Representing or transmitting a real number with perfect accuracy requires storing an infinite number of bits, which is not feasible, therefore, a finite

set of codewords is chosen so as to approximate the numbers of source symbols as well as possible.

One can say that the central problem in rate distortion theory is to find the average number of codewords that will be required for a representation. This quantity is referred to as the rate, R, of a code. The minimum rate achievable for any given distortion is called the rate distortion function R(D) and, correspondingly, the minimum distortion achievable for any given rate is the distortion rate function $D(R)$. Essentially, $R(D)$ and $D(R)$ provide a way to formalize how many codewords to use and how successfully [11].

The functions that relate the rate and distortion are found as the solution of the following minimization problem:

$$R(D) = \min_{p(y|x):E_{X,Y}[d(X,Y)]\leq D} I(X,Y) \tag{6}$$

where $p(y|x)$ is the conditional probability density function (pdf) of the communication channel output (compressed signal) Y for a given input (original signal) X, and $I(Y,X)$ is the mutual information between Y and X defined as:

$$I(Y,X) = H(Y) - H(Y \mid X) \tag{7}$$

where $H(Y)$ and $H(Y|X)$ are the entropy of the output signal Y and the conditional entropy of the output signal given the input signal, respectively.

The mutual information can be understood as a measure for prior uncertainty the receiver has about the sender's signal $H(Y)$, diminished by the uncertainty that is left after receiving information about the sender's signal $H(Y|X)$. Of course the decrease in uncertainty is due to the communicated amount of information, which is $I(Y,X)$.

In the definition of the rate-distortion function, $d(X,Y)$ and D are the distortion between X and Y for a given $p(y|x)$ and the prescribed maximum distortion, respectively.

In our approach, the number of clusters, K, is equivalent to the number of codewords, the cluster centers provide canonical representation of members of their respective groups, and the distortion function is mean square error MSE, as follows: if K is the number of clusters, N_i the number of pixels in the cluster i, X_j the six dimensional vector (RGB + texture features) corresponding to the pixel j, C_i the six dimensional vector corresponding to the center of the cluster i, and m indicates the component of the vectors, then the distortion is:

$$MSE = \sum_{i=1}^{K} \sum_{j=1}^{N_i} \sqrt{\sum_{m=1}^{6} (X_{jm} - C_{im})^2} \tag{8}$$

After computing and plotting the rate-distortion function, we analyse the point at which the resulting distortion curve levels off, this point giving the optimal number of clusters, since past this point the drops are much smaller. The number of clusters is proportional to the image information content, so finding the number of clusters also gives us the possibility to rank the images according to their "complexity".

5 Experimental Results

As stated earlier, we performed a series of experiments on a Daedalus super-spectral image, divided into 16 squares (285x285 pixels). As color features we used the first 3 bands, corresponding to R, G, B and as texture features, we computed the norm, variance and evidence of the image. We applied the k-means algorithm and computed the distortion function for each square, as shown in figure 2.

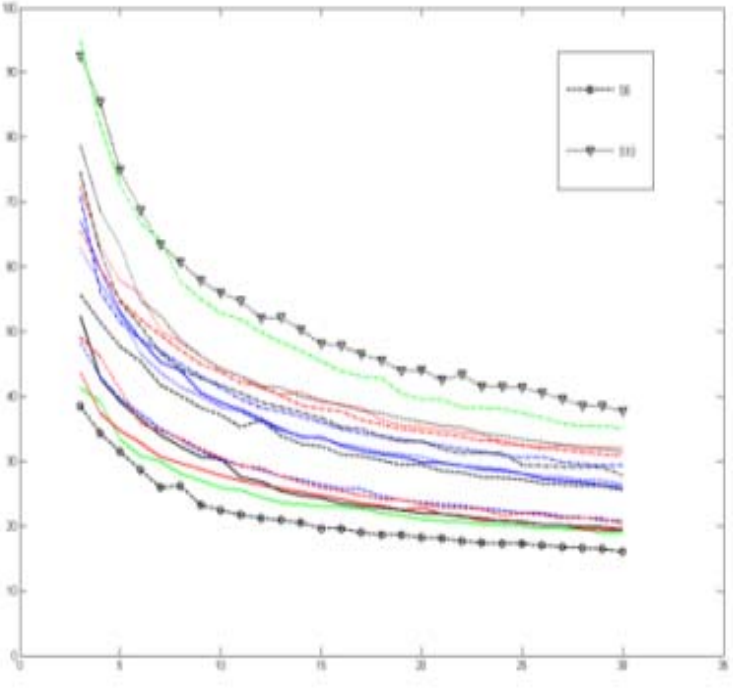


Fig. 2. The distortion function for the 16 squares, computed for 3 to 30 clusters. The highest distortion gives the most complex square (S10) and the lowest distortion gives the simplest square (S6).

In order to emphasize the difference of complexity between squares, we assigned each square a certain level of gray, based on the distortion: the image with the highest distortion, which is also considered the most "complex" image, is represented with white, and the image with the lowest distortion, meaning the "simplest" image is represented with black, thus creating a map of complexity, in grey levels, for the initial image. The initial image and the mapped image are shown in figure 3.

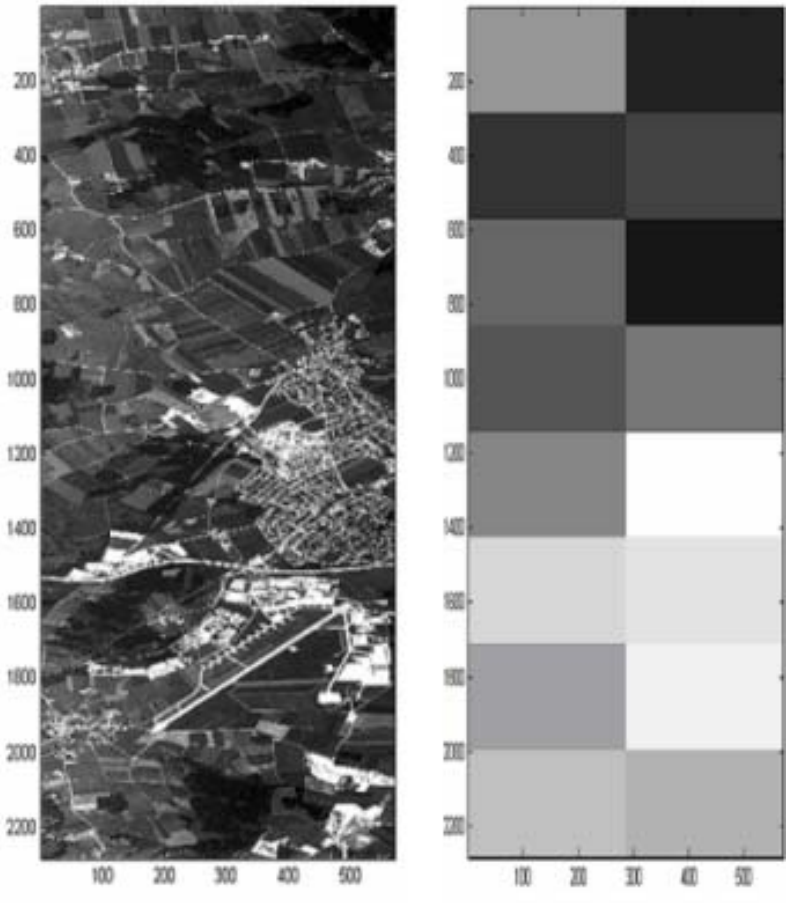


Fig. 3. The initial image and the grey map image: white represents the most complex square, and black the simplest square

6 Conclusions

As one can see, the ranking of images based on their "complexity", realized with the proposed method, corresponds to the ranking resulting after a simple man-made visual inspection.

It is to see that really, the more "complex" region in the image can be characterized for a given error with a larger number N of clusters. For example, if we consider the most complex square, S10, for a number of clusters equal to 30, the obtained error is very high (37%). In order to obtain a lower error, the number of clusters should be much higher, in concordance with its complexity. For the simplest image, S6, an error of 37% is obtained for 3 clusters, while for 30 clusters the error is much lower, 17%. Therefore, we conclude that the classification

algorithm proposed in our paper is efficient, and can be used to automatically rank images after "complexity", expressed as the necessary number of clusters to attain a certain error.

Acknowledgements. *Work in the frame of FP6-EC project -"Technology to Support Sustainable Humanitarian Crisis Management STREAM".*

Daedalus superspectral image was provided for research purposes by Dr. Peter Strobl from the German Aerospace Center, DLR.

References

1. Datcu, M., Seidel. K., D'Elia S., Marchetti P.G: Knowledge - driven information mining in remote sensing image archives. ESA bulletin 110 (2002)
2. Schröder, M., Rehrauer, H., Seidel, K., Datcu, M.: Interactive learning and probabilistic retrieval in remote sensing image archives. IEEE Trans. on Geoscience and Remote Sensing, vol. 38, no. 5 (2000) 2288–2298
3. Faur, D., Datcu, M., Gavat, I.: Mutual Information based measures for image content characterization. Proceedings of 11th Conference of the Spanish Association for Artificial Intelligence CAEPIA 2005, Santiago de Compostela, Spain, November, 16-18, 2005
4. Celik, M. U., Sharma, G., Tekalp, A. M.: Universal image steganalysis using rate-distortion curves. Proc. SPIE: Security, Steganography and Watermarking of Multimedia Contents VI, vol. 5306, San Jose, USA (2004) 19–22
5. Goldberger, J., Greenspan, H., Gordon, S.: Unsupervised Image Clustering using the Information Bottleneck Method. The Annual Pattern Recognition Conference DAGM, Zurich (2002)
6. Tasto, M., Wintz, P.: A bound on the rate-distortion function and application to images. IEEE Transactions on Information Theory, vol.18, issue 1, (1972) 150–159
7. Schröder, M., Walessa, M., Rehrauer, H., Seidel, K., Datcu, M.: Gibbs random field models: a toolbox for spatial information extraction. Computers and Geosciences 26 (2000) 423–432 2000
8. Schröder, M., Rehrauer, H., Seidel, K., Datcu, M.: Spatial information retrieval from remote sensing images:Part II Gibbs Markov Random Field. IEEE Trans. on Geoscience and Remote Sensing, vol. 36, (1998) 1446–1455
9. Datcu, M., Stoichescu, D.A., Seidel, K., Iorga, C.: Model fitting and model evidence for multiscale image texture analysis. American Institute of Physics, AIP Conference Proceedings 735, (2004) 35–42
10. Sugar, C. A., James, G. M.: Finding the Number of Clusters in a Data Set: An Information Theoretic Approach. Journal of the American Statistical Association 98 (2003), 750–763,
11. Blahut, E. R.: Computation of Channel Capacity and Rate-Distortion Functions. IEEE transactions on Information Theory, Vol. IT-18, No. 4, (1972)

Improving the Computational Efficiency in Symmetrical Numeric Constraint Satisfaction Problems

R.M. Gasca, C. Del Valle, V. Cejudo, and I. Barba

Departmento de Lenguajes y Sistemas Informáticos. Universidad de Sevilla. (Spain)
{gasca, carmelo, cejudo, irene}@lsi.us.es

Abstract. Models are used in science and engineering for experimentation, analysis, diagnosis or design. In some cases, they can be considered as numeric constraint satisfaction problems ($NCSP$). Many models are symmetrical $NCSP$. The consideration of symmetries ensures that $NCSP$-solver will find solutions if they exist on a smaller search space. Our work proposes a strategy to perform it. We transform the symmetrical $NCSP$ into a new $NCSP$ by means of addition of symmetry-breaking constraints before the search begins. The specification of a library of possible symmetries for numeric constraints allows an easy choice of these new constraints. The summarized results of the studied cases show the suitability of the symmetry-breaking constraints to improve the solving process of certain types of symmetrical $NCSP$. Their possible speed-up facilitates the application of modelling and solving larger and more realistic problems.

1 Introduction

Symmetries are common in science and engineering applications. It is due to the inherent symmetry of the physical world. Some examples are: electron spin in atoms, inertial changes in mechanics, electromagnetism, some organic chemical compounds, etc... Many problems in these applications exhibit a high degree of symmetry that can be exploited successfully for solving them.

Backtracking and consistency techniques are conventional methods to solve constraint satisfaction problems ($CSPs$). Reducing complexity is the major issue in solving $CSPs$, specially when there is a large number of constraints and/or wide domains of the variables. In first works, the symmetries have been considered in problems where variables have discrete domains and to remove symmetrical solutions by changing the CSP solved [14]. Also, the symmetries are used in the detection and exploitation in planning problems [6], reasoning and optimization [4], the generation of balanced incomplete block designs [13] and applications to low autocorrelation binary sequences [8]. Other approaches consider the design of a new search method that avoids testing of possible symmetrical subsolutions [1] and the addition of constraints during the search [15].

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 269–279, 2006.

Numeric Constraint satisfaction problems ($NCSP$) are more and more often used to solve engineering problems arisen in different areas of Artificial Intelligence (qualitative reasoning, diagnosis, planning, scheduling, configuration, distributed artificial intelligence, etc...). These problems are formed by a set of constraints among variables whose domains are real interval values. Constraint-solving systems have as a goal to find all solutions, only one solution, or if the model that represents the problem is or not consistent.

The symmetrical $NCSP$ ($SNCSP$) has a set of symmetrical properties or symmetrical constraints. These problems that contain symmetries may be solved most efficiently. The search effort can be reduced specially on hard and large problems. In Numerica [16] the symmetry property of $NCSP$ has been considered by means of soft constraints. These constraints are equivalent to other ones except that they are ignored when the existence of solutions is proved.

Reasoning about the ranges of values of variables is a type of reasoning often used when there are inaccurate data or partially defined parameters. It can be also generalized as a $NCSP$. A natural way of reasoning on the ranges of values is to propagate the domains of the variables through the constraints. It involves assigning values to variables in order to satisfy constraints among subsets of those variables. These problems can be solved efficiently by combining local consistency methods, such as approximations of arc-consistency, together with a backtracking-based search. Different problem solving techniques have been proposed in the bibliography [9] [11] [2] [16] [10]. The search space in numeric constraint problems is usually too wide and a lot of these techniques have a major drawback since they introduce choice points and they are exponential. The efficiency of some previous algorithms was analyzed in a previous work [3]. Another work [16] for improving the accuracy and efficiency in solving $NCSP$ has been proposed by reducing dependencies and variable elimination. Our proposal considers that the symmetry property of $NCSP$ can speed-up significantly its solving process.

Although $NCSP$ modeling is the necessary step preceding CSP solving, little work has been done to help modelers. In this line, our method provides a library of symmetries that allows modelers to remove some symmetries in $SNCSP$. The modeler selects and adds symmetry-breaking constraints of this library, reducing the search space. These additions are performed by hand and require a previous analysis of the modeler. It must not affect the soundness and completeness of the solutions. Nevertheless, the modeler must also consider the increase of the computational cost after adding these new constraints in the original $NCSP$. This paper is interested in reformulating concise models that support more efficient solutions.

The rest of the article is organized as follows. In Section 2, we start presenting some examples of $SNCSPs$ to introduce the problem domain. Section 3 presents some definitions and preliminaries. Section 4 exposes a simple library of symmetries to be taken into account in symmetrical $NCSPs$ and proposes the selection of modelling schemas and their solving process. The experimental

results are presented in Section 5. Finally, in the last section we present our conclusions and future work.

2 Illustrative Examples

In order to clarify the aim of this work, the following geometrical problem is very illustrative:

$$Model \equiv \begin{cases} X = \{x, y\} \\ D = \{x, y \in (-\infty, +\infty), \\ C = \{x^2 + y^2 = 4, x * y = 1\} \\ G = AllSolutions(X) \end{cases}$$

For this problem, the search space is $I\!R^2$, but if we consider the symmetries, this space is reduced four times. In fact, this reduction improves significantly the computational complexity of the search process.

Also we have selected two examples from different Artificial Intelligence areas, the first problem is commonly used in order of magnitude reasoning and the second one is an example of a configuration task. Both examples are specified by means of a four-tuple:

- **X**, the set of variables of the model,
- **D**, the set of domains of the variables,
- **C**, the set of relations of the variables of the model, and
- **G** the goal of the model analysis.

This representation allows an easy mapping from this specification to a $NCSP$.

2.1 A Countercurrent Heat-Exchanger

This problem is studied in order of magnitude reasoning [12] and [5]. A schema of a countercurrent heat-exchanger is shown in Figure 1. The important variables in the analysis of the device are the molar-heat KH and the molar-flowrate FH of the hot stream, and the molar-heat KC and the molar-flowrate FC of the cold stream. Also, the temperature differences have been named $DTH = Th_1 - Th_2$, $DTC = Tc_1 - Tc_2$, $DT_1 = Th_1 - Tc_1$, $DT_2 = Th_2 - Tc_2$. The temperature drop of the hot stream is DTH, the temperature rise of the cold stream is DTC and the driving force at the left and right ends of the device are DT_1 and DT_2

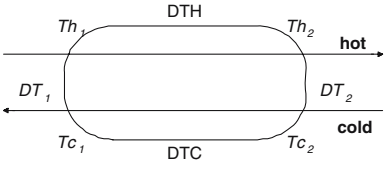


Fig. 1. A countercurrent heat-exchanger

respectively. The numeric constraints of the problem are the energy balance of the exchanger and the result from the definition of the temperature differences: $DTH * KH * FH = DTC * KC * FC$ and $DTH - DT_1 - DTC + DT_2 = 0$. In a particular case, the following order of magnitude relations may be known: DT_2 is *moderately smaller than* DT_1 and DT_1 is *much smaller than* DTH. A possible analysis of the model could consists in obtaining the qualitative relation between FC and FH.

$$Model \equiv \begin{cases} \mathbf{X} = \{DTH, KH, FH, DTC, KC, FC, DT_1, DT_2\} \\ \mathbf{D} = \{DTH, KH, FH, DTC, KC, FC, DT_1, DT_2 \in (-\infty, +\infty), \\ \quad r_1 \in [0.1, 0.9]\ r_2 \in [0, 0.1]\ r_3 \in [0, +\infty)\} \\ \mathbf{C} = \{-DTC * FC * KC + DTH * FH * KH = 0, \\ \quad -DTC + DTH - DT_1 + DT_2 = 0, DT_2 - DT_1\, r_1 = 0, \\ \quad DT_1 - DTH\, r_2 = 0, FC - FH\, r_3 = 0\} \\ \mathbf{G} = Solution(r_3) \end{cases}$$

where r_1, r_2 are the corresponding intervals for the order of magnitude relations and r_3 is the interval for the unknown order of magnitude relation. Due to the symmetrical properties of this model it may be considered as a $SNCSP$.

In the last section, the modeler specifies that he would like to know one solution of the unknown order of magnitude relation.

2.2 Structural Configuration of Resistors

This example presents a given configuration of resistors as shows Figure 2. The modeler has the aim of selecting from two types of resistors whose values may be $[9.8, 10.1]\,\Omega$ or $[99.6, 100.4]\,\Omega$ in order to obtain an equivalent total resistor whose value is $[149.5, 150.5]\ \Omega$ according to the previous configuration. This problem can be modelled by means of the four-tuple below. The goal of the modeler is to search for only one possible solution.

This problem can be solved using the previous mentioned $NCSP$ techniques, but the computational effort is exponential. We propose to exploit the symmetries of this problem in order to achieve better computational costs.

$$Model \equiv \begin{cases} \mathbf{X} = \{R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, R_9, R_{10}, R\} \\ \mathbf{D} = \{R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8, R_9, \\ \quad R_{10} \in [9.8, 10.1], [99.6, 100.4], R \in [149.5, 150.5]\} \\ \mathbf{C} = \{(R_1 * R_2)/(R_1 + R_2) + \\ \quad (R_3 + R_4) * R_5 * R_6/(R_3 + R_4 + R_5 + R_6) + \\ \quad (R_7 + R_8) * (R_9 + R_{10})/(R_7 + R_8 + R_9 + R_{10}) = R \\ \mathbf{G} = OneSolution(X) \end{cases}$$

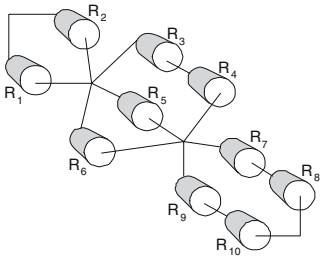


Fig. 2. Configuration problem with resistors

3 Definitions and Notation

We have presented informally $SNCSP$ in terms of examples. This section presents a number of definitions for the formalization of symmetrical NCSP. The variable domains in these problems are real intervals.

Definition 1 (Interval). Let $\mathbb{F}$ denote a finite subset of $\mathbb{R}$ augmented with the symbols $\{-\infty, \infty\}$. If $a, b \in \mathbb{F}$ then an interval $[a, b]$ represents the set of real numbers $\{r \in \mathbb{R} \mid a \leq r \leq b\}$.

Definition 2 (Numeric Variable). A variable of the model whose domain is a real interval. The set of numeric variables of the problem is denoted by X.

Definition 3 (Numeric Constraint). It is a relation (equations) involving a finite subset of numeric variables.

Definition 4 (Goal). A predicate that denotes the users' preferences to search for only one solution, all solutions or the consistent domain of certain variables.

Definition 5 (Numeric Satisfaction Problem). A four-tuple $P = (X,D,C,G)$ where $X = \{x_1, ..., x_n\}$ is a set of variables, whose continuous domains are respectively $D = \{d_1, ..., d_n\} (n \geq 1)$, $C = \{c_1, ..., c_m\} (m \geq 1)$ is a set of numeric constraints and G are the goals.

Definition 6 (Symmetrical NCSP). A $NCSP$ that has symmetry constraints.

Definition 7 (Solution). An instantiation of the numeric variables such that all constraints are satisfied. They correspond to n-dimensional cubes, that are named *hypercubes*.

We use Ψ to denote a $SNCSP(X, D, C, G)$. Ψ has a set of transformations $T_\Psi = \{T_1, ...T_m\}$. Every transformation $T_i \in T_\Psi$ has associated a finite set of expressions over the variables of the problem $\alpha_i = \{\alpha_i^1(X), \alpha_i^2(X), ..., \alpha_i^k(X)\}$ and a set of real regions associated to the previous expressions $\Omega_i = \{\Omega_i^1(X), \Omega_i^2(X), ..., \Omega_i^k(X)\}$. For example in the following $SNCSP$:

$$Model \equiv \begin{cases} X = \{x, y\} \\ D = \{x, y \in (-\infty, +\infty), \\ C = \{x^2 + y^2 = 4, x * y = 1\} \\ G = AllSolutions(X) \end{cases}$$

some transformations are shown in Table 1.

Table 1. Symmetrical transformations of a previous example

Transformations	α	Ω
$T_1 \equiv Permutations(x,y)$	$\alpha_1^1(x,y) \equiv x = y \wedge y = x$	$\Omega_1^1(x,y) \equiv y \leq x$
		$\Omega_1^2(x,y) \equiv y \geq x$
$T_2 \equiv Symmetry(x,y)$	$\alpha_2^1(x,y) \equiv x = -x \wedge y = -y$	$\Omega_2^1(x,y) \equiv x + y \geq 0$
		$\Omega_2^2(x,y) \equiv x + y \leq 0$

Every symmetrical transformation belonging to the set T_Ψ must be invariant. Let $T_i \in T_\Psi$ be a transformation that has associated an expression $X = \alpha_i(X)$. A transformation is invariant if $T_i(\Psi) \equiv \Psi$ where $\equiv$ denotes symbolic equality. The transformation of Ψ is obtained by means of the symbolic substitution of the variables X by expressions $\alpha_i(X)$,

$$T_i(\Psi) = \Psi[\alpha_i(X)/X] \quad i \in \{1, ..., m\}$$

Let $\Omega_i(X)$ be a finite set of associated regions for a transformation $T_i(\Psi)$, where

$$\Omega_i(X) = \{\Omega_i^1(X), \Omega_i^2(X), ..., \Omega_i^k(X)\}$$

The elements of this set must satisfy the following properties:

- Minimum overlapping of regions $\forall s, t : s, t \in 1..k \wedge s \neq t \mid \Omega_i^s(X) \cap \Omega_i^t(X)$ may be a hyperplane of a lower dimension with respect to the dimension of the search space.
- To fill in the search space $\bigcup_j \Omega_i^j(X) = I\!R^n$ where n is the cardinality of the set X and $j \in \{1, 2, ..., k\}$.
- Obtaining the next regions

$$\forall s \in 1..k-1 \mid T_i(\Omega_i^s(X)) = \Omega_i^{s+1}(X)$$

  and if k is finite, then $T_i(\Omega_i^k(X)) = \Omega_i^1(X)$

4 Symmetry Analysis in Symmetrical NCSP

The aim of symmetry analysis is the identification of symmetries and the generation of the corresponding symmetry-breaking constraints. The search for symmetries in the initial $SNCSP$ are performed by hand and requires a previous symmetry analysis. This analysis always must ensure the soundness

and completeness of the search. The symmetry analysis has the following steps:

1. Identification of possible symmetries of a $SNCSP$. These problems may have an infinite number of symmetries and also some symmetries are hard to calculate. It determines that in this work we only identify the simplest symmetries. There are several approaches to identify these symmetries:
   - the symmetry-breaking constraints are derived from the model.
   - the symmetry-breaking constraints are derived from a visual representation of the model
   - Some tools calculate the symmetry-breaking constraints.

   The first item of the above list may be adequate to carry out the identification of the symmetries in symmetrical $NCSP$.
2. Reduction of the domain variables or generation of symmetry-breaking constraints, that are related to the different regions of the transformations.
3. Determination of the expressions to calculate symmetrical solutions and transformation of the variables' domains in order to ensure the completeness of the search. It depends on the users' goals. For example, if the goal is to obtain all the solutions of a $SNCSP$ then we determine these expressions.

4.1 Numeric Symmetries Library

The consideration of symmetries in symmetrical $NCSP$ can be a hard process. There is no known polynomial algorithm to detect all the possible symmetries. This work only considers a partial analysis of the symmetries which covers the most elemental numeric symmetries. It is named Numeric Symmetry Library (NSL) and includes the following symmetries:

- **Permutations**
  1. Transformation: Given a set $X' \subseteq X$ whose elements are $X' = \{x_1, ..., x_m\}$, the number of permutations in this set is $k = m!$.
  2. The regions of the transformation *Permutation* are

$$\Omega^1(X) \equiv x_1 \leq x_2 \leq ... \leq x_m$$

$$\Omega^2(X) \equiv x_2 \leq x_1 \leq ... \leq x_m$$

$$.....$$

$$\Omega^k(X) \equiv x_m \leq ... \leq x_2 \leq x_1$$

     One of these regions can be added to break symmetries.
  3. The symmetric solutions are obtained as the permutations of the previous solutions. According to the notation used before, if s_1 is a solution of Ψ then:

$$s_2 = P^2(s_1), \quad s_3 = P^3(s_2), ..., s_k = P^k(s(k-1))$$

- **Symmetries with respect to a hyperplane with all variables of the SNCSP**
  1. One possible transformation T_i could be $T_i \equiv \alpha_i(X) = -X$
  2. The regions could be
  $$\Omega_i^1 \equiv x_1 + ... + x_n \geq 0$$
  $$\Omega_i^2 \equiv x_1 + ... + x_n \leq 0$$
  3. The symmetric solutions have the same absolute values than solutions obtained in solving the $SNCSP$ but the sign of these values must be changed.
- **Symmetries with respect to a hyperplane with a subset of variables of the SNCSP**
  Given a subset $X' \subset X$ whose variables are $\{x_1, ..., x_m\}$
  1. The transformation in this case could be $T_i \equiv \alpha_i(X') = -X'$
  2. The regions could be $\Omega_i^1 \equiv x_1 + x_2 + ... + x_m \geq 0$ and $\Omega_i^2 \equiv x_1 + x_2 + ... + x_m \leq 0$.
  3. The symmetric solutions of the $SNCSP$ are obtained changing the sign of the following variables $\{x_1, .., x_m\}$ in the solutions of the $SNCSP$.

  The modeler must check if the hyperplane $x_1 + x_2 + ... + x_m = 0$ in $\mathbb{R}^m$ is a symmetry hyperplane for all the surfaces that represent the $SNCSP$.
- **Translations**
  These transformations are convenient in $SNCSPs$ with trigonometric or periodic functions.
  1. The transformation in this case is $T_i \equiv \alpha_i(x) = x + \tau$ where τ is a real number.
  2. The regions are $\Omega_i^1 \equiv 0 \leq x \leq \tau$, $\Omega_i^2 \equiv k \leq x \leq 2 * \tau$, ... The number of regions is determined by the domain of the variable x.
  3. The symmetric solutions are obtained from the original $SNCSP$ such that if s_1 is a solution then $s_2 = s_1 + \tau, s_3 = s_2 + \tau, ...$

Therefore, the symmetrical analysis must determine from the previous *NSL* which type of symmetries are most convenient in a particular $SNCSP$. Modeler must also analyze the pruning of the search space resulted from the addition of symmetry-breaking constraints and the complexity of the treatment of these symmetry-breaking constraints. The efficiency of the numeric constraint solver must be improved in any case.

The modelling of symmetrical $NCSPs$ is oriented to a search process of solutions. A broad variety of approaches have been focused to solve $NCSPs$, essentially, exhaustive or/and local search techniques. Our work considers exclusively the exhaustive approach. Modeler must determine a modelling schema of $SNCSPs$ that considers the symmetries before the search begins.

The key idea is to obtain from a $SNCSP$ an equivalent $NCSP$ by the addition of symmetry-breaking constraints and/or the update domains. This idea can be represented as

$$SNCSP(X, D, C, G) \overset{\text{SymmetryAnalysis}}{\longrightarrow} NCSP(X, D, C', G, T)$$

The initial Symmetrical $NCSP$, denoted as Ψ, is transformed by means of a previous symmetry analysis into a new $NCSP$ where $C \subset C'$ and T are the set of transformations identified in the symmetry analysis. The conversion is shown in the following expression:

$$\Psi \equiv \{X, D, C \wedge \Omega^1_{T_1} \wedge \Omega^1_{T_2} ... \wedge \Omega^1_{T_k}, G\}$$

This modeling methodology proposes that the modeler must perform the symmetry analysis according to the particular syntax of a symmetrical $NCSP$ and *NSL*. It allows the identification of the symmetry-breaking constraints that participate in the modelling schema of this problem. The transformations T, which allows the subsequent calculation of the symmetrical solutions are obtained from *NSL*.

The specification of a $SNCSP$ must consider the symmetries belonging to the *NSL*. Illustrative examples, that we presented in the section 2 of this article, have some symmetries. Table 2 shows the symmetries that we propose in the different example models.

Table 2. Symmetry-breaking constraints of the example models

Problem	Symmetry-breaking constraints
Geometrical	$x \leq y, x + y \leq 0$
Heat-exchanger	$DTH + KH + FH + DTC + KC + FC + DT1 + DT2 \geq 0$
Resistors	$R_1 \geq R_2,\ R_5 \geq R_6, R_7 + R_8 \geq R_9 + R_{10},\ R_7 \geq R_8,\ R_9 \geq R_{10}$

These previous schemas must be solved by means of the adequate algorithm. If the goal is to search for all solutions, then we will apply the corresponding transformations to obtain the symmetric solutions.

5 Experimental Results

We have used the previous examples in the experimentation with some extensions. The heat-exchanger example considers one and two heat exchangers in series. The resistors example considers two configurations with ten and thirteen resistors respectively. The results are a mean value considering different variable domains and goals. All the experiments have been performed on the same machine. The program with schema modelling has been run five times on each problem instance, and the results displayed are an average of these five runs. The application to the different proposed problems is shown in Table 3. These results show solutions calculation without considering the CPU computing time of obtaining symmetry-breaking constraints and symmetrical solutions. In the underconstrained problem (heat-exchanger), the computational time is the same in both cases. It indicates that the reduction of the search space is not compensated with a reduction of the computational time. In the other problems, the

Table 3. Computational results of the symmetry and no symmetry excluding in schema models with bounded solutions

Model	**Symmetry-excluding**		**No symmetry exclusion**	
	Fails/Choice Points	Cpu sec.	Fails/Choice Points	Cpu sec.
Heat-exchanger	7/0	0.015	7/0	0.016
Heat-exchanger2	10/0	0.018	10/0	0.019
Resistors10	232/285	0.083	1343/2009	0.631
Resistors13	1746/2535	1.590	15852/26824	25.839

computational time is reduced. Then we can conclude that the computational efficiency of symmetries depends on the type of symmetrical $NCSP$ that the modeler specifies.

6 Conclusions and Future Work

In this work we propose a strategy to consider symmetry-breaking in symmetrical numeric constraint satisfaction problems. The addition of constrains to break symmetries in these problems reduces the search space. The modeler must consider the tradeoff between the increase of computational treatment of these constraints and the previous reduction what is important in certain underconstrained problems. The use of breaking-symmetries constraints provides significant computational savings in a lot of problems. Their speed-up facilitates the application of modeling and solves larger and more realistic problems.

In future work, we also would like to eliminate the possible redundancies in the calculation of symmetry-breaking constraints to reduce the computational complexity of the $SNCSP$. Another interesting research area is the automatic insertion of symmetry-breaking constraints in runtime, when a new symmetry appears during the search. A future application of the symmetrical reduction of $NCSP$ will be the efficient modelling in engineering projects.

Acknowledgements

This work has been partially supported by the Spanish *Ministerio de Ciencia y Tecnología* through a coordinated research project(grant DIP2003-0666-02-2) and Feder (ERDF).

References

1. Benhamou F. and Sais L. *Theoretical study of symmetries in propositional calculus and applications* in Proceedings of CADE92. (1992).
2. Benhamou F. and Older W. *Applying Interval Arithmetic to Real, Integer and Boolean Constraints*. In *The Journal of Logic Programming* pp. 1-24, (1997).
3. Collavizza H., Delobel F. and Rueher M. *Extending consistent domains of numeric CSP*. In *Proceedings of Sixteenth IJCAI'99*, Stockholm, pp. 406-411, (1999).

4. Crawford J., Ginsberg M., Luks E. and Roy A. *Symmetry-breaking Predicates for search problems*. In *Proc. of KR-96*, pp. 148-159, (1996).
5. Dague P. *Numeric Reasoning with relative orders of magnitude*. In *Proc. of the Thirteenth IJCAI*, Cambery, pp. 541-547, (1993).
6. Fox M. and Long D. *The Detection and Explotation of Symmetry in Planning Problems*. In *Proceedings IJCAI'99* pp. 956-961, (1999).
7. Gent I.P. and Smith B. M. *Symmetry Breaking During Search in Constraint Programming*. In *Report 99.02 University of Leeds*, (1999).
8. Gent I.P. and Smith B. M. *Symmetry Breaking in Constraint Programming*. In *Proc. ECAI 2000*, (2000).
9. Hyvönen E. *Constraint reasoning based on interval arithmetic: the tolerance propagation*. In *Artificial Intelligence* 58, pp. 1-112, (1992).
10. Jussien N. and Lhomme O. *Dynamic domain splitting for numeric CSPs*. In *Proceedings ECAI98*, pp. 224-228, (1998).
11. Lhomme O. *Contribution à la résolution de constraintes sur les réels par propagation d'intervalles*. Ph. D. Nice-Sophia University. Antipolis. (1994).
12. Mavovrouniotis M. L. and Stephanopoulos G. *Formal Order of Magnitude Reasoning in process engineering*. Comput. Chem Engineering 12(9-10), pp. 67-880, (1988).
13. Meseguer P. and Torras C. *Solving Strategies for Highly Symmetric CSPs*. In *Proceedings IJCAI'99* pp. 400-411, (1999).
14. Puget J. F. *On the satisfiability of symmetrical constrained satisfaction problems*. In *Proceedings of ISMIS'93*, pp. 350-361, (1993).
15. Puget J. F. *Symmetry Breaking using stabilizers*. In *Principles and Practice of Constraint Programming- CP 2003 LNCS 2833* pp. 585-589, (2003).
16. Van Hentenryck P., Michel L. and Deville Y. *Numerica. A modeling language for global optimization. The MIT Press*, (1997).

Inferring Multidimensional Cubes for Representing Conceptual Document Spaces

Roxana Danger and Rafael Berlanga

Departament de Llenguatges i Sistemes Informàtics
Universitat Jaume I, Castelló, Spain
{berlanga, roxana.danger}@uji.es

Abstract. This paper proposes a new method for representing document collections with conceptual multidimensional spaces inferred from their contents. Such spaces are built from a set of interesting word co-occurrences, which are properly arranged into taxonomies to define orthogonal hierarchical dimensions. As a result, users can explore and analyze the contents of large document collections by making use of well-known OLAP operators (On-Line Analytic Processing) over these spaces.

Keywords: Multidimensional models, Text Mining, Information Retrieval.

1 Introduction

One of the major challenges of current Information Systems, most of them relying on the Web, consists of the effective processing of the enormous information stream managed by them. It seems clear that the analysis of all this information can be very helpful in the knowledge discovering and decision-making processes within the involved organizations. However, nowadays most of this information is provided as textual documents, which cannot be properly treated by current analysis tools such as Data Warehouses and Multidimensional Databases [8]. Instead, it is necessary to semantically structure the document contents so that they can be represented in a proper conceptual multidimensional space.

In Information Retrieval, one traditional way of structuring large document collections consists of using a hierarchical classifier [7], which makes use of a predefined taxonomy of topics (e.g. Open Directory Project[1]). Basically, a document classifier automatically assigns each document to the topic that better fits its contents. Document classifiers are supervised as they require a training set (i.e. documents manually assigned to topics) to select the features that characterizes each topic. In this way, the resulting document classification can be seen as a mono-dimensional conceptual model for representing the collection. However, this approach is very poor for analysis purposes. Firstly, the classification taxonomy is given a priori and therefore it can be inappropriate for describing

[1] http://www.dmoz.org/

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 280–290, 2006.

the actual contents of the collection. Secondly, topics may be very broad for being useful for analysis tasks, for example "computers", "health" and "cars".

Alternatively, hierarchical document clustering algorithms [11] can be applied to obtaining semantically-related groups of documents, from which hierarchies can be derived. Contrary to document classifiers, these algorithms do not require any training dataset, and the obtained hierarchies correspond to the actual collection contents. However, they are very sensitive to both the chosen document similarity measure and the clustering strategy. Therefore the quality and semantics of the built clusters can be very different from a method to another. Moreover, these algorithms do not provide any information about the contents of clusters, being users unaware of their underlying concepts, nor ensure that the formed hierarchical levels correspond to interesting abstraction levels for analysis purposes [10].

Finally, another way to find the underlying semantics of a document collection consists of finding interesting association rules (e.g. term co-occurrences) with which decision trees or similar informational structures can be built. The former definition of association rule stem from Data Mining works. In them, the most accepted measures of interest are the *support* and *confidence* of frequent item sets [4]. Unfortunately, these techniques have not been successful applied to document collections, mainly due to the great bias that *Zipf law* produces in the distribution of terms.

In this article, we present a new method for automatically building multidimensional models with hierarchical dimensions, with the purpose of conceptually structuring, summarizing and analyzing unknown large textual collections. This method works as follows. Firstly, interesting word co-occurrences are extracted to find out the relevant terms and their contexts (section 3). With these terms, and by using external lexical resources, a specific taxonomy covering all the interesting co-occurrences is built (section 4.1). Finally, orthogonal hierarchical dimensions are inferred from the specific taxonomy according to relevance criteria (section 4.2).

2 Motivation and Proposed Methodology

Vector models have been widely applied to representing documents in Information Retrieval Systems [11]. In these models, each document is represented as a vector where each of its dimensions represents a term of the document collection. Clearly, this model has a very high dimensionality (in the order of ten of thousands) and it is also very sparse. Moreover, many dimensions are not orthogonal because of the high correlation of some terms. Despite these drawbacks, the vector models have been applied successfully to document retrieval, document similarity calculation and automatic document clustering.

Contrary to this kind of models, we propose to use multidimensional models as those proposed for analysis tasks in databases through OLAP operators [6]. These models rely in the concept of *cube*, which is a structure that allows the analysis of a set of *facts* described with both a set of orthogonal hierarchical

dimensions and a set of *measures*. With these models, users can analyze and summarize huge amounts of historical data by selecting their dimensions of interest and the proper detail level at each dimension. Measures are automatically aggregated as users move through the different levels of the dimensions by using the roll-up and drill-down operators [5].

For example, let Person, Group, Activity and Country be three dimensions to represent facts about politics described in a news collection. In this context, Person can include different levels of detail, also called categories, for example: "leader", "functionary" and "exponent". Each of these categories can be further divided into more detailed subcategories, for example "Person.leader" can comprise the subcategories "chief of state", "politico" and "legislator". As a result, dimensions can be viewed as concepts taxonomies that can be arranged into different stratified abstraction levels (see Figure 1). Furthermore, we can include different measures to be analyzed, such as the number of news involved in each fact, the event-time periods and other quantities extracted from the news texts.

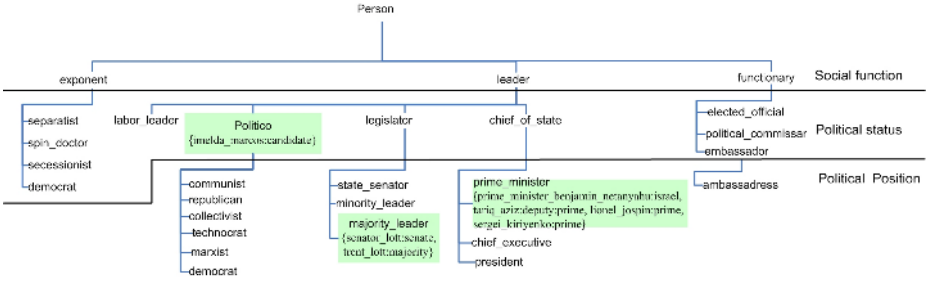


Fig. 1. Dimension example created by our method

This kind of multidimensional models can be very helpful for the analysis of large document collections, mainly in the following tasks:

- **Exploration of document contents.** Starting from the most abstract level at each dimension, users can select and drill-down the dimensions of interest, obtaining as a result the relevance of each category that can guide their information search.
- **Event Detection.** By means of the specification of proper cubes (i.e. set of dimensions at a proper detail level), users can express the event types in which they are interested. Measures can account for the relevance of the obtained facts (e.g. through a TF-IDF weighting) as well as the time-span of the associated events. For example, we can analyze events about "politics activities" with the cube (Person, Activity, Date).
- **Named Entity Recognition.** Named entities extracted from document texts can be associated to the corresponding facts. In this way, if a named entity co-occurs with a value of some dimension category, then the entity can be viewed as an instance of that category value. For instance, if the name

"Pancho García" usually co-occurs with the term "president", which belongs to the category Person.leader.[chief of state], then it can be viewed as a member of it. In this way, some dimensions can have an additional abstraction level (i.e. the lowest one) containing all the named entities associated to the dimension.

3 Obtaining Interesting Word Co-occurrences

Let $\mathcal{CT} = \{Tx_1, \cdots, Tx_N\}$ be a document collection, where each document Tx_i consists of a sequence of terms $Tx_i = (w_1, \cdots, w_m)$. In our applications, terms correspond to meaningful lemmas and phrases (e.g. named entities) directly extracted from texts. We denote $\mathcal{T}$ the set of all the terms in the collection.

A co-occurrence cw is a set of terms that jointly appear in a non-unitary set of documents, which is denoted $hits(cw)$. In collections of complex documents where different logical parts can be identified (e.g. paragraphs, sections, etc.), we can take the collection to be mined as the partition obtained by dividing these complex documents at some logical levels. In this way, users can focus the mining process to the appropriate document granularity in order to find co-occurrences with different strength degrees.

An interest measure for a co-occurrence is a function $f_{interest}(cw)$ that returns the relevance of the co-occurrence for a particular purpose. Interest measures can be based on both the co-occurrence terms and the co-occurrence hit list. The most popular interest measure in data mining is the *support*, which is only based on the length of the co-occurrence hit list. Recently other alternative measures have been proposed, for example the *leverage* [12], which also takes into account the hit lists of their sub-co-occurrences.

In order to manage a small and useful set of co-occurrences, we associate to each term of the collection with the terms that maximize the selected interest measure. In this way, we define the *set of interesting pairs of terms* as follows:

$$\sigma_{f_{interest}}(w_i) = argmax_{w_j \in \mathcal{T}, w_j \neq w_i} f_{interest}(\{w_i, w_j\})$$

$$InterestingPairs(\mathcal{T}) = \{\{w_i, w_j)\} | w_i \in \mathcal{T}, w_j \in \sigma_{f_{interest}}(w_i)\}$$

From this set we can select those pairs where a filter measure (f_{filter})is greater than a given strength threshold α:

$$FilteredPairs(\mathcal{T}, \alpha) = \{pair | pair \in InterestingPairs(\mathcal{T}), f_{filter}(pair) \geq \alpha\}$$

Finally, we calculate all the maximal co-occurrences that cover the filtered interesting pairs previously defined:

$$CoOcc(\mathcal{T}, \alpha) = \{ \bigcap_{d_i \in hits(cw)} terms(d_i) | cw \in FilteredPairs(\mathcal{T}, \alpha)\}$$

In the following sections we will show how these maximal co-occurrences are used to infer a multidimensional model for the document collection $\mathcal{CT}$.

4 Multidimensional Models for Document Collections

Let C be the set of all possible categories and M the set of elements from which categories can take their member values.

Definition 1. *A dimension is a pair $D = (C_d, \leq_{C_d})$, where $C_d \subseteq C \cup All$ (All, a distinguished category) and $\leq_{C_d}$ a partial order of C_d (let $\leq^*_{C_d}$ be the reflexive and transitive clausure of the relation $\leq_{C_d}$). Each category $c \in C_d$ satisfies that $c \leq^*_{C_d} All$ and $c \leq_{C_d} c$. Let $Member : C \rightarrow 2^M$ be a function to assign the set of values of each category. Besides, a dimension satisfies various restrictions over the values of the members of its categories [5]; basically, each value appears in only one category (partitioning) and it is related with only one other value of a more abstract category (upper conectivity).*

Definition 2. *A set of values of a datatype V describes a measure if a summarizing distributive function $fr : 2^V \rightarrow V$ can be defined over it.*

Definition 3. *A multidimensional cube is a n+m-tuple that consists of n-dimensions $D_1, \cdots, D_n$ and m-measures $M_1, \cdots, M_m$. The associated facts to the cube are those $n + m$-tuples $(m_1, \cdots, m_n, v_1, \cdots, v_m)$, where $m_i \in Member(All_{D_i})$, $i \in \{1, \cdots, n\}$ y $v_j \in M_j$, $j \in \{1, \cdots, m\}$.*

In our scenario, we deal with a document warehouse, which mainly comprises textual documents. We distinguish two kinds of dimensions: contents dimensions, which are associated to the concepts directly extracted from document texts, and metadata dimensions, which are associated to document metadata such as the publication date, author, source, etc. Whereas metadata dimensions are directly designed by the warehouse administrator, contents dimensions must be inferred from texts. In this paper, we will focus only in the latter ones.

To build contents dimensions it is necessary to use some kind of external knowledge that gives us information about the semantic relationships between terms or concepts appearing in texts.

In our application, we use external taxonomies defined as pairs $T = (C, <)$, being C the set of concepts and $<$ a partial order over them. Usually the $<$ order defines a "is-a" relationship between concepts, although other semantic relationships can be expressed with it (e.g. part-of hierarchies). The only constraint we impose over $<$ is that it must be transitive.

Regarding the previous issues, we define the dimension inference problem as follows.

Definition 4. *Let $\mathcal{T}$ be the set of terms of a text collection $\mathcal{CT}$. Let f_t be a function that returns the relevance of a term in $\mathcal{T}$, $\{T_1, ..., T_n\}$ be a set of external taxonomies, and f_c be a function that evaluates the relevance of a dimension category. The* problem of extracting contents dimensions *consist of identifying the sub-trees from the external taxonomies that cover all the relevant collection terms according to f_t, and that only contain relevant concepts according to f_c. As a result, each sub-tree will constitute a contents dimension, where each category corresponds to each non-leaf node, and being its members all its child nodes.*

4.1 Building Specific Taxonomies

The aim of the dimension inference problem is to build a new taxonomy from the external ones so that it contains all the relevant terms appearing in the collection texts at the same time that it preserves the concept relationships specified in those taxonomies.

In Section 3 it was shown that *interesting* co-occurrences can identify the relevant terms of a collection as well as the term relationships that frequently occur in documents. For this reason, we will use the set of interesting co-occurrences to define the relevance function of terms as follows:

$$f_t(t) = \begin{cases} 1, & t \in \bigcup_{cw \in CoOcc(\mathcal{T},\alpha)} cw \\ 0, & \text{otherwise} \end{cases}$$

Once relevant terms have been selected, it is necessary to apply a disambiguation algorithm to assign each term to its proper taxonomy concept. We use the correlation between the context of each term and that of each possible concept to which it can be assigned. We define the context of a term as the set of relevant terms that jointly appears with it (i.e. its co-occurrences), whereas the context of a concept is the set of terms associated to the neighbor concepts at the corresponding external taxonomy. Thus, each term will be assigned to the concept that obtains the maximum correlation, if it surpasses a given confidence threshold. As a result, the disambiguation algorithm returns a trio $(term, concept, Tax)$ for each term occurrence.

To build the specific taxonomy over the calculated concepts, we start with a base concept, called All_T, from which all concepts will specialize. Then, for each trio $(term, concept, Tax)$ generated by the disambiguation algorithm, we extract the path of concepts that connects concept to the Tax's root as the intersection of the all paths in the taxonomy Tax. The specific taxonomy is constructed by following those paths.

4.2 Deriving Contents Dimensions

In this section we show how contents dimensions are derived from the specific taxonomy associated to a document collection. Basically, this process consists of selecting a set of disjoint sub-trees that cover only the interesting concepts of the taxonomy. From now on, we will call category to all the non-leaf concepts of the specific taxonomy, being their members the corresponding child concept nodes.

We say that a category is interesting if it appears in at least δ interesting co-occurrences, being δ a given confidence threshold.

Let $hitsco(c)$ be the number of co-occurrences where the concept c is assigned to some of their terms. Taking into account the transitivity of the taxonomy order, the overall hits of a concept can be calculated as follows:

$$hitsAc_{co}(t_T) = \begin{cases} \bigcup_{m \in Member(t_T)} hitsAc_{co}(m) \bigcup hits_{co}(t_T), & if\, t_T \text{ is category} \\ hits_{co}(t_T), & \text{otherwise} \end{cases}$$

The relevance of a category is then defined as follows:

$$f_c(c) = \begin{cases} 1, & |hitsAc_{co}(c)| > \delta \\ 0, & \text{otherwise} \end{cases}$$

By using the relevance of categories, Algorithm 1 extracts all the disjoint sub-trees of the specific taxonomy that can be used as content dimensions. The algorithm works as follows. First, it selects the direct relevant sub-concepts of All_T that will be eventually associated to the content dimensions. For each of them, the algorithm traverses their sub-trees following the corresponding taxonomy relationships and performing the following actions: whenever it finds an irrelevant category, all the leaf nodes of its corresponding sub-tree are moved to its direct parent, and the sub-tree is removed from the taxonomy. As a result, all the leaf nodes belonging to an irrelevant category are transformed into members of their nearest relevant ancestor.

Algorithm 1. Algorithm of dimensions extraction

Require: T, $hitsAc_{co}$, f_c, CB
{ T, specific taxonomy generated from the domain's taxonomies;
$hitsAc_{co}$, co-ocurrencies function of the members of T;
f_c, relevance function of the categories;
CB, base categories for the dimensions; }
Ensure: CD
{CD, Set of dimensions; }
if $Recostructed(All_T) \neq 1$ **then**
Return $CD = \{\}$
end if
Create a dictionary of useful child categories, d_{cu}, which links each category c with the set of child categories c_h of maximun level of abstraction such that $hitsAc_{co}(c) \neq hitsAc_{co}(c_h)$.
for all category, $c \in T$, following the order $\leq_T$ defined by the taxonomy T **do**
if $\neg\exists D = (C_D, \leq_D) \in CD$, $c \in C_D$ y $\exists c_b \in CB$, $c \leq_T c_b$ **then**
Make a new dimension with base concept c and child concepts according to d_{cu}.
end if
end for
Function $Reconstructed(c)$: {c is a category;}
if $f_c(c) = 1$ **then**
for all $c_h \leq_c c$ **do**
if $Reconstructed(c_h) = 1$ **then**
Remove c_h from T and adds to c all the members of c_h with $hitsAc_{co}(c) \neq \{\}$
end if
end for
Return 1
else
Return 0
end if

5 Experiments and Evaluation

Currently we have implemented a prototype of the content dimensions extractor. This prototype makes use of the morpho-syntactic parser Tree-Tagger [2], the lexical database WordNet [2] and the Magnini's domain fields [1]. Magnini's domain fields are used to build different taxonomies from WordNet. More specifically, we take all the frequent combinations of domain fields as the root concepts of the external taxonomies. So, all the resulting dimensions and its concepts are associated to these domains.

To evaluate the proposal, we have tested the prototype over the TDT2[3] collection version 4.0 of the Linguistic Data Consortium (LDC). This collection comprises 9824 news and 55112 lemmatized terms. To evaluate the quality of interesting co-occurrences we have conducted several experiments to compare the obtained co-occurrences with respect to the 178 topics manually identified by LDC annotators (from now on LDC topics). We have selected two quality measures, namely: the traditional F1 measure, which compares the co-occurrences hits lists with respect to the set of documents of each topic, and the cosine measure, which compares the words in the co-occurrence with respect to the words contained in the descriptions of LDC topics. Global quality measures have been obtained by macro-averaging the best pairs topic/co-occurrence that maximize the corresponding quality measure.

Table 1 shows the obtained results for the interest measure $f_{interest}(cw) = |hits(cw)|$ and filter function $f_{filter} = \frac{|hits(cw)|}{min_{w_i \in cw}|hits(w_i)|}$. The way of obtaining the interesting pairs is straightforward from the inverted file of the document collection. Thus, taking the term hit lists ordered by their length, we can calculate efficiently the intersections required to compute the pairs with maximum interest. The expanded co-occurrence of a pair cw to be included in $CoOcc$ can be also efficiently calculated while detecting maximum interest pairs.

It can be noticed that the number of interesting co-occurrences decreases linearly as the parameter α increases, whereas the quality of the co-occurrences increases linearly with respect to α. In this way, the coverage of topics decreases proportionally with respect to the number of filtered co-occurrences. Therefore, contrary to classical data mining algorithms, our method is not affected by the bias produced by the Zipf law over the distribution of topics. Additionally, these results are comparable with those obtained in ad-hoc Topic Detection (TD) systems.Therefore, the quality of co-occurrences is acceptable for our analysis purposes, which is the focus of this paper.

We also have evaluated co-occurrences obtained with traditional data mining techniques. We applied the FP-growth algorithm [3] to the document term vectors in order to obtain the frequent term co-occurrences. Table 2 shows the obtained results of the FP-growth algorithm with different minimum support values. As it can be noticed, this algorithm only generates an acceptable num-

[2] http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/DecisionTreeTagger.html

[3] http://www.ldc.upenn.edu/Catalog/index.jsp

Table 1. Co-ocurrence quality measures

α	|CoOccs|	F1	Cosine	α	|CoOccs|	F1	Cosine
0,7	5865	0,620	0,420	0,5	8976	0,727	0,505
0,6	7416	0,689	0,446	0,3	12325	0,743	0,512

Table 2. FP-growth results

Minimum Support	Generated itemsets	% Covered Topics
60	3528717	14%
100	483719	9%
300	9071	3.4%

Table 3. Inferred Cubes for the TDT2 collection

Cubes	Domains	Support
(Planet, Earth)	astronomy, earth	58
(Territory, Region, Organization, Leader)	administ.#earth, economy, politics	58
(Territory, Substance, Creature)	administ.#earth, aliment., biology	54
(Territory, Healt_problem, Artifact)	administ.#earth, medicine	47
(Territory, Due_process_of_law, request)	administ.#earth, law	42
(Amount, Artifact)	economy, medicine	24

ber of co-occurrences with high support values, and leaving uncovered many topics. One critical issue of the dimension inference process is the precision of the disambiguation method applied to assigning terms to concepts. Methods relying on the correlation of word context with concept context have an approximate precision of 60% for nouns in the SENSEVAL2[4] collection [9]. However, in our scenario words considered useful for analysis purposes use to be restricted to some concrete domain and they have little ambiguity degree, therefore their disambiguation can be easier performed by using additional knowledge sources such as Magnini domain fields. Finally, as interesting co-occurrences contain semantically related words, we can deal with richer contexts (e.g. co-occurrence documents) that can help their disambiguation. In future work we will evaluate the actual precision of the disambiguation method and its effect to the cube construction.

A snippet of the specific taxonomy created by our method is shown in Figure 1. The coverage of this taxonomy is around 90% of the collection documents. Table 2 shows the most frequent cubes calculated by mining all the possible collection facts with the FP-growth algorithm. The whole set of dimensions can be found in the Web page http://tempus.dlsi.uji.es/MMTDT.html. In summary, for the TDT collection 107 dimensions were obtained. These dimensions contain 3551 distinct concepts, and they have an average depth of 2.4 levels. Focusing

[4] http://www.d.umn.edu/ tpederse/senseval2.html

only in those co-occurrences with a high F1 value with respect to the LDC topics, the number of concepts is reduced to 1270 and the average depth is of 3.2. Thus, many category dimensions are not directly associated to topics, and they are poorly defined according to the detail levels. Nevertheless, categories that are not associated to topics give to users new views that have not been taken into account by LDC annotators, and that could be useful for analysis tasks.

6 Conclusions

Building content dimensions from a large text collection can be very useful for analysis tasks relying on multidimensional models. In this work we propose a method that first identifies the interesting terms starting from the interesting co-occurrences where they participate, then assigns them to external taxonomy concepts, and finally it builds possible content dimensions to be used in multidimensional models. As experimental results show, the obtained interesting co-occurrences present a good quality when compared to actual collection topics. Moreover, inferred dimensions are meaningful for analysis purposes and their number is reasonable to be browsed by users. As future work, we are planning to introduce more restrictions over the inferred dimensions that simplify their structure (e.g. balanced and stratified categories). Furthermore, we are studying interesting measures for the proposed multidimensional models, such as the document relevance or document metadata. One immediate use of these measures is to create contexts for corporate cubes [8].

References

1. Bentivogli, L.; Forner P.; Magnini, B.; Pianta, E.: Revising WordNet Domains Hierarchy: Semantics, Coverage, and Balancing. In Proc. of COLING 2004 Workshop on Multilingual Linguistic Resources, 101–108, 2004.
2. Fellbaum, C. (editor):WordNet: An Electronic Lexical Database. Mit Press, 1998.
3. Han, J.; Pei, J.; Yin, Y.: Mining frequent patterns without candidate generation. In Proc. ACM SIGMOD Intl. Conference on Management of Data, 1–12, 2000.
4. Han, J.; Kamber, M.: Data Mining: Concepts and Techniques. Morgan Kaufmann, 2000.
5. Hurtado, C.A.; Mendelzon, A.O.: OLAP Dimension Constraints. In Proc. PODS 2002, 169–179, 2002.
6. Jarke, M.; Lenzerini, M.; Vassiliou, Y.; Vassiliadis, P.: Fundamentals of DataWarehouses. Springer-Verlang Ed. 2003.
7. Koller, D.; Sahami, M.: Hierarchically classifying documents using very few words. In Proc. of the 14th Conf. on Machine Learning ICML'97, 143–151, 1997.
8. Pérez, J.M.; Berlanga, R.; Aramburu, M.J.; Bach, T.: A relevance-extended multi-dimensional model for a data warehouse contextualized with documents. In Proc. of the 8th ACM International Workshop on Data Warehousing and OLAP. DOLAP'05. 19–28. 2005.
9. Pons, A.; Berlanga, R.; Ruíz-Shulcloper, J.: Un nuevo método de desambiguación del sentido de las palabras usando WordNet. In Proc. X Conf. de la Asociación para la Inteligencia Artificial CAEPIA 2003, 63–66, 2003.

10. Pons, A.: Desarrollo de algoritmos para la estructuración dinámica de información y su aplicación a la detección de sucesos. Tesis Doctoral, Depto. Lenguajes y Sistemas Informáticos, 2004.
11. Salton, G.: Automatic Text Processing. Addison-Wesley, 1989.
12. Webb, G.; Zhang, S.: K-Optimal Rule Discovery. Data Mining and Knowledge Discovery. Volume 10, Issue 1. 39–79. Jan. 2005.

Internal Categories with Irregular Geometry and Overlapping in ART Networks

Dinani Gomes, Manuel Fernández-Delgado, and Senén Barro

Intelligent Systems Group, Dept. of Electronics and Computer Science,
University of Santiago de Compostela, Santiago de Compostela,
15782, A Coruña, Spain
dinani@usc.es, {delga, senen}@dec.usc.es
http://www-gsi.dec.usc.es

Abstract. PolyTope ARTMAP (PTAM) [6] is an ART neural network based on internal categories with irregular polytope (polygon in $\mathbb{R}^n$) geometry. Categories in PTAM do not overlap, so that their expansion is limited by the other categories, and not by the category size. This makes the vigilance parameter unnecessary. What happens if categories have irregular geometries but overlap is allowed? This paper presents Overlapping PTAM (OPTAM), an alternative to PTAM based on polytope overlapping categories, which tries to answer this question. The comparison between the two approaches in classification tasks shows that category overlap does not reduce neither the classification error nor the number of categories, and it also requires vigilance as a tuning parameter. Futhermore, OPTAM provides a significant variability in the results among different data sets.

1 Introduction

ART neural networks [2] have been widely used in pattern recognition tasks which require on-line learning. ART networks have internal categories, which expand during training towards the input patterns. Each category covers a different region in the input space. The activation function of a category determines its response to an input pattern, which is higher for patterns laying inside the regions covered by the category. In the most popular ART nets (Fuzzy ARTMAP [3] and Distributed ARTMAP [4]), the internal categories have hyperbox geometry. During the training stage, the internal categories associated to an output prediction expand in order to cover the regions of the input space populated by patterns with that output prediction. Since these regions may not be hyperboxes, a high number of categories may be required to cover them. Recently, some ART networks have been proposed based on categories with hyperellipsoidal geometries [1], as well as based on irregular geometries [5]. Specifically, PolyTope ARTMAP (PTAM) [6] uses irregular polytope-shaped internal categories, in order to increase its ability to approximate regions with variable geometries. Another feature of PTAM is the absence of category overlap, as opposite to the classical ART networks. During training, the category expansion is

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 291–300, 2006.

limited by the other categories, because they can not overlap. So, PTAM does not use the vigilance parameter, due the fact that the categories do not need an upper limit on its size, given by the vigilance, to limit its expansion.

The absence of the vigilance parameter allows PTAM to work in a fully automatic way, without any parameter tuning. However, it is interesting to evaluate how useful are polytope categories when overlap is allowed. This alternative version of PTAM, which we call Overlapping Polytope ARTMAP (OPTAM) is described in Section 2. Section 3 compiles the results of OPTAM and PTAM, including other ART networks, on several benchmark tasks, and Section 4 compares the results of both approaches. Finally, Section 5 summarizes the main conclusions and future work.

2 Overlapping PolyTope ARTMAP (OPTAM)

As the direct mathematical description of irregular polytopes is complex, we describe each polytope category C_i ($i = 1, \ldots, N_c$, where N_c is the number of categories) as a set of N_i^s adjacent simplexes, $S_{\xi(i,1)}, \ldots, S_{\xi(i,N_i^s)}$ ($\xi(i,j)$ is the index of the j-th simplex of C_i). Each simplex $S_{\xi(i,j)}$ is defined by $n+1$ weight vectors or vertices $\mathbf{w}_{\nu(i,j,1)}, \ldots, \mathbf{w}_{\nu(i,j,n+1)}$ ($\nu(i,j,k)$ is the index of the k-th vertex in simplex $S_{\xi(i,j)}$) and it is delimited by $n+1$ hyperplanes $h_{\zeta(i,j,1)}, \ldots, h_{\zeta(i,j,n+1)}$ ($\zeta(i,j,k)$ is the index of k-th hyperplane of simplex $S_{\xi(i,j)}$). Each hyperplane $h_{\zeta(i,j,k)}$ is also defined by n vectors $\mathbf{w}_{\pi(i,j,k,1)}, \ldots, \mathbf{w}_{\pi(i,j,k,n)}$. In some situations, a category C_i may be defined by only one vector (single-vector category), denoted by $\mathbf{w}_{\eta(i)}$, and zero simplexes ($N_i^s = 0$). The following subsections describe the training stage of OPTAM.

2.1 Category Activation Function

Given a training pattern $\mathbf{I}$, we define the activation function $T_i^t(\mathbf{I})$ for a polytope category C_i during the training stage as $T_i^t(\mathbf{I}) = \chi_i^t(\mathbf{I})$. Function $\chi_i^t(\mathbf{I})$ satisfies $\chi_i^t(\mathbf{I}) = 1$ if $\mathbf{I}$ falls inside the region covered by C_i and $\chi_i^t(\mathbf{I}) < 1$ otherwise. In this case, $\chi_i^t(\mathbf{I})$ is a decreasing function with the distance between $\mathbf{I}$ and C_i. We define $\chi_i^t(\mathbf{I})$ as:

$$\chi_i^t(\mathbf{I}) = \max_{j=1,\ldots,N_i^s} \{\chi_{ij}(\mathbf{I})\} \qquad i = 1, \ldots, N_c \ . \tag{1}$$

where $\chi_{ij}(\mathbf{I})$ is the activation function of simplex $S_{\xi(i,j)}$, so that $\chi_i^t(\mathbf{I})$ is the highest activation of the simplexes in C_i. The simplex activation function $\chi_{ij}(\mathbf{I})$ is defined in the following way:

$$\chi_{ij}(\mathbf{I}) = \begin{cases} 1 & g_{ijk}(\mathbf{I}) > 0, \quad k = 1, \ldots, n+1 \\ e^{-d(\mathbf{I}, S_{\xi(i,j)})/\gamma} & \text{otherwise} \end{cases} \ . \tag{2}$$

Function $g_{ijk}(\mathbf{I})$ is defined so that $g_{ijk}(\mathbf{I}) = 0$ if $\mathbf{I} \in h_{\zeta(i,j,k)}$, $g_{ijk}(\mathbf{I}) < 0$ if $\mathbf{I}$ is in the side of the hyperplane $h_{\zeta(i,j,k)}$ outside the simplex $S_{\xi(i,j)}$ and $g_{ijk}(\mathbf{I}) > 0$ otherwise. Concisely:

$$g_{ijk}(\mathbf{I}) = sgn(\phi_{ijk}(\mathbf{w}^*_{\varsigma(i,j,k)}))\phi_{ijk}(\mathbf{I}) \ . \tag{3}$$

$$\phi_{ijk}(\mathbf{x}) = \begin{vmatrix} x_1 - w_{\pi(i,j,k,1),1} & \dots & x_n - w_{\pi(i,j,k,1),n} \\ w_{\pi(i,j,k,2),1} - w_{\pi(i,j,k,1),1} & \dots & w_{\pi(i,j,k,2),n} - w_{\pi(i,j,k,1),n} \\ \dots & & \\ w_{\pi(i,j,k,n),1} - w_{\pi(i,j,k,1),1} & \dots & w_{\pi(i,j,k,n),n} - w_{\pi(i,j,k,1),n} \end{vmatrix} . \quad (4)$$

where $sgn(x) = 1$ if $x > 0$, $sgn(x) = -1$ if $x < 0$ and $sgn(0) = 0$. We denote by $w_{\pi(i,j,k,l),m}$ the m-th component of the vector $\mathbf{w}_{\pi(i,j,k,l)}$. If $\mathbf{I} \in h_{\zeta(i,j,k)}$, $\mathbf{I}$ is a linear combination of the vectors $\mathbf{w}_{\pi(i,j,k,2)} - \mathbf{w}_{\pi(i,j,k,1)}, \dots, \mathbf{w}_{\pi(i,j,k,n)} - \mathbf{w}_{\pi(i,j,k,1)}$, so that $\phi_{ijk}(\mathbf{I}) = 0$. On the other hand, $\mathbf{w}^*_{\varsigma(i,j,k)}$ is the vertex in the simplex $S_{\xi(i,j)}$ that does not belong to $h_{\zeta(i,j,k)}$, so that $g_{ijk}(\mathbf{I}) > 0$ if $\mathbf{I}$ falls in the side of $h_{\zeta(i,j,k)}$ inside the simplex. If $g_{ijk}(\mathbf{I}) > 0, \forall k = 1, \dots, n+1$, then $\mathbf{I}$ falls inside the simplex $S_{\xi(i,j)}$ and $\chi_{ij}(\mathbf{I}) = 1$ (upper side in Eq. 2).

In the lower side of Eq. 2, which applies if $\mathbf{I}$ falls outside $S_{\xi(i,j)}$, $\gamma = \sqrt{n}/10$ is a factor that scales $\chi_{ij}(\mathbf{I})$ in the range $0 \simeq e^{-10} \leq \chi_{ij}(\mathbf{I}) \leq 1$, and $d(\mathbf{I}, S_{\xi(i,j)})$ is the distance between $\mathbf{I}$ and $S_{\xi(i,j)}$ (note that $0 \leq d(\mathbf{I}, S_{\xi(i,j)}) \leq \sqrt{n}$). Its exact calculation in $\mathbb{R}^n$ is complex, so we have used an approximated value. Let us consider the hypersphere with center $\mathbf{c}_{ij} = \sum_{l=1}^{n+1} \mathbf{w}_{\nu(i,j,l)}/(n+1)$, which is the centroid of the vertices in the simplex $S_{\xi(i,j)}$, whose radius R_{ij} is the maximum distance between $\mathbf{c}_{ij}$ and a vertex in $S_{\xi(i,j)}$: $R_{ij} = max_l\{\|\mathbf{c}_{ij} - \mathbf{w}_{\nu(i,j,l)}\|\}_{l=1}^{n+1}$. If $\mathbf{I}$ falls inside this hypersphere, we approximate $d(\mathbf{I}, S_{\xi(i,j)})$ using the distance between $\mathbf{I}$ and the hyperplane $h_{\zeta(i,j,k)} \in S_{\xi(i,j)}$ which is the closest to $\mathbf{I}$ and satisfies $g_{ijk}(\mathbf{I}) < 0$, i.e., $\mathbf{I}$ falls in the side of $h_{\zeta(i,j,k)}$ outside $S_{\xi(i,j)}$. Otherwise, $d(\mathbf{I}, S_{\xi(i,j)})$ is approximated using the distance between $\mathbf{I}$ and the closest to $\mathbf{I}$ vertex in $S_{\xi(i,j)}$:

$$d(\mathbf{I}, S_{\xi(i,j)}) \simeq \begin{cases} \min\limits_{k:g_{ijk}(\mathbf{I})<0} \dfrac{|g_{ijk}(\mathbf{I})|}{\|\mathbf{w}^*_{ijk}\|} & \|\mathbf{I} - \mathbf{c}_{ij}\| < R_{ij} \\ \min\limits_{l=1,\dots,n+1} \left\{\|\mathbf{I} - \mathbf{w}_{\nu(i,j,l)}\|\right\} & \|\mathbf{I} - \mathbf{c}_{ij}\| \geq R_{ij} \end{cases} . \quad (5)$$

In this equation, $\|\mathbf{w}^*_{ijk}\|$ is the norm of the direction vector of the hyperplane $h_{\zeta(i,j,k)}$, which can be calculated using its vertices $\mathbf{w}_{\pi(i,j,k,1)}, \dots, \mathbf{w}_{\pi(i,j,k,n)}$.

2.2 Category Expansion

In OPTAM, similarly to the classical "winner-takes-all" ART networks, the winner of the competition among internal categories is the one with the highest activation: $T^t_I(\mathbf{I}) = max_i\{T^t_i(\mathbf{I})\}_{i=1}^{N_c}$. The winner category C_I tries to expand towards $\mathbf{I}$, either creating a new simplex between them, or replacing a vertex in C_I by $\mathbf{I}$. The creation of a new simplex requires n vectors $\{\mathbf{w}_p\}_{p=1}^n \in C_I$ for which the segments $\{\overrightarrow{\mathbf{I}\mathbf{w}_p}\}_{p=1}^n$ do not overlap with C_I (i.e., they are connectables with $\mathbf{I}$). In order to select these vectors, the intersection between $\overrightarrow{\mathbf{I}\mathbf{w}_p}$ and the hyperplanes of C_I is tested solving the corresponding systems of linear equations. Depending on the number of connectable vectors there are several cases (Fig. 1). If $T_I(\mathbf{I}) = 1$, C_I already covers $\mathbf{I}$, so C_I does not need to expand towards $\mathbf{I}$ (panel 1). If $T_I(\mathbf{I}) < 1$, C_I tries to expand towards $\mathbf{I}$. If there are n vertices in C_I connectables with $\mathbf{I}$, C_I expands towards $\mathbf{I}$ creating a new simplex (panel 2a) with the n connectable vertices and $\mathbf{I}$. If

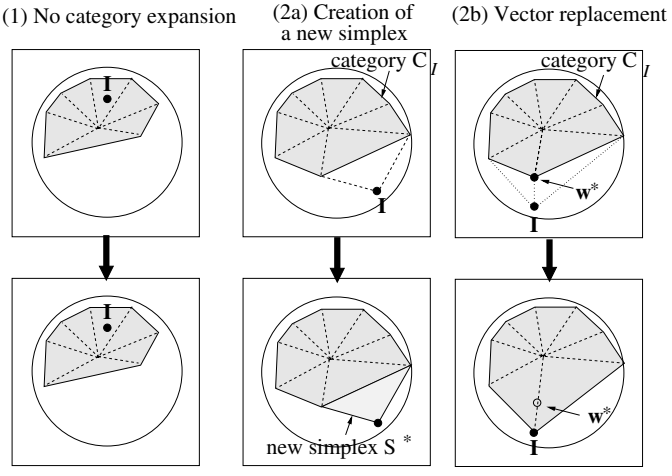


Fig. 1. Polytope category learning examples for the data set "Circle-In-the-Square" (CIS) in $\mathbb{R}^2$. See text for details.

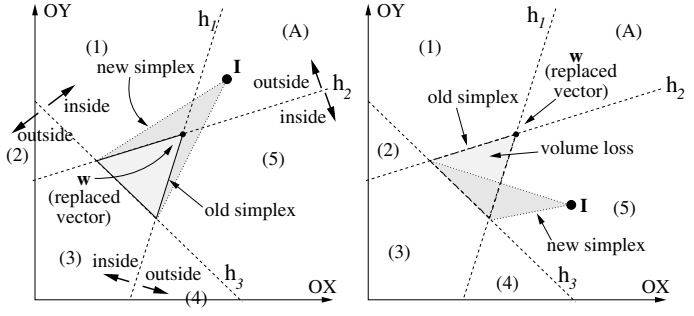


Fig. 2. Example of replacement of vector $\mathbf{w}$ in $\mathbb{R}^2$. Only patterns located in region A can replace $\mathbf{w}$ (left panel). The replacement of $\mathbf{w}$ by patterns situated in the remaining regions would reduce the simplex volume (right panel).

there are more than n vertices in C_I connectables with $\mathbf{I}$, then C_I expands towards $\mathbf{I}$ replacing some vertex in C_I by $\mathbf{I}$ (panel 2b).

The replacement of a vector $\mathbf{w}$ in C_I by $\mathbf{I}$ can reduce the volume of the simplexes to which $\mathbf{w}$ belongs (Fig. 2, right panel). In particular, the patterns that can replace $\mathbf{w}$ in Fig. 2 are those located in region A, which is outside the hyperplanes (lines in $\mathbb{R}^2$) that contain $\mathbf{w}$. Therefore, the vectors $\mathbf{w}$ that can be replaced by $\mathbf{I}$ must verify that $g_{ijk}(\mathbf{I}) < 0, \forall k : \mathbf{w} \in h_{\zeta(i,j,k)}$. If there are several vectors that fulfill this condition, $\mathbf{I}$ replaces the closest one.

2.3 Vigilance Test

In the ART networks, the Vigilance Test determines if the winner category C_I can expand towards $\mathbf{I}$. The vigilance parameter ρ determines the maximum cat-

egory size. The size S_i of a polytope category C_i in OPTAM is defined similarly to Fuzzy ARTMAP:

$$S_i = \sum_{l=1}^{n} M_{il} - m_{il}; \qquad M_{il} = \max_{j=1,\ldots,N_i^s} \left\{ \max_{k=1,\ldots,n+1} \left\{ \mathbf{w}_{\nu(i,j,k)l} \right\} \right\} \tag{6}$$

$$m_{il} = \min_{j=1,\ldots,N_i^s} \left\{ \min_{k=1,\ldots,n+1} \left\{ \mathbf{w}_{\nu(i,j,k)l} \right\} \right\} .$$

Note that $0 \leq S_i \leq n$. The maximum category size S_{max} allowed in OPTAM is defined as $S_{max} = (1-\rho)n$. If the winner category C_I satisfies $S_I \geq S_{max}$ after expansion towards $\mathbf{I}$, C_I is reset (Vigilance Test). Then, a new winner category is selected in the competition, and the category expansion (Subsection 2.2) is repeated again.

2.4 Prediction Test

If the active category C_I passes the Vigilance Test ($S_I < S_{max}$), its associated prediction $P(C_I)$ is compared with the desired prediction P_d for the input pattern (Prediction Test). If $P_d = P(C_I)$, the winner category C_I passes the Prediction Test and it codifies the input pattern (Resonance). Otherwise, C_I is reset and a new winner category is selected in the competition. OPTAM does not use *Match-Tracking* [3], usual in ART networks, because it reduces the maximum category size and, consequently, its ability of expansion, and it increases the classification error and the number of categories created.

2.5 Creation of New Categories

In the classical ART networks, if no category passes the Vigilance Test, a new category is created, defined by the input pattern. The categories of OPTAM are defined by $(n+1) + (N_i^s - 1) = n + N_i^s$ weight vectors (vertices). Therefore, the creation of a new category C_{N_c+1} defined by just one simplex S^* requires at least n weight vectors (vertices), belonging to single-vector categories, with the same prediction as the input pattern. If there are n vectors of this kind, OPTAM creates a new single-simplex category C_{N_c+1} with just the simplex S^* ($N_{N_c+1}^s = 1$ and $S_{\xi(N_c+1,1)} = S^*$). If there are not n vectors of this kind, OPTAM creates a new single-vector category C_{N_c+1} with zero simplexes ($N_{N_c+1}^s = 0$) and with just one vector equal to the input pattern $\mathbf{w}_{\eta(N_c+1)} = \mathbf{I}$.

2.6 Processing

During the processing stage, the output of OPTAM for an input pattern is the prediction associated to the category with the highest activation function $T_i^p(\mathbf{I})$. Similarly to the training stage, we define the function $\chi_i^p(\mathbf{I})$ of category C_i during the processing stage as:

$$\chi_i^p(\mathbf{I}) = \begin{cases} \max\limits_{j=1,\ldots,N_i^s} \{\chi_{ij}(\mathbf{I})\} & N_i^s \geq 1 \\ e^{-\|\mathbf{I}-\mathbf{w}_{\eta(i)}\|/\gamma} & N_i^s = 0 \end{cases} . \tag{7}$$

where $\chi_{ij}(\mathbf{I})$ is defined in Eq. 2. In the Eq. 7, $\chi_i^p(\mathbf{I})$ is equal to $\chi_i^t(\mathbf{I})$ except for single-vector categories, for which $\chi_i^p(\mathbf{I})$ is decreasing with the distance between $\mathbf{I}$ and $\mathbf{w}_{\eta(i)}$. Similarly to Fuzzy ARTMAP, when the input pattern falls inside a region covered by several overlapping categories, for which $\chi_i^p(\mathbf{I}) = 1$, the category with the lowest size S_i has the highest activation function $T_i^p(\mathbf{I})$. Using the function $\Theta(x) = 1$ if $x \geq 0$ and $\Theta(x) = 0$ if $x < 0$, we define the category activation function $T_i^p(\mathbf{I})$ in the processing stage by:

$$T_i^p(\mathbf{I}) = [1 - \psi(\mathbf{I})]\chi_i^p(\mathbf{I}) + \psi(\mathbf{I})\frac{\Theta[\chi_i^p(\mathbf{I}) - 1]}{\alpha + S_i} \tag{8}$$

$$\psi(\mathbf{I}) = \Theta\left[\sum_{l=1}^{N_c} \Theta[\chi_l^p(\mathbf{I}) - 1] - 2\right] . \tag{9}$$

where $\alpha \gtrsim 0$ and S_i is defined in Eq. 6. Function $\psi(\mathbf{I})$ will be $\psi(\mathbf{I}) = 1$ if several categories exist with $\chi_i^p(\mathbf{I}) = 1$. In such a case, $T_i^p(\mathbf{I}) = 1/(\alpha + S_i)$ for the categories with $\chi_i^p(\mathbf{I}) = 1$, and $T_i^p(\mathbf{I}) = 0$ for the remaining ones, so that $T_i^p(\mathbf{I})$ is decreasing with the category size S_i. If there is only one category with $\chi_i^p(\mathbf{I}) = 1$, then $\psi(\mathbf{I}) = 0$ and $T_i^p(\mathbf{I}) = \chi_i^p(\mathbf{I})$. The winner category C_I is that with the highest $T_i^p(\mathbf{I})$:

$$T_I^p(\mathbf{I}) = \max_{i=1,\ldots,N_c} \{T_i^p(\mathbf{I})\} . \tag{10}$$

The output of OPTAM is the prediction $P(C_I)$ associated to the winner category C_I.

3 Results

Experiments were developed comparing OPTAM with PTAM [6] and with some of the most popular ART networks: Distributed ARTMAP (DAM) [4] (rectangular categories) and Gaussian ARTMAP (GAM) [11] (hyperellipsoidal categories). We have selected these networks because they achieve the best results in our simulations with data sets of both geometries (we have also tried with Fuzzy ARTMAP [3], Ellipsoid ARTMAP [1] and FasArt [8], achieving worse results, not included in the tables). We have also compared OPTAM with SVM [9], because it is a reference network for classification tasks. We have used two data sets with circular geometry ("Circle-in-the-Square" (CIS) [10] and T5 [8]) and two data sets with rectangular geometry (Chess [7] and T4 [8]), in order to evaluate each network in data sets with the same and different geometries (Fig. 3).

DAM, GAM and OPTAM require the optimization of the vigilance ρ for each data set. So, we developped cross-validation trials using 20 training, 20 validation and 20 test sets. Each set contains 10,000 randomly generated patterns with equiprobable predictions. We trained a version of each network in each training set using vigilance values in the range $0.00 : 0.05 : 0.95$. We proved each version in its validation set, and selected the vigilance value with the best tradeoff between average classification error and number of categories over the 20 validation sets.

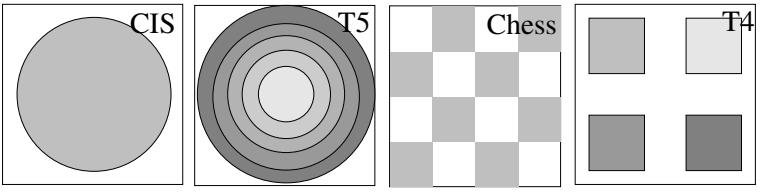


Fig. 3. Data sets CIS, T5 (circular) and Chess, T4 (rectangular), with 2, 6, 2 and 5 output predictions respectively

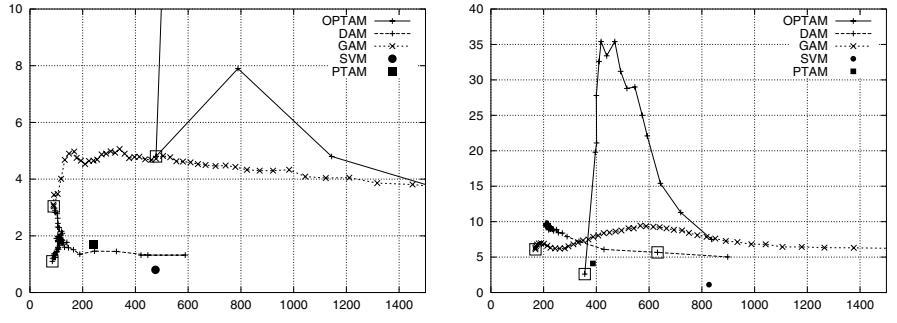


Fig. 4. Average classification error against the number of categories over the validation sets for data sets Chess (left panel) and T5 (right panel) varying vigilance. The best tradeoff between error and categories is marked with an empty square for each network.

Table 1. Test error and number of internal categories (vectors for OPTAM-PTAM and support vectors for SVM) for each network and data set, and average values (the best values are in bold)

	CIS		Chess		T4		T5		Average	
	Error	No. Cat	Error	No. Cat	Error	No. Cat	Error	No. Cat	Error	No. Cat
OPTAM	**0.6**	119	4.8	478	0.9	170	**2.6**	356	2.2	281
PTAM	1.0	102	1.7	242	1.0	158	4.1	387	**2.0**	222
GAM	1.3	**43**	3.0	90	1.9	89	6.1	**169**	3.1	**98**
DAM	1.3	238	**1.1**	**89**	**0.4**	**48**	5.6	630	2.1	251
SVM	**0.2**	578	**0.8**	475	0.6	271	**1.1**	827	**0.7**	538

Fig. 4 shows the classification error against the number of categories, averaged over the validation sets, for OPTAM, DAM, GAM, SVM and PTAM, varying the vigilance in the data sets Chess (left panel) and T5 (right panel). Finally, we evaluated each network, trained with its best value of ρ, on the 20 test sets. Table 1 shows the average results over the 20 test sets. OPTAM uses $\rho = 0$ for data sets CIS, T4 and T5, and $\rho = 0.75$ for Chess.

4 Discussion

SVM achieves significantly lower errors than the ART networks in all the data sets (Table 1), although they create less categories than support vectors in the SVM. The behavior of ART networks varies among data sets. In the rectangular ones (Chess and T4), DAM achieves the lowest error (1.1% and 0.4%, respectively), which is lower than SVM in T4. OPTAM achieves low error (0.9%) in T4 and a high error (4.8%) in Chess, whereas PTAM achieves intermediate errors (1.7% and 1.0%). DAM and GAM create less categories, although GAM makes the highest error in both data sets. In circular data sets CIS and T5, OPTAM achieves the smallest error (0.6% and 2.6%, respectively), followed by PTAM, with less error than DAM and GAM. GAM provides the highest errors (1.3% and 6.1%), in spite of its circular geometry.

On average, PTAM achieves the lowest classification error (2.0%), slightly lower than DAM and OPTAM (2.1% and 2.2% respectively), and clearly lower than GAM (3.1%)[1]. On the other hand, GAM achieves the lowest number of categories (98), but with the highest error, while PTAM creates less categories (222) than DAM (251) and OPTAM (281). Therefore, the use of overlapping categories in OPTAM does not reduce the error with respect to PTAM or to DAM and, in addition, it increases the number of categories compared with these two networks.

It is interesting to analyse the behavior of OPTAM in the data set Chess. Using $\rho = 0$, $S_{max} = n$ (Subsection 2.3) and OPTAM creates only two overlapping categories, that cover all the input space. Since the two predictions are distributed among several separated regions, these two categories are not able to discriminate among these regions, and the error is very high ($\sim$50%, left end of Fig. 4, left panel). Therefore, OPTAM needs a high vigilance value ($\rho = 0.75$) to achieve an error comparable to PTAM, DAM and GAM, and it creates a high number of categories (478). Nevertheless, when a pattern falls in a region of category overlap during the processing stage, the smallest category is not necessarily the right one, so that OPTAM achieves a high error (4.8%) compared with the other networks, even with high vigilance values (right end of Fig. 4, left panel). However, in the data sets CIS, T4 and T5 the value $\rho = 0$ provides the smallest error and number of categories, because each prediction has only one associated region, which can be correctly covered by only one category using $\rho = 0$. For example, in T5 the predictions are concentric, so the categories created by OPTAM for the different predictions overlap. If a pattern falls inside an overlap region during the processing stage, it is assigned to the smallest category, which is the right one in T5. This behavior can also be observed in CIS and T4.

This variability in the results, which depend on the number of regions associated to each prediction, shows that the behavior of OPTAM is not very stable. In fact, the error of PTAM varies less among data sets, since its standard

[1] We have made also additional experiments with OPTAM using *Match-Tracking*, and we achieved higher average error (4.2%) and number of categories (384), results that justify their absence in Table 1.

deviation is the lowest among the ART networks (1.5 against 1.9 for OPTAM, 2.1 for DAM and 2.4 for GAM). These results suggest that polytope categories are more effective– both in error and number of categories– if they can not overlap. In addition, for an accurate learning of data sets with complex geometries, it turns out contradictory to use irregular polytope categories and, on the other hand, to give more relevance– as in OPTAM– to the category size compared with the category geometry when patterns fall inside several overlapping categories.

5 Conclusions

In this work we have presented an ART network based on categories with polytope irregular geometric shape. In a preliminary work [6] we proposed the model PolyTope ARTMAP (PTAM), based on non-overlapping polytope categories. The expansion of a PTAM category is limited by the other categories during training, so they do not need a vigilance parameter. In the present work we compare PTAM with an alternative approach, called Overlapping PolyTope ARTMAP (OPTAM), that allows category overlap. The results conclude that category overlap does not reduce the error significantly and it increases the number of categories created. Furthermore, it raises the variability in the results among different data sets, which strongly depend on the number of regions associated to each output prediction. On the other hand, the selection of the vigilance value requires to make cross-validation trials which raise the computational cost. Although in simple data sets OPTAM achieves good results with $\rho = 0$, it is essential to tune the vigilance in more complex data sets– case of Chess, whose predictions are mixed– and this does not reduce error or categories compared with PTAM.

Future work will improve the category expansion in PTAM, which must be more efficient in order to cover the input space. Specifically, we are working to define the polytope categories by means of hyperplanes, instead of simplexes, in order to make expansion more flexible and efficient. Another line of future work is to use distributed learning in PTAM, as a solution to the category proliferation in the ART networks with competitive *winner-takes-all* learning.

Acknowledgments

The authors wish to thank the Supercomputing Center of Galicia (CESGA), for allowing to use their computational resources. This work was supported by the Spanish CICyT and the Xunta de Galicia under projects TIC2003-09400-C04-03 and PGIDIT04SIN206003PR, respectively.

References

1. Anagnostopoulos, G., Georgiopoulos M.: Ellipsoid ART and Ellipsoid ARTMAP for incremental clustering and classification. IEEE Int. Joint Conf. on Neural Networks. Vol. 2, 2001, pp. 1221–1226.

2. Carpenter, G., Grossberg, S.: The ART of adaptive pattern recognition by a self-organizing neural network. Computer 21 (3), 1988, pp. 77–88.
3. Carpenter, G., Grossberg, S., Markuzon, N., Reynolds, J., Rosen, D.: Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps. IEEE Transactions on Neural Networks, 3 (5), 1992, pp. 698–713.
4. Carpenter, G., Milenova, Noeske, B.: Distributed ARTMAP: a neural network for fast distributed supervised learning. Neural Networks, 11 (5), 1998, pp. 793–813.
5. Gomes, D., Fernández-Delgado, M., Barro, S.: Simplex ARTMAP: building general geometry borders among predictions with simplex-shaped classes. IASTED Int. Conf. on Artificial Intelligence and Soft Computing. Vol. 1, 2004, pp. 232–237.
6. Gomes, D., Fernández-Delgado, M., Barro, S.: A vigilance-free ART network with general geometry internal categories. 2005 IEEE Int. Joint Conf. on Neural Networks, 2005, pp. 463–468.
7. Gómez-Sánchez, E., Dimitriadis, Y., Cano-Izquierdo, J., Coronado, J.: MicroARTMAP: use of mutual information for category reduction in Fuzzy ARTMAP. IEEE Int. Joint Conf. on Neural Networks, 2000, pp. 647–652.
8. Parrado-Hernández, E., Gómez-Sánchez, E., Dimitriadis, Y.: Study of distributed learning as a solution to category proliferation in Fuzzy ARTMAP based neural systems. Neural Networks, 16 (7), 2003, pp. 1039–1057.
9. Vapnik, V.: Statistical Learning Theory. Wiley, 1998.
10. Wilensky, G.: Analysis of neural network issues: scaling, enhanced nodal processing, comparison with standard classification. DARPA Neural Network Program Review, 1990.
11. Williamson, J.: Gaussian ARTMAP: A neural network for fast incremental learning of noisy multidimensional maps. Neural Networks, 9 (5), 1996, pp. 881–897.

Legal Ontologies for the Spanish e-Government

Asunción Gómez-Pérez, Fernando Ortiz-Rodríguez, and Boris Villazón-Terrazas

Ontology Engineering Group, Laboratorio de Inteligencia Artificial. Facultad de Informática.
Universidad Politécnica de Madrid. Spain
{asun, fortiz, bvillazon}@fi.upm.es

Abstract. The Electronic Government is a new field of applications for the semantic web where ontologies are becoming an important research technology. The e-Government faces considerable challenges to achieve interoperability given the semantic differences of interpretation, complexity and width of scope. In this paper we present the results obtained in an ongoing project commissioned by the Spanish government that seeks strategies for the e-Government to reduce the problems encountered when delivering services to citizens. We also introduce an e-Government ontology model; within this model a set of legal ontologies are devoted to representing the Real-estate transaction domain used to illustrate this paper.

1 Introduction and Motivation

Electronic Government (e-Gov) is an important application field[3] for the transformations that governments and public administrations will have to undergo in the next decades. Therefore, to transform the e-Gov into the e-Governance, the e-Gov research needs to be based on a robust theory, on modelling approaches, and on planning. In this scenario, a crucial issue is to manage in different ways the legal knowledge to improve the systems applications.

For more than two decades, the AI and Law community has been very active and productive. In the early 80's, research was focused on logic programming, and all the efforts were centered on legislation and legal reasoning. Other approach adopted was the case-based reasoning, which was not as formal as logic programming was, that aimed at finding similarities in legal cases and allowed retrieving relevant cases for the judges. Knowledge Engineering was also of interest for the research community and the field most applied since it allowed developing and using the legal ontologies that underlie the growth of the Semantic Web.

The Semantic Web was proposed by Tim Berners-Lee[8] as a new field of research, and according to the World Wide Web Consortium[1] (W3C) the Semantic Web is defined as "an extension of the current Web in which information is given well-defined meaning, better enabling computers and people to work in cooperation. It is based on the idea of having data on the Web defined and linked such that it can be used for more effective discovery, automation, integration, and reuse across various applications".

The application of the Semantic Web to the e-Gov domain is completely new; it features knowledge representation, knowledge engineering, database design, information

[1] http://www.w3.org/2001/sw

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 301–310, 2006.

systems, database integration, natural language understanding, information retrieval and semantic portals, among others. The Semantic Web is considered to be the infrastructure upon which all intelligent e-Gov applications will be built in the near future. Within the objectives of the Semantic Web the ontologies play an important role.

In the field of the Artificial Intelligence, Neches[12] was the first to define an ontology, and he did it as follows: "Ontology defines the basic terms and the relations that include the vocabulary of a specific area, in addition to the rules to combine terms and relations to define extensions to the vocabulary". Gruber[6,7] defines the ontology as: "An explicit specification of a conceptualization", being this definition the most referenced in the literature. Borst[1] slightly modify Gruber's definition saying that: "Ontologies are defined as a formal specification of a shared conceptualization". These last two definitions have been merged and explained by Studer and colleagues[15] as follows: "An ontology is a formal, explicit specification of a shared conceptualization. Conceptualization refers to an abstract model of some phenomenon. Explicit means that the type of concepts used, and the constraints on their use are explicitly defined. Formal refers to the fact that the ontology should be machine-readable. Shared reflects the notion that an ontology captures consensual knowledge, that is, it is not private of some individual, but accepted by a group".

The e-Gov has been strengthened with all these previous studies carried out by the research community and now its main concern is data representation and information management. By its nature, the e-Gov is supported by the legal domain. In Spain, legal ontologies for e-Gov applications have been scarce and to reverse this is the first goal of our paper. The second is to build ontologies that help reduce some important semantic problems presented when providing e-Gov services[4].

This research is based on the needs stated in a Spanish Project that seeks strategies for e-Gov, In this case, the citizens have relations with another citizens by the own wills reflected in contracts or agreements. The contracts are presented, normally, in documentation The fulfilment of these relations are guaranteed by the State or Governments by the Laws, reglaments and judicial decisions. One of the main goals of this project is, to provide knowledge conceptualizations (given by legal experts) that help improve information retrieval of legal sources in general.

This paper is organized as follows: section 2 deals with the related work carried out; section 3 presents the Reimdoc Project; section 4 describes the legal ontologies already built. And finally, section 5 is devoted to the conclusions.

2 Related Work

Nowadays the joint efforts put in by different research communities have made possible the birth of the semantic e-Gov. Since e-Gov ontologies are still in their initial state, only a few works carried out in this field are known; thus, in this section we provide a brief state of the art of those works performed in AI, in the law field and in the Semantic Web. The sum up of all these efforts will produce robust ontologies for the e-Gov domain in the near future.

2.1 Law and e-Gov within the Semantic Web

Currently, the Semantic Web is a new area of research and applications within the legal system and e-Gov domains and is a promise for the Web of the next generation; this new area, which is now used mainly to communicate with people but not with machines, will transform the current web since the capability of communication with machines is one of the main objectives of the Semantic Web. If the Web were equipped with more meaning, every citizen would extract answers in a new, easy and simple way and this action could be carried out by web powered semantics, what would enable citizens and businesses to obtain better information from the government. Web powered semantics could help the e-Gov in two ways: first, by allowing the government to delegate more intelligent tasks to computers and second, by solving daily problems with logic deductions and reasoning. But at present, the web is merely a common framework that allows data to be shared and reused.

Currently the legal and e-Gov Semantic Web applications are still in an experimental phase, but their potential impact on social, economical and political issues is extremely significant.

The main goals of e-Gov are to develop user-friendly and efficient services for the public and the business community, though semantic interoperability is also seen as an important issue to solve within this domain. Some of the works aimed at covering the semantic e-Gov domain are the following: the DIP project[2], the IFIP Working Group 8.5[3], the Ontogov project[4], the Egov project[5], and the WEBOCRAT project[6].

2.2 Ontologies: Domain Considerations

The e-Gov scenario is a promising application field for the ontologies underlying the legal engineered knowledge. Many ontologies have been built in the legal domain but not all of them are available or modelled just for a specific domain. The research efforts made in the legal domain by the AI community have contributed to the making of ontologies such as: LLD[10], NORMA[13,14], FOL[16], FBO[9,18] and LRI-Core Legal Ontology[2].

The emergence of legal ontologies as part of the Semantic Web initiative has provided a new opportunity for the research community and has brought about a solution to retrieve legal documents within the e-Gov domain. We can mention some of the efforts carried out by AI community on building e-Gov ontologies:

- The Government R&D[7] describes organizations and individuals participating in a government R&D program.

[2] http://dip.semanticweb.org
[3] http://falcon.ifs.uni-linz.ac.at/research/ifip85.html#aim
[4] http://www.ontogov.com/
[5] http://www.egov-project.org
[6] http://www.webocrat.org/
[7] http://www.daml.org/projects/integration/projects-20010811

– The Government type[8] describes government concepts used in the CIA World Fact Book 2002.
– The E-Government Ontology[9] describes a seamless UK taxonomy.

3 Reimdoc Project

We use a reference model to focus on and build a common understanding of the problem stated; Figure 1 shows the different actors within the e-Gov.

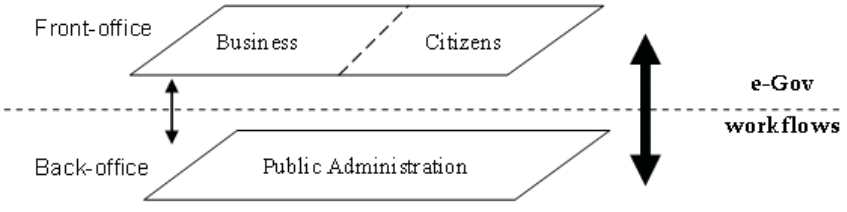


Fig. 1. The e-Government Reference Model

At the Back-office, the main actor is the Public Administration; it has many processes inside which should work properly to provide efficient services. The dynamics of the Public Administration provides a huge amount of information to be processed and these data should be managed in a transparent and efficient way.

Within the Public Administration many processes take place and these must be carried out properly to provide efficient services; since the Public Administration functions in a decentralized way and the dynamics of this field generates a huge amount of information to be processed, it is necessary to manage this vast amount of information in a transparent and efficient way. Therefore, the implementation of e-Gov ontologies and applications is crucial.

A particular case is being developed within the Reimdoc Project[10]. This project aims to develop tools that allow the legal document to be modelled in electronic support and be semantically retrieved to facilitate the government-citizen document transaction. The domain selected is related to the Real-estate transaction market and offers sufficient juridical guarantees.

This project will permit verifying the Real-estate processes gathered in digital support. These processes consist of procedures that occur in three areas: the Property Title, the Tributary Administration of the Autonomous Communities and the Justice Administration. In Spain these procedures are meticulously regulated in a coherent form by the context, which is marked by the legal knowledgeable community. Reimdoc Project is currently developing an application based on the proposed Legal Ontologies described

[8] http://reliant.teknowledge.com/DAML/Government.owl
[9] http://dip.semanticweb.org/documents/D9.3e-Governmentontology.doc
[10] http://reimdoc.atosorigin.es/

in section 4: EgoIR, an Information Retrieval system. EgoIR is a java-based system that offers an ontology-based approach to Information Retrieval, and its main goal is to retrieve e-Gov documentation. The system deals with Real-estate transaction documents, and gives citizens, business and governments the opportunity to integrate and recover documents. For this purpose EgoIR provides facilities that manage, search, and share e-Gov documentation. EgoIR also offers an ontology browsing capability using the ontologies described in section 4. These ontologies are stored in WebODE[5] (workbench for ontological engineering). Besides, EgoIR allows the construction of a query from the ontology concepts; the query obtained is composed of a set of concepts extracted from the ontologies. EgoIR connects to WebODE throughout WebODE's ODE service to obtain ontology concepts and it employs Lucene[11] (search engine library) to retrieve the documents that match the given query. The possible main users of EgoIR are: a) end users, who require consulting juridical documentation; b) agencies, which need to know the current legislation; and c) lawyers, who have to consult concrete aspects.

4 Legal Ontologies

The Legal Ontologies described in this section were built to represent the Real-estate transactions within the Spanish Government domain. These Legal Ontologies were developed with knowledge acquired by experts from academic and private sectors and built with the methodology METHONTOLOGY[5] and the workbench WebODE[5].

The Legal Ontologies provide support to the EgoIR aforementioned in three important ways: by concept-based indexing, by querying by inference and by improving the navigation. The EgoIR based on these Legal Ontologies bring much focused information, well-defined queries, well-organized information and a sophisticated navigation. The Legal Ontologies presented here are part of an EGO Ontology Model (Figure 2) being develop on this project, this model aims to represent a part of the legal processes carried out within the government.

The EGO Ontology Model reuses parts of the first two layers of LRI-Core model and is being adapted to the legal system of the Spanish government. The EGO Ontology Model is one of the first efforts not intended for legal domain but for e-Gov domain instead, which is a domain that needs to consider the law, regulations, citizen services, administrative processes, best-practices, and also the different languages spoken within the nation.

This domain by its nature is in need of special axioms. For instance, in the context of Government and public services, regulation (as a process) is the control of something by rules, accepted by the citizens. The acceptation is made in last instance: indirectly, by their democratic representants in the Parliament by the promulgation of "Laws". Regulation is a compromise between prohibition and no control at all. Public services can encounter conflict between commercial procedures and the interests of the people using these services. The Governments have some form of control or regulation to manage this possible conflict. This regulation needs to ensure that a safe and appropriate service is delivered, while not discouraging the effective functioning and development of

[11] `http://lucene.apache.org/`

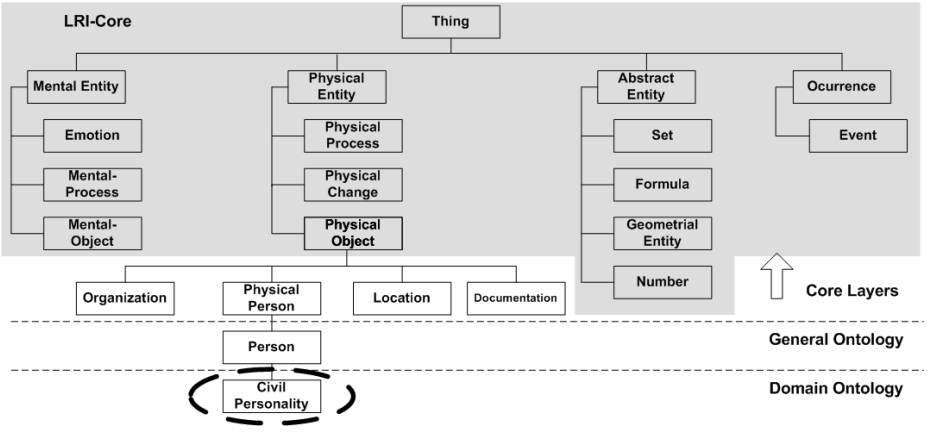


Fig. 2. Excerpt of the EGO Ontology Model

businesses. To model this legal knowledge, is needed a significant amount of axioms, that a this time are under develop.

4.1 Real-Estate Transaction Ontologies Roles

In [11,17] the five main roles of ontologies are identified: organizing and structuring information; reasoning and problem solving; semantic indexing and searching; semantics integrating and interoperating; and understanding the domain. Before building the Real-estate Transaction Ontologies, we think it should be useful to settle the proper role(s) that the ontology will play.

The Real-estate Transaction Ontologies (Figure 3) will perform three of the five roles mentioned above: the first role is that of organizing and structuring information in the e-Gov domain, mainly by defining the terms used. The second role is that of reasoning and problem solving; this role basically represents the knowledge of the domain so that an automated reasoner can represent problems and generate solutions for these problems, what implies the use of an inference engine to achieve specific goals. The third role is that of semantic indexing and searching (where the ontology will represent the contents of documents) that will enable semantic search for content.

Figure 3 shows the relationships between the Real-estate Transaction Ontologies aforementioned (each ontology is represented by a triangle). The aim of this figure is to show all the ad-hoc relations between the Real-estate Transaction Ontologies.

For the Reimdoc Project eleven ontologies have been developed: *person*, *civil personality*, *organization*, *location*, *tax*, *contract model*, *jurisprudence*, *Real-estate transaction verifications*, *Real-estate*, *legislation*, and *Real-estate transaction*.

Individually, they play the specific goals and model knowledge used in the Reimdoc Project. We describe next the relationships between the main ontologies.

The Civil Personality Ontology has as main concept the `civil person`, which is split into two subclasses: `natural person` (representing citizens), `juridical person` (representing enterprises, public administrations, etc.). The ad-hoc relations

specified for each concept are those relations whose domain is the concept. For example, the concept `civil person` has six binary relations: 'has data from juridical person', 'has residence', 'is buyer', 'is seller', 'realizes' and 'has data from Natural Person'.

The Real-estate Transaction Ontology has `Real-estate transaction` as main concept, which is split into two subclasses: `buy` (representing the action of buying), `sell` (representing the action of selling.). The concept `Real-estate transaction` has eight binary relations: 'is bought', 'is sold', 'based on' (tax, legislation, jurisprudence), 'acquires ', 'verifies' and 'uses'.

The Location Ontology has as main concept the `location`, which is split into three subclasses: `geographic division`, `town` and `country`. The concept `location` has two binary relations: 'is residence' and 'is associated'.

The Person Ontology has as main concept the `person`. The concept `person` has one binary relation: 'is associated'.

The Organization Ontology has as main concept the `organization`. The concept `organization` has one binary relation: 'is associated'.

The Real-estate Ontology has as main concept the `Real-estate`. The concept `Real-estate` has one binary relation: 'is associated'.

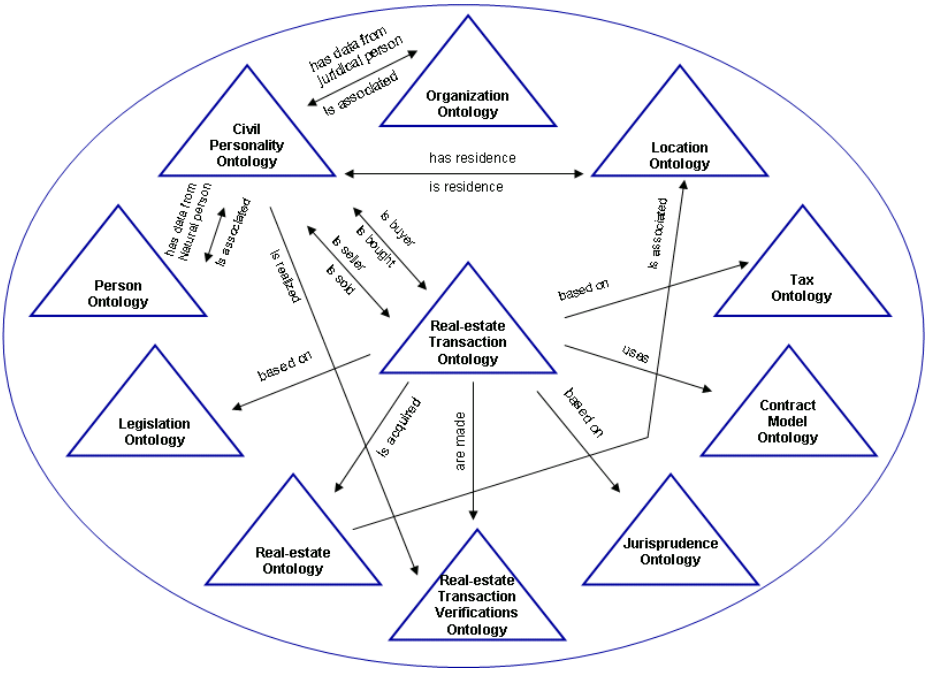


Fig. 3. Main ad-hoc relationships for the Real-estate Transaction Ontologies

4.2 Main Ontology Modelling Components

METHONTOLOGY[5] proposes to conceptualize ontologies with a set of tabular and graphical intermediate representations. Such intermediate representations allow modeling the components described in this section.

Concepts are taken in a broad sense. For instance, in the legal domain, concepts are: `Civil Personality`, `Natural Person`, `Juridical Person`, etc. Concepts in the ontology are usually organized in taxonomies through which inheritance mechanisms can be applied. For instance, we can represent a taxonomy of legal entities (which distinguishes persons and organizations), where a `Real-estate Contract` is a subclass of a `Contract`, etc.

Relations represent a type of association between concepts of the domain. If the relation links two concepts, for example, *has civil personality*, which links `Natural Person` to `Civil Personality`, it is called binary relation. An important binary relation is *Subclass-Of*, which is used for building the class taxonomy, as shown above. Each binary relation may have an inverse relation that links the concepts in the opposite direction.

Instances are used to represent elements or individuals in an ontology. An example of instance of the concept `Contract` is `Contract of merchanting Real estate`. Relations can be also instantiated. For example, we can express that Contract of merchanting real estate has a location in Madrid as follows: `has location (Contract of merchanting real estate, Madrid)`, using a first order logic notation.

Constants are numeric values that do not change for long time. For example: `legal age`.

Attributes describe properties of instances and of concepts. We can distinguish two types of attributes: instance and class attributes. ***Instance attributes*** describe concept instances, where they take their values. These attributes are defined in a concept and inherited by its sub-concepts and instances. For example, the `date of a Contract` is proper to each instance. ***Class attributes*** describe concepts and take their values in the concept where they are defined. Class attributes are neither inherited by the subclasses nor by the instances. An example is the attribute `First Name` as a part of `Natural Person`. Ontology development tools usually provide predefined domain-independent class attributes for all the concepts, such as the concept documentation, synonyms, acronyms, etc. Besides, other user-defined domain dependent class attributes can be usually created.

Formal axioms are logical expressions that are always true and are normally used to specify constraints in the ontology. An example of axiom is a Natural Person has legal capacity at the age of sixteen if he/she gets married.

Rules are generally used to infer knowledge in the ontology, such as attribute values, relation instances, etc. An example of rule is: a Natural Person could be a part of the Juridical Person.

Finally, we present the Real-estate Transaction Ontologies statistics: the number of concepts is 58, the number of relations is 20, the number of attributes is 59, the number of constants is 2 and 15 axioms.

5 Conclusions and Future Prospects

In this paper we have presented a set of legal ontologies for Real-estate transactions within the Spanish government domain as a part of the EGO Ontology model, which in turn is part of an ongoing project aiming, on the one hand, at supporting semantic applications to retrieve legal documents and, on the other, at delivering services from the public administration (within the government) to citizens. These legal ontologies are built following the methodology METHONTOLOGY and the workbench WebODE and are application independent.

The e-Gov domain does still have many needs: knowledge, for instance, has not been modeled at all. These needs represent real challenges for researchers. One problem to be solved in the near future is that of knowledge acquisition by legal experts. We must add here that the legal domain is very complex and evolving and its complexity provides a different situation than that provided by domains such as physics or mathematics, and this fact will bring about the deployment of future e-Gov ontologies.

At this time, legal ontologies built for e-Gov domain in Spain are at initial deployment and design, we used a couple of developed legal ontologies like FOL or LRI to have a initial reference framework. In Spain there are no specific government's efforts to fund this kind of research. However these projects are being funded under more general disciplines.

In our future work, we will be focused on further enhancement and evaluation of the Real-estate Transaction Ontologies; we will be centred on the reasoning capabilities of these ontologies; we will continue integrating the legal knowledge captured on the EGO Ontology Model and we will compare the model with other ontology models. Finally, we will evaluate the EgoIR application to improve its performance.

Acknowledgments. We would like to thank our project partners for their helpful comments. Specially thanks to María del Carmen Suárez-Figueroa, Fernando Galindo, Ana Ibarrola and Ferran Jornet.

This work is done within the ongoing Reimdoc Project which is supported by the Spanish Technology and Science Ministry (Project FIT-340100-2004-22). This project is partially funded by a scholarship granted through the Program of Teaching Staff Improvement (PROMEP) at the Tamaulipas, University.

EgoIR includes software developed by the Apache Software Foundation (`http://www.apache.org`).

References

1. Borst, W.N.: Construction of Engineering Ontologies. University of Tweenty. Enschede, NL-Centre for Telematica and Information Technology. (1997)
2. Breuker, J.: Constructing a Legal Core Ontology: LRI-Core.University of Amsterdam, Leibniz Center for Law, Amsterdam, the Netherlands (2004)
3. Curtain, G., Sommer, M., Vis-Sommer, V.(editors): The World of E-Government. The Haworth Press. (2003)
4. Electronic Government: First International Conference, EGOV 2002, Aix-en-Provence, France, September 2-5,. Proceedings (Lecture Notes in Computer Science) by Roland Traunmller (Editor), Klaus Lenk (Editor) (2002)

5. Gómez-Pérez, A.; Fernández-López, M.; Corcho, O. Ontological Engineering. Springer Verlag. 2003
6. Gruber, T.R.: A Translation Approach to Portable Oontologies. Knowledge Acquisition. 5(2) (1993) 199-220
7. Gruber, T.R.: Toward Principles for the Design of Ontologies Used for Knowledge Shar-ing. Presented at the Padua workshop on Formal Ontology, to appear in an edited collec-tion by Nicola Guarino (1993)
8. Hendler, J., Berners-Lee, T., and Miller, E. Integrating Applications on the Se-mantic Web, (2002) http://www.w3.org/2002/07/swint.html
9. Kralingen, R.W.: Frame-based Conceptual Models of Statute Law, Computer/Law Se-ries. No.16, Kluwer Law International. The Hague, the Netherlands (1995)
10. McCarty, L.T.: A Language for Legal Discourse, I. Basic Features, Proceedings of the Second International Conference on Artificial Intelligence and Law, Vancouver, Canada (1989) 180-189
11. Melz, E., Valente, A.: Modelling the Tax Code. The Second International Workshop on Regulatory Ontologies (2004)
12. Neches, R., Fikes, R.E., Finin, T., Gruber, T.R., Senator, T., Swartout, W.R.: Enabling Technology for Knowledge Sharing. AI Magazine. 12(3) (1991) 36-56.
13. Stamper, R.K.: LEGOL: Modelling Legal Rules by Computer. Computer Science and Law. Bryan Niblett. Cambridge University Press. Cambridge, United Kingdom (1980)
14. Stamper, R.K.: The Role of Semantics in Legal Expert Systems and Legal Reasoning. Ratio Juris, Vol. 4, No. 2 (1991) 219-244.
15. Studer, R., Benjamins, V.R., Fensel, D.: Knowledge Engineering: Principles and Methods. Data and Knowledge Engineering. 25: 161-197. (1998)
16. Valente, A.: Legal Knowledge Engineering; A Modelling Approach. University of Amsterdam, The Netherlands, IOS Press, Amsterdam, The Netherlands (1995)
17. Valente, A.: Types and Roles of Legal Ontologies. V.R. Benjamins et al. (Eds.): Law and the Semantic Web, (2005) 65-76
18. Visser, P.R.S.: Knowledge Specification for Multiple Legal Tasks; A Case Study of the Interaction Problem in the Legal Domain, Computer / Law Series. No. 17, Kluwer Law International. The Hague, The Netherlands (1995)

Mapping Conformant Planning into SAT Through Compilation and Projection

Héctor Palacios[1] and Héctor Geffner[2]

[1] Universitat Pompeu Fabra.
Paseo de Circunvalació, 8. Barcelona, Spain
hector.palacios@upf.edu
[2] ICREA & Universitat Pompeu Fabra.
Paseo de Circunvalació, 8. Barcelona, Spain
hector.geffner@upf.edu

Abstract. Conformant planning is a variation of classical AI planning where the initial state is partially known and actions can have non-deterministic effects. While a classical plan must achieve the goal from a given initial state using deterministic actions, a conformant plan must achieve the goal in the presence of uncertainty in the initial state and action effects. Conformant planning is computationally harder than classical planning, and unlike classical planning, cannot be reduced polynomially to SAT (unless P = NP). Current SAT approaches to conformant planning, such as those considered by Giunchiglia and colleagues, thus follow a generate-and-test strategy: the models of the theory are generated one by one using a SAT solver (assuming a given planning horizon), and from each such model, a candidate conformant plan is extracted and tested for validity using another SAT call. This works well when the theory has few candidate plans and models, but otherwise is too inefficient. In this paper we propose a different use of a SAT engine where conformant plans are computed by means of *a single SAT call over a transformed theory*. This transformed theory is obtained by *projecting* the original theory over the action variables. This operation, while intractable, can be done efficiently provided that the original theory is compiled into d–DNNF (Darwiche 2001), a form akin to OBDDs (Bryant 1992). The experiments that are reported show that the resulting COMPILE-PROJECT-SAT planner is competitive with state-of-the-art optimal conformant planners and improves upon a planner recently reported at ICAPS-05.

1 Introduction

Conformant planning is a variation of classical AI planning where the initial state is partially known and actions can have non-deterministic effects. While a classical plan must achieve the goal from a given initial state using deterministic actions, a conformant plan must achieve the goal in the presence of uncertainty in the initial state and action effects. Conformant planning is computationally harder than classical planning, and unlike classical planning, cannot be reduced

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 311–320, 2006.

polynomially to SAT. Current SAT approaches to conformant planning thus follow a generate-and-test strategy [1]: the models of the theory are generated one by one using a SAT solver (assuming a given planning horizon), and from each such model, a candidate conformant plan is extracted and tested for validity using another SAT call. This works well when the theory has few candidate plans and models, but otherwise is too inefficient. In this paper we propose a different use of a SAT engine where conformant plans are computed by means of *a single SAT call over a transformed theory.* This transformed theory is obtained by *projecting* the original theory over the action variables. Projection is the dual of variable elimination (also called forgetting or existential quantification): the projection of a formula over a subset of its variables is the strongest formula over those variables; e.g., the projection of $((x \wedge y) \vee z)$ over $\{x, z\}$ is $x \vee z$. While projection is intractable, it can be done efficiently provided that the theory is in a certain canonical form such as *deterministic Decomposable Negated Normal Form* or *d*–DNNF [2], a form akin to OBDDs [3].

Our scheme for planning is thus based on the following three steps: the planning theory in CNF is first compiled into d-DNNF, the compiled theory is then transformed into a new theory over the action variables only, and finally the conformant plan, if there is one, is obtained from this theory by a single invocation of a SAT engine. The experiments that are reported show that this COMPILE-PROJECT-SAT planner is competitive with state-of-the-art optimal conformant planners and improves upon a planner recently reported at ICAPS-05 [4].

Two recent optimal conformant planners are Rintanen's [5] and Palacios *et al.*'s [4]. The first performs heuristic search in belief space with a powerful, admissible heuristic obtained by precomputing distances over belief states with at most two states. The second is a branch-and-prune planner that prunes partial plans that cannot comply with some possible initial state. This is achieved by performing model-count operations in linear-time over the d–DNNF representation of the theory. Both schemes assume that all uncertainty lies in the initial situation and that *all actions are deterministic.* In this work, we maintain this simplification which is not critical as non-deterministic effects can be eliminated by adding a polynomial number of hidden fluents. An appealing feature of the new conformant planning scheme is that it is based on the *two off-the-shelf components:* a d–DNNF compiler and a SAT solver.

The paper is organized as follows. First we define the conformant planning problem and its formulation in propositional logic. Then we study how to obtain models corresponding to conformant plans. We look then at projection as a logical operation, and at d–DNNFs as a compiled normal form that supports projection in linear time. Finally, we present the complete conformant planning scheme, the experimental results, and some conclusions.

2 Planning and Propositional Logic

Classical planning consists of finding a sequence of actions that transforms a known initial state into a goal, given a description of states and actions in terms of a set

of variables. Conformant planning is a variation of classical planning where the initial state is partially known and actions can have non-deterministic effects. A conformant plan must achieve the goal for *any* possible initial state and transition.

We consider a language for describing conformant planning problems P given by tuples of the form $\langle F, O, I, G\rangle$ where F stands for the fluent symbols f in the problem, O stands for a set of actions a, and I and G are sets of clauses over the fluents in F encoding the initial and goal situations. In addition, every action a has a precondition $pre(a)$, given by a set of fluent literals, and a list of conditional effects $\text{cond}^k(a) \rightarrow \text{effect}^k(a)$, $k = 1, \ldots, n_a$, where $\text{cond}^k(a)$ and $\text{effect}^k(a)$ are conjunctions of fluent literals. As mentioned above, we assume that actions are deterministic and hence that all uncertainty lies in the initial situation.

In the SAT approach to classical planning [6], the problem of finding a plan for P within N time steps is mapped into a model finding problem over a suitable propositional encoding. In this encoding there are variables x_i for fluents and actions x where i is a temporal index in $[0, N]$ (no action variables x_i are created for $i = N$ though). For a formula B, B_i refers to the formula obtained by replacing each variable x in B by its time-stamped counterpart x_i. The encoding $T(P)$ of a *conformant planning problem* $P = \langle F, I, O, G\rangle$ is obtained as a slight variation of the propositional encoding used for classical planning [6]. For an horizon N, the CNF theory $T(P)$ is given by the following clauses:

1. **Init:** a clause C_0 for each init clause $C \in I$.
2. **Goal:** a clause C_N for each goal clause $C \in G$.
3. **Actions:** For $i = 0, 1, \ldots, N-1$ and $a \in O$:

$$\begin{aligned} a_i &\supset \text{pre}(a)_i && \text{(preconditions)} \\ \text{cond}^k(a)_i \wedge a_i &\supset \text{effect}^k(a)_{i+1}, \quad k = 1, \ldots, k_a && \text{(effects)} \end{aligned}$$

4. **Frame:** for $i = 0, 1, \ldots, N-1$, each fluent literal

$$l_i \wedge \bigwedge_{\text{cond}^k(a)} \neg[\text{cond}^k(a)_i \wedge a_i] \supset l_{i+1}$$

   where the conjunction ranges over the conditions $\text{cond}^k(a)$ associated with effects $\text{effect}^k(a)$ that support the complement of l.
5. **Exclusion:** $\neg a_i \vee \neg a'_i$ for $i = 0, \ldots, N-1$ if a and a' are incompatible.

The meaning of *Init*, *Goal*, and *Actions* is straightforward. *Frame* expresses the persistence of fluents in the absence of actions that may affect them. Finally *Exclusion* forbids the concurrent execution of actions that are deemed incompatible. For the serial setting, we regard every pair of different actions as incompatible, while in the parallel setting, we regard as incompatible pairs of different actions that interfere with each other.

A conformant planner is optimal when it finds conformant plans for the minimum possible horizon N (makespan). This is achieved by setting the horizon N to 0, and increasing it, one unit at a time, until a plan is found.

3 Conformant Planning and Models

In classical planning the relation between a problem P and its propositional encoding $T(P)$ is such that the models of $T(P)$ are in one-to-one correspondence with the plans that solve P (for the given horizon). In conformant planning, this correspondence no longer holds: the models of $T(P)$ encode 'optimistic plans', plans that work for *some* initial states and transitions but may fail to work for others, and hence are not conformant. We will see however that it is possible to transform the theory $T(P)$ so that the models of the resulting theory are in correspondence with the conformant plans for P.

Let $Plan$ denote a maximal consistent set of *action* literals a_i (*i.e.*, a full action valuation), let $Init$ denote the fragment of $T(P)$ encoding the initial situation, and let s_0 refer to a maximal consistent set of fluent literals f_0 compatible with $Init$ (i.e., a possible initial state which we denote as $s_0 \in Init$). Then for a *classical planning* problem P, $Plan$ is a solution if and only if

$$T(P) \ + \ Plan \qquad \text{is satisfiable.} \tag{1}$$

For a *conformant problem P with deterministic actions only*, on the other hand, $Plan$ is a solution if and only if

$$\forall\, s_0 \in Init: \ \ T(P) \ + \ Plan \ + \ s_0 \qquad \text{is satisfiable.} \tag{2}$$

In other words, in the conformant setting $Plan$ must work for all possible initial states.

In order to find a $Plan$ that complies with (1) it is enough to find a model of $T(P)$, and then set $Plan$ to the set of action literals that are true in the model. On the other hand, for finding a $Plan$ that complies with (2) this is not enough. As we will show, however, this will be enough when the theory $T(P)$ is transformed in a suitable way. As a first approximation, consider the problem of finding a $Plan$ that complies with

$$T(P)' \ + \ Plan \qquad \text{is satisfiable.} \tag{3}$$

where $T(P)'$ is a conjunction that takes into account *all* the initial states

$$T(P)' \quad = \quad \bigwedge_{s_0 \in Init} T(P) \,|\, s_0 \tag{4}$$

Here $T \,|\, X$ refers to theory T with variables x in T replaced by the value they have in X: true if $x \in X$, and false if $\neg x \in X$. This operation is known as value substitution or *conditioning* [2].

If equations (3) and (4) provided a correct formulation of conformant planning, we could obtain a conformant plan by finding a model for $T(P)'$, and extracting $Plan$ from the value of the action variables in that model.

The formulation (3-4), however, is *not* correct. The reason is that the theory $T(P)$ contains fluent variables f_i for times $i > 0$ which are neither in $Init$ nor in

Plan. In (2), these variables can take different values for each s_0, while in (3-4), these variables are *forced* to take the *same* value over all possible s_0.

We can modify however the definition of $T(P)'$ in (4) for obtaining a correct SAT formulation of conformant planning. For this we need to *eliminate* or *forget* the fluent variables f_i, for $i > 0$, from each conjunct $T(P) \,|\, s_0$ in (4).

The forgetting of a set of variables S from a theory T [7], also called elimination or existential quantification, is the dual operation to *Projection* of T over the *rest* of variables V; $V = vars(T) - S$. The projection of T over V, denoted $\mathsf{project}[\, T \,;\, V \,]$, refers to a theory over the variables V whose models are exactly the models of T restricted to those variables. For example, if $\phi = (a_1 \wedge f_1) \vee a_2$ then $\mathsf{project}[\, \phi; \{a_1, a_2\} \,] = a_1 \vee a_2$, which can also be understood as $\exists f_1 \phi \;=\; (\phi \,|\, f_1 = \mathsf{true}) \vee (\phi \,|\, f_1 = \mathsf{false}) \;=\; ((a_1 \wedge \mathsf{true}) \vee a_2) \vee ((a_1 \wedge \mathsf{false}) \vee a_2) \;=\; (a_1 \vee a_2)$. Getting rid of the fluent variables f_i for $i > 0$ in the conjuncts $T(P) \,|\, s_0$ in (4) simply means to project such formulas over the action variables, as the variables in $T(P) \,|\, s_0$ are either action variables or fluent variables f_i for $i > 0$ (the fluent variables f_i for $i = 0$ have been substituted by the their values in s_0).

The result is that the transformed theory $T(P)'$ becomes:

$$T_{\mathrm{cf}}(P) \quad = \quad \bigwedge_{s_0 \in Init} \mathsf{project}[\, T(P) \,|\, s_0 \;;\; Actions \,] \tag{5}$$

for which we can prove:

Theorem 1. *The models of $T_{cf}(P)$ in (5) are one-to-one correspondence with the conformant plans for the problem P.*

Equation 5 suggests a simple *scheme* for conformant planning: construct the formula $T_{\mathrm{cf}}(P)$ according to (5), and then feed this theory into a state-of-the-art SAT solver. The crucial point is the generation of $T_{\mathrm{cf}}(P)$ from the original theory $T(P)$: the transformation involves *conditioning* and *conjoining* operations, as well as *projections*. The key operation that is intractable is projection. It is well known however that projection, like many other intractable boolean transformations, can be performed in polynomial time provided that the theory is in a suitable compiled form [2]. Of course, the compilation itself may run in exponential time and space, yet this will not be necessarily so on average. We will actually show that the theory $T_{\mathrm{cf}}(P)$ in (5) can be obtained in time and space that is *linear* in the size of the d-DNNF compilation of $T(P)$.

4 Projection and d–DNNF

Knowledge compilation is concerned with the problem of mapping logical theories into suitable target languages that make certain desired operations tractable [2]. The compilation of theories into OBDDs is intractable, but has been found useful in formal verification and more recently in planning [8, 9, 10].

A more recent compilation language is *Deterministic Decomposable Negation Normal Form* (d–DNNF [2]). d-DNNFs support a rich set of polynomial

time operations and queries; in particular projection and model counting, that are intractable over CNFs, become linear operations over d-DNNFs. OBDDs are a special, less succinct class of d-DNNFs; in fact, there are OBDDs that are exponentially larger than their equivalent d–DNNFs but not the other way around [2].

4.1 Decomposability and Determinism of NNF

A propositional sentence is in Negation Normal Form (NNF) if it is constructed from literals using only conjunctions and disjunctions. A practical representation of NNF sentences is in terms of rooted directed acyclic graphs (DAGs), where each leaf node in the DAG is labeled with a literal, true or false; and each non-leaf (internal) node is labeled with a conjunction $\wedge$ or a disjunction $\vee$. Decomposable NNFs are defined as follows:

Definition 1 ([2]). *A* decomposable negation normal form (DNNF) *is a negation normal form satisfying* decomposition: *for any conjunction $\wedge_i \alpha_i$ in the form, no variable appears in more than one conjunct α_i.*

The satisfiability of a DNNF can thus be tested in linear time by means of a single bottom-up pass over its DAG. A useful subclass of DNNFs closely related to OBDDs is d–DNNF [2].

Definition 2 ([11]). *A* deterministic DNNF (d–DNNF) *is a* DNNF *satisfying* determinism: *for any disjunction $\vee_i \alpha_i$ in the form, every pair of disjuncts α_i is mutually exclusive.*

Model counting over d-DNNF can be done in time linear in the size of the DAG also by means of a simple bottom-up pass. A theory can be compiled into d–DNNF by applying the expansion $\Delta \equiv (\Delta \,|\, a \wedge a) \vee (\Delta \,|\, \neg a \wedge \neg a)$ recursively [12].

4.2 Projection and Conditioning in d–DNNF

For generating a theory equivalent to $T_{\mathrm{cf}}(P)$ in (5), we need to perform three logical transformations: conditioning, conjoining, and projection. Provided that the original theory $T(P)$ is compiled in d–DNNF, all these transformations can be performed in time linear in the size of its DAG representation by a single bottom-up pass. The resulting theory $T_{\mathrm{cf}}(P)$, however, is in NNF but not in DNNF due to the added conjunction, and this is why it cannot be checked for consistency in linear time, and has to be fed into a SAT solver.

5 Conformant Planner

Integrating the previous observations, the proposed conformant planner involves the following steps. First, a CNF theory $T(P)$ is obtained from a PDDL-like description of the planning problem. Then

1. The theory $T(P)$ is **compiled** into the d–DNNF theory $T_{\mathrm{C}}(P)$
2. From $T_{\mathrm{C}}(P)$, the transformed theory

$$T_{\mathrm{cf}}(P) \quad = \quad \bigwedge_{s_0 \in Init} \mathsf{project}[\, T_{\mathrm{c}}(P) \,|\, s_0 \;;\; Actions \,]$$

is obtained by operations that are linear in time and space in the size of the DAG representing $T_{\mathrm{c}}(P)$. The resulting theory $T_{\mathrm{cf}}(P)$ is in NNF but due to the added conjunction is *not* decomposable.

3. The NNF theory $T_{\mathrm{cf}}(P)$ is converted into CNF and a **SAT solver** is called upon it.

This sequence of operations is repeated starting from a planning horizon $N = 0$ which is increased by 1 until a solution is found.

Some of the details of the generation of the target theory $T_{\mathrm{cf}}(P)$ from the compiled theory $T_{\mathrm{c}}(P)$ are important. In particular, it is necessary to compile $T(P)$ into $T_{\mathrm{c}}(P)$ using an ordering of variables that expands on the $Init$ variables first; this is so that the DAG representing the d-DNNF subtheories $T_{\mathrm{c}}(P)|s_0$ for each possible initial state s_0, all correspond to (non-necessarily disjoint) fragments of the DAG representing the compiled d-DNNF theory $T_{\mathrm{c}}(P)$. Then the DAG representing the target NNF theory $T_{\mathrm{cf}}(P)$, which is no longer decomposable, is obtained by conjoining these fragments.

6 Experimental Results

We performed experiments testing the proposed optimal conformant planner on a Intel/Linux machine running at 2.80GHz with 2Gb of memory. Runs of the d–DNNF compiler and the SAT solver were limited to 2 hours and 1.8Gb of memory. The d–DNNF compiler is Darwiche's `c2d` v2.18 [12], while the SAT solver is `siege_v4` except for very large CNFs that would not load, and where *zChaff* was used instead. We used the same suite of problems as [4] and [5]. Most are challenging problems that emphasize the critical aspects that distinguish conformant from classical planning. **Ring:** A robot can move in n rooms arranged in a circle. The goal is to have all windows closed and locked. **Sorting Networks:** The task is to build a circuit of compare-and-swap gates to sort n boolean variables. **Square-center:** A robot without sensors can move in a grid of $2^n \times 2^n$, and its goal is to get to the middle of the room. For this it must first locate itself into a corner. **Cube-center:** Like the previous one, but in three dimensions. **Blocks:** Refers to the blocks-world domain with move-3 actions but in which the initial state is completely unknown. Actions are always applicable but have an effect only if their normal 'preconditions' are true. The goal is to get a fixed ordered stack with n blocks. None of the problems feature preconditions, and only sorting, square-center, and cube-center admit parallel solutions.

We report compilation and search times. The first is the time taken by the d–DNNF compiler; the second is the time taken by the SAT solver. For the search part, we show the results for both the optimal horizon N^* and $N^* - 1$. The first shows the difficulty of finding conformant plans; the second, the difficulty of proving them optimal. These times dominate the times consumed in the previous iterations.

In Table 1 we show results of the compilation for optimal horizons in the serial setting. The compilation of theories for smaller horizons or parallel formulations is normally less expensive. The table shows the optimal horizon N^* for each problem, the size of the original CNF theory $T(P)$, the size of the DAG representing the compiled theory $T_{\mathrm{c}}(P)$ with the time spent in the compilation, and finally the size of the target theory $T_{\mathrm{cf}}(P)$ in CNF that is fed to the SAT solver. The first thing to notice is that all the problems considered in [5] compile. Thus, as in [4], *the compilation is not the bottleneck.*

Table 1. Compilation data for sequential formulation and optimal horizon N^*. On the left, the size of the theories $T(P)$ encoding the conformant planning problems, on the center, the size of the DAGs representing the compiled theories $T_{\mathrm{c}}(P)$ and the times spent in the compilation; on the right, the size of the target theories $T_{\mathrm{cf}}(P)$ in CNF that are passed to the SAT engine.

		CNF theory		d–DNNF theory			$T_{\mathrm{cf}}(P)$	
problem	N^*	vars	clauses	nodes	edges	time	vars	clauses
ring-r7	20	1081	3683	1008806	2179064	192.2	976203	3105362
ring-r8	23	1404	4814	3887058	8340295	1177.1	3779477	11957085
blocks-b3	9	444	2913	5242	20229	0.3	4667	23683
blocks-b4	26	3036	40732	226967	888847	124.5	223260	1104383
sq-center-e3	20	976	3642	11566	22081	1.1	9664	27956
sq-center-e4	44	4256	16586	90042	174781	47.1	81404	238940
cube-center-c9	33	2700	10350	282916	574791	98.9	276474	839027
cube-center-c11	42	4191	16227	658510	1330313	371.6	647994	1958472
sort-s7	16	1484	6679	115258	283278	12.4	112756	390997
sort-s8	19	2316	12364	363080	895247	77.2	359065	1246236

Table 2 shows the results of the SAT solver over the transformed theory $T_{\mathrm{cf}}(P)$ for both the optimal horizon N^* and N^*-1, and for both the sequential and parallel formulations. While not all problems are solved, the results improve upon those recently reported in [4], solving one additional instance in square-center and sorting. This represents an *order of magnitude* improvement over these domains. In blocks, on the other hand, there is no improvement, while the largest ring instances resulted in very large CNF theories that could not be loaded into Siege but were loaded and solved by zChaff (except for ring-r8 under the optimal planning horizon).

7 Discussion

We presented a COMPILE-PROJECT-SAT scheme for computing optimal conformant plans. The scheme is simple and uses two off-the-shelf components: a d–DNNF compiler and a SAT solver. We are currently exploring a variation of the COMPILE-PROJECT-SAT scheme that may be more suitable for dealing with problems that are not that far from classical planning, such as those considered

Table 2. Results for the Search: SAT calls over the tranformed theory $T_{\rm cf}(P)$ for the optimal horizon N^* (left) and $N^* - 1$ (right), both for sequential and parallel formulations (when they differ). We show the number of initial states, the time spent on the SAT call, the number of decisions made, and the number of actions in the plan found. Entries '$> 2h$' and '> 1.8Gb' mean time or memory exceeded. The sign $^\circ$ indicates that the SAT solver used was *zChaff*, as `siege_v4` could not load $T_{\rm cf}(P)$ due to its size. Times are in seconds.

			search with horiz k			s. with horizon $k-1$	
problem	N^*	$\#S_0$	time	decisions	#act	time	decisions
serial theories							
ring-r7	20	15309	$^\circ$ 2.1	2	20	$^\circ$ 0.8	0
ring-r8	23	52488	$>$ 1.8Gb			$^\circ$ 2.4	0
blocks-b3	9	13	0.1	1665	9	0.2	3249
blocks-b4	26	73	$> 2h$			$> 2h$	
sq-center-e3	20	64	18.8	52037	20	207.4	207497
sq-center-e4	44	256	5184.4	1096858	44	$> 2h$	
cube-center-c7	24	343	3771.5	578576	24	5574.2	736567
cube-center-c9	33	729	$> 2h$			$> 2h$	
sort-s5	9	32	0.0	352	9	22.0	35053
sort-s6	12	64	40.0	34451	12	$> 2h$	
sort-s7	16	128	3035.6	525256	16	$> 2h$	
sort-s8	19	256	$> 2h$			$> 2h$	
parallel theories							
sq-center-e3	10	64	0.5	2737	20	0.3	1621
sq-center-e4	22	256	423.1	244085	44	1181.5	439532
cube-center-c7	8	343	6.1	4442	24	2.9	1892
cube-center-c9	11	729	114.6	27058	33	156.0	32760
cube-center-c11	14	1331	$>$ 1.8Gb			181.5	13978
sort-s7	6	128	46.1	18932	18	355.4	48264
sort-s8	6	256	$^\circ$ 4256.6	533822	23	$> 2h$	

in [13]. When the number of possible initial states s_0 is low, rather than getting rid of the fluent variables f_i, $i > 0$, by projection as in

$$L \quad = \quad \bigwedge_{s_0 \in Init} \mathsf{project}[\, T(P) \,|\, s_0 \;;\; Actions \,] \tag{6}$$

it may be more convenient to introduce copies of them, one for each possible initial state $s0$, resulting in the different formula

$$L' \quad = \quad \bigwedge_{s_0 \in Init} [\, T(P)^{s_0} \,|\, s_0 \,] \tag{7}$$

where each $T(P)^{s_0}$ denotes a theory which is like $T(P)$ except that the fluent variables f_i, $i > 0$, are replaced by fresh copies $f_i^{s_0}$. Action variables, on the other hand, remain shared among all these theories. It can be shown that models of L' as well as the models of L, are in one-to-one correspondence with the conformant plans that solve the problem. The latter approach, which does not

require projection or compilation, may work better when the number of possible initial states is low, and collapses to the standard SAT approach to classical planning when all the uncertainty is gone.

In [14] Rintanen solves conformant planning problems by mapping them into QBFs of the form $\exists plan \forall s_0 \exists f_i \ \phi$. Our scheme for conformant planning can be extended also for QBFs of this form where it would have many elements in common with the "Eliminate and Expand" method of Biere [15], where the elimination is carried out by resolution rather than projection.

Acknowledgments

We thank Adnan Darwiche and Blai Bonet for our joint work in [4] that led to this research, and for making their d-DNNF compiler and PDDL to CNF translator, respectively, available to us. H. Geffner is partially supported by grant TIN2005-09312-C03-03 from MEC/Spain.

References

[1] Ferraris, P., Giunchiglia, E.: Planning as satisfiability in nondeterministic domains. In: Proceedings AAAI-2000. (2000) 748–753
[2] Darwiche, A., Marquis, P.: A knowledge compilation map. Journal of Artificial Intelligence Research **17** (2002) 229–264
[3] Bryant, R.E.: Symbolic Boolean manipulation with ordered binary-decision diagrams. ACM Computing Surveys **24** (1992) 293–318
[4] Palacios, H., Bonet, B., Darwiche, A., Geffner, H.: Pruning conformant plans by counting models on compiled d-DNNF representations. In: Proc. of the 15th Int. Conf. on Planning and Scheduling (ICAPS-05), AAAI Press (2005) 141–150
[5] Rintanen, J.: Distance estimates for planning in the discrete belief space. In: Proc. AAAI-04. (2004) 525–530
[6] Kautz, H., Selman, B.: Pushing the envelope: Planning, propositional logic, and stochastic search. In: Proceedings of AAAI-96. (1996) 1194–1201
[7] Lin, F., Reiter, R.: Forget it! In: Working Notes, AAAI Fall Symposium on Relevance, American Association for Artificial Intelligence (1994) 154–159
[8] Giunchiglia, F., Traverso, P.: Planning as model checking. In: Proceedings of ECP-99, Springer (1999)
[9] Clarke, E., Grumberg, O., Peled, D.: Model Checking. MIT Press (2000)
[10] Cimatti, A., Roveri, M.: Conformant planning via symbolic model checking. Journal of Artificial Intelligence Research **13** (2000) 305–338
[11] Darwiche, A.: On the tractable counting of theory models and its applications to belief revision and truth maintenance. J. of Applied Non-Classical Logics (2002)
[12] Darwiche, A.: New advances in compiling cnf into decomposable negation normal form. In: Proc. ECAI 2004. (2004) 328–332
[13] Brafman, R., Hoffmann, J.: Conformant planning via heuristic forward search: A new approach. In: Proceedings of the 14th International Conference on Automated Planning and Scheduling (ICAPS-04). (2004)
[14] Rintanen, J.: Constructing conditional plans by a theorem-prover. Journal of Artificial Intelligence Research **10** (1999) 323–352
[15] Biere, A.: Resolve and expand. In: Proc. 7th Intl. Conf. on Theory and Applications of Satisfiability Testing (SAT'04). Volume LNCS 3542., Springer (2005)

Multiagent Architecture for Monitoring the North-Atlantic Carbon Dioxide Exchange Rate

Javier Bajo[1] and Juan M. Corchado[2]

[1] Universidad Pontificia de Salamanca
C/Compañía 5, 37002, Salamanca, Spain
jbajope@upsa.es
[2] Departamento Informática y Automática
Universidad de Salamanca
Plaza de la Merced s/n
37008, Salamanca, Spain
corchado@usal.es

Abstract. This paper presents an architecture that makes it possible to construct dynamic systems capable of growing in dimension and adapting its knowledge to environmental changes. An architecture must define the components of the system (agents in this case), as well as the way in which those components communicate and interact with each other in order to achieve the system's goals. The work presented here focuses on the development of an agent-based architecture, based on the use of deliberative agents, that incorporate case based reasoning. The proposed architecture requires an analysis and design methodology that facilitates the building of distributed systems using this technology. The proposal combines elements of existing methodologies such as Gaia and AUML in order to take advantage of their characteristics. Moreover the architecture takes into account the possibility of modelling problems in dynamic environments and therefore the use of autonomous models that evolve over time. To solve this problem the architecture incorporates CBR-agents whose aim is to acquire knowledge and adapt themselves to environmental changes. The architecture has been applied to model for evaluating the interaction between the atmosphere and the ocean, as well as for the planification and optimization of sea routes for vessels. The system has been tested successfully, and the results obtained are presented in this paper.

1 Introduction

An architecture must define the components of the system, as well as the way in which those components communicate and interact with each other in order to achieve the system's goals. An agent architecture should provide support for the basic properties of an agent: autonomy, communication, learning, goal orientation, mobility, persistence, etc. Autonomy, learning and reasoning are especially important aspects for an agent. These capabilities can be modelled in different ways and with different tools [22]. One of the possibilities is the use of Case

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 321–330, 2006.

Based Reasoning (CBR) systems. This paper presents a CBR-agent based architecture that is the core of a distributed system, principally characterized by the utilization of CBR-BDI agents [5]. These agents are capable of learning from initial knowledge. They interact with the environment and with the users within the system and adapt themselves to environmental changes. The mission of the distributed system is to monitor the interaction between the ocean surface and the atmosphere. Initially the system has been used to evaluate and predict de quantity of CO2 exchanged in the North Atlantic Ocean as well as generating optimal sea routes for vessels. The aim of this work is to obtain an architecture that makes it possible to construct dynamic systems capable of growing in dimension and adapting its knowledge to environmental changes. Several architectures have been proposed for building deliberative agents, most of them based on the BDI model. In the BDI model the internal structure of an agent and therefore its ability to choose a course of action is based on mental attitudes. The advantage of using mental attitudes in the design and realization of agents and multi-agent systems is the natural (human-like) modelling and the high abstraction level. The BDI model uses Beliefs as information attitudes, Desires as motivational attitudes and Intentions as deliberative attitudes for each agent. The method proposed in [4,10] facilitates the CBR systems incorporation as a reasoning engine in BDI agents, which makes it possible for an agent to have at its disposal a learning, adaptation and a greater degree of autonomy than a pure BDI architecture [11].

One of the greatest obstacles for the development of agent based architecture is that, we do not yet have at our disposal compliance standards or fully developed methodologies that guide us through the steps needed for optimal analysis and design. Some methodologies have been proposed such as Gaia [22], AUML [1,15,16], MAS-CommonKADS [12], MaSE [8], ZEUS [14], MESSAGE [9]. However, generally, these methodologies are incomplete or present certain restrictions. In this study the decision was made to carry out an analysis and design for our MAS using a combination of elements from the Gaia and Agent Unified Modelling Language (AUML) methodologies. Gaia is an uncomplicated methodology that allows a simple analysis and initial design, through which the problem can be studied at a general level. The advantage is that it is possible to obtain a quick, low-detailed study. On the other hand, the problem arises once the Gaia design has been completed and the level of abstraction is found to be too high. As far as AUML is concerned, the final design is sufficiently accurate for immediate implementation, but it begins the study of the problem at a level that is overly specific and detailed level. Our idea was to take advantage of both methodologies: to carry out an initial Gaia analysis and design, and subsequently, after taking into account the appropriate changes, to continue with a detailed AUML design. This makes it possible to obtain a general view of the problem in terms of organization as well as a detailed description of the MAS, greatly facilitating in the development of the project.

BDI agents can be implemented by using different tools. One very interesting tool is Jadex [17], a BDI reasoning engine that can be used on top of different

middleware infrastructures such as JADE [2]. Jadex agents deal with the concepts of beliefs, goals and plans. Beliefs, goals and plans are objects that can be created and handled within the agent at execution time. A belief can be any type of java object and is stored within the beliefs base. A goal represents a motivation that has influence on agent behaviour. A plan is a java procedure and is executed in order to achieve goals. Jadex has the advantage of allowing programmers to include their own deliberative mechanisms. In our case this mechanism will be a CBR system. Moreover our system will benefit from all the communication advantages that JADE provides.

In the next section we review the relationships that can be established between CBR and BDI concepts. Section three describes the environmental problem that motivates most of this research. Section four describes the multiagent based system developed, paying special attention to the CBR-BDI agents constructed. Finally the conclusions and some preliminary results are presented.

2 CBR-BDI Agents

CBR is a paradigm based on the idea that similar problems have similar solutions, so that a new problem is solved by consulting the cases memory in search of a similar case solved in the past. The deliberative agents, proposed in the framework of this research, use this concept to gain autonomy and improve their problem solving capabilities. Figure 1 shows the activity diagram of a CBR-BDI agent for one of the possible actions, which is composed of a reasoning cycle that consists of four sequential phases: retrieve, reuse, revise and retain. An additional activity, revision of the expert's knowledge, is required because the memory can change as new cases appear during this process. Each of these activities can be automated, which implies that the whole reasoning process can be automated to a certain extent [11]. According to this, agents implemented using CBR systems could reason autonomously and therefore adapt themselves to environmental changes.

The CBR system is completely integrated into the agents' architecture. The CBR-BDI agents incorporate a 'formalism' which is easy to implement, whereby the reasoning process is based on the concept of intention. Intentions can be seen as cases, which have to be retrieved, reused, revised and retained. This makes

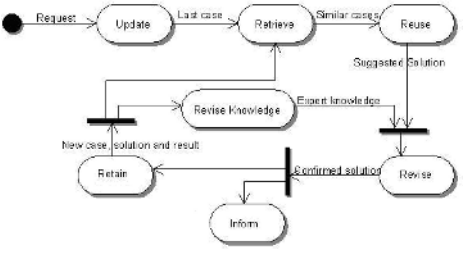


Fig. 1. Activity diagram for a CBR-BDI agent, including the reasoning cycle

the model unique in its conception and reasoning capabilities. The structure of the CBR system has been designed around the concept of a case.

The relationship between CBR systems and BDI agents can be established by implementing cases as beliefs, intentions and desires which lead to the resolution of the problem. As described in [7], in a CBR-BDI agent, each state is considered as a belief; the objective to be reached may also be a belief. The intentions are plans of actions that the agent has to carry out in order to achieve its objectives [3], so an intention is an ordered set of actions; each change from state to state is made after carrying out an action (the agent remembers the action carried out in the past when it was in a specified state, and the subsequent result obtained). A desire will be any of the final states reached in the past (if the agent has to deal with a situation, which is similar to one in the past, it will try to achieve a similar result to that previously obtained).

3 Air Sea Interaction Problem

In recent years a great interest has emerged in climactic behaviour and the impact that mankind has had on the climate. One of the most worrying factors is the quantity of CO2 present in the atmosphere. CO2 is one of gases produced by the greenhouse effect, and contributes to the fact that the Earth has a habitable temperature, provided that its quantity is limited. Without carbon dioxide, the earth would be covered in ice. On the other hand, excess CO2 blocks the heat transfer into the atmosphere, acting as an infrared radiation absorbent, preventing heat from leaving the atmosphere, therevy causing excessive warming of the planet [18]. Until only a few years ago, the photosynthesis and breathing processes in plants were considered as the regulatory system that controls the presence of CO2 in the atmosphere. However, the role played by the ocean in the regulation of carbon volume is very significant and so far remains indefinite [19]. Current technology offers the possibility of obtaining data and estimates that were beyond expectations only a few years ago. These data offer insights into the biological processes which govern the sink/source conditions for carbon dioxide [13,20]. Based on the knowledge acquired, we will be able to make predictions on the future behaviour of the atmosphere.

The goal of our project is to construct a model that calculates the global air-sea flux of CO2 exchanged between the atmosphere and the surface waters of the ocean, as well as the global budgets of CO2 for the whole oceanographic basin. In order to create a new model for the CO2 exchange between the atmosphere and the oceanic surface a number of important parameters must be taken into consideration: sea surface temperature, air temperature, sea surface salinity, atmospheric and hydrostatic pressures, the presence of nutrients and the wind speed vector (module and direction). These parameters can be obtained from oceanographic ships as well as from satellite images. Satellite information is vital for the construction of oceanographic models, and in this case, in order to produce estimates of air-sea fluxes of CO2 with much higher spatial and temporal resolution, using artificial intelligence models than can be achieved

realistically by direct in situ sampling of upper ocean CO2. In order to handle all the potentially useful data to create daily models in reasonable time and at a reasonable cost, it is necessary to use automated distributed systems capable of incorporating new knowledge. Our proposal is presented in the following section.

4 Multiagent System for the Air-Sea Interaction

The option we have chosen in order to find a suitable analysis and design methodology that could be applied to our problem, was to use a mixed methodology with concepts from the Gaia [22] and AUML [1,15,16] methodologies. Our aim is to take advantage of the strengths of both concepts. The Gaia methodology is based on organizational criteria that allows a quick and effective analysis and design. The results obtained after applying Gaia consist of a high abstraction level of complexity. At that moment, the Gaia design must be adapted so that AUML techniques can be applied. Figure 2 shows the steps followed during our development process. It can be seen that Gaia is used to obtain a high level analysis and design and AUML is applied then to obtain a detailed low level design.

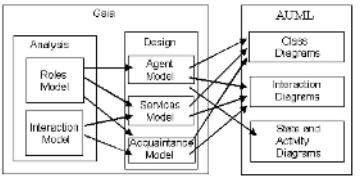


Fig. 2. Methodology used for the development process

4.1 Gaia Analysis and Design

Gaia is a methodology for agent-oriented analysis and design. The Gaia methodology is both general, in the sense that it is applicable to a wide range of multi-agent systems, and comprehensive, in the sense that it deals with both the macro-level (societal) and the micro-level (agent) aspects of systems. Gaia is founded on the view of a multi-agent system as a computational organisation consisting of various interacting roles [22]. Gaia analysis involves two models, the role model and the interaction model. Based on the requirements of the air-sea interaction problem, six roles are defined: The STORING role deals with the obtaining and storing of permanent data in the appropriate data bases. The PROCESSING role transforms the satellite images into cases. The DATACAPTURING role obtains data coming from Vessels. The CONSTRUCTAPARTIALCO2MODEL role deals with the generation of models. The OBTAINCO2EXCHANGE role calculates the CO2 exchange rate by means of the use of the models available. The AUTOEVALUATION role evaluates a model by comparing its parameters with the real data obtained through the Vessels' sensors. Finally, the PROCESSINGINFORMATION role allows a

user to interact with the system. Figure 3 shows the role model for the OBTAINCO2EXCHANGE role, with a specification of its characteristic attributes such as: responsibilities, permissions, activities and protocols [22].

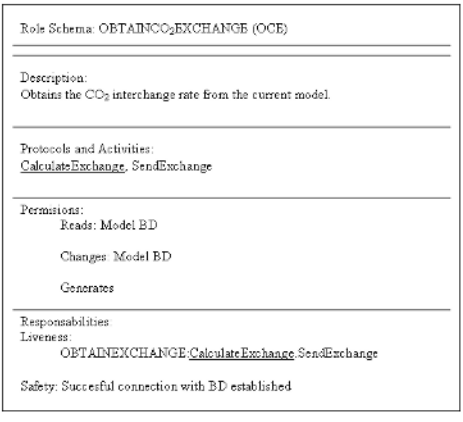


Fig. 3. Gaia roles model for the OBTAINCO2EXCHANGE role

As far as interaction model is concerned, dependences and relationships between roles must be established. Each interaction between two roles is modelled by means of a protocol. In our MAS the following interaction protocols are required: ObtainNewModelSuper, ObtainNewModelAuto, ObtainStore, ObtainStModel, ObtainNewModelStoring, ObtainInsituData, ObtainConstructData, ObtainVessel, ObtainEvaluationSuper, ObtainEvaluationDC, Activate/Deactivate Sensors, Delete EPROM, ChangeStore, ChangeCase, ObtainVesselData and ObtainStExchange.

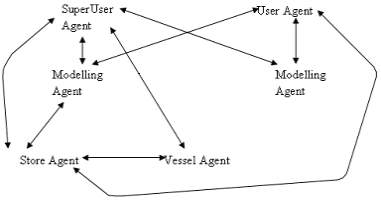


Fig. 4. Gaia acquaintance model for the air-sea interaction MAS.

As far as the Gaia design is concerned, the aim is to reduce the abstraction level so that traditional techniques can be applied. Three models are studied: agent model, service model and acquaintance model [22]. Figure 4 shows the acquaintance model for our system. Each agent communicates with the other agents. For example, the Vessel agent communicates with the SuperUser and Store agents.

4.2 Detailed AUML Design

AUML is a methodology that works at a highly detailed level, maybe too highly detailed in its initial stages if the problem we are working with is of a significant size. Our proposal deals with how to use the high level analysis and design obtained through the Gaia methodology to achieve a low level AUML design, with enough detail for an implementation to be carried out.

There are three concepts that vary slightly with respect to their meaning in Gaia and AUML: role, service and capability [1,15]. The AUML design provides class diagrams for each agent, collaboration or sequence diagrams, state and activity diagrams and protocol diagrams [1,15,16].

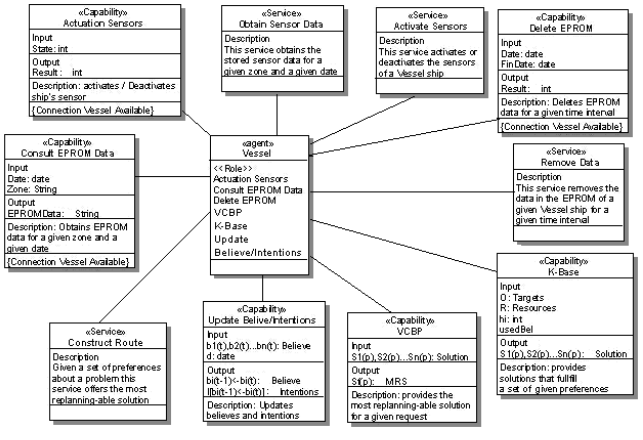


Fig. 5. Class diagram for the Vessel agent

Two CBR-BDI are used in our MAS, the Modelling agent and the Vessel agent. The first controls the generation of air-sea interaction models, as well as the management of model interaction. The second controls the management of vessels (oceanographic ships that collect insitu data required to validate the models), predominantly, the planning and optimizing of the routes taken by the vessels [10]. Figure 5 shows the class diagram for one of the two CBR-BDI agents in our system. The Vessel agent has six capabilities and offers four services to the rest of the agents in the MAS. Meanwhile, through its capabilities of Update Relieve/Intentions, K-Base and VCBP, the Vessel agent is able to carry out the stages of a CBR cycle and provide the most suitable route for a vessel based on parameters such as sea currents, obstacles, meteorological conditions, etc., when a destiny request is made. We complete the AUML design obtaining the collaboration and sequence diagrams and the protocol diagrams to represent the different interactions in the system.

Once the design is complete, the implementation can be carried out. The implementation is carried out by using the Jadex platform, which is a tool that incorporates the BDI model into the JADE agents, and the JADE platform.

The Modelling and Vessel agents are constructed with Jadex and the rest of the agents that compose the MAS are constructed with JADE. The communication mechanisms are those defined by JADE [2,17].

5 Results and Conclusions

The system described above was tested in the North Atlantic Ocean during 2004. During this period the multi-agent system has been tuned and updated and the first autonomous prototype began work in May 2004. Although the system is not fully operational and the aim of the project is to construct a research prototype and not a commercial tool, the initial results have been very successful from the technical and scientific point of view. The construction of the distributed system has been relatively simple using previously developed CBR-BDI libraries [5,6,7,10]. From the software engineering point of view AUML [1,15,16] and Gaia [22] provide an adequate framework for the analysis and design of distributed agent based systems. The formalism defined in [11] facilitates the straight mapping between the agent definition and the CBR construction. Figure 6 shows a screen shot of one of the possible views of the User agent. The user can interact with the Modelling agent (via his/her User agent) and obtain information about the carbon dioxide exchange rate of a given area.

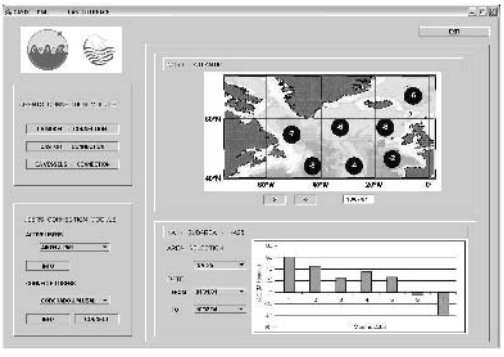


Fig. 6. Screen shot of an User agent

The fundamental concept when we work with a CBR system is the concept of case, and it is necessary to establish a case definition. A case in our problem, managed by the Modelling agent, is composed of the attributes described in Table 1. Cases can be viewed, modified and deleted manually or automatically by the agent (during its revision stage). The agent plans (intentions) can be generated using different strategies since the agent integrates different algorithms.

The interaction between the system developers and oceanographers with the multiagent system has been continuous during the construction and pruning period, from December 2003 to September 2004. The system has been tested

Table 1. Cases values

Case Field	Measurement
DATE	Date
LAT	Latitude
LONG	Longitude
SST	Temperature
S	Salinity
WS	Wind strength
WD	Wind direction
Fluo_calibrated	fluorescence calibrated with chlorophyll
SWp CO2	Surface partial pressure of CO2

Table 2. Million of tones of C02 exchanged in the North Atlanthic

	Oct. 04	Nov. 04	Dec. 04	Jan. 05	Feb. 05
Multiagent System	-18	19	31	29	28
Manual models	-20	25	40	37	32

during the last three months of 2004 and the results have been very accurate. Table 2 presents the results obtained with the Multiagent systems and with mathematical Models [13] used by oceanographers to identify the amount of CO2 exchanged. The numerical values represent the million of Tonnes of carbon dioxide that have been absorbed (negative values) or generated (positive value) by the ocean during each of the three months. The values proposed by the CBR-BDI agent are relatively similar to the ones obtained by the standard technique. In this case the case base has been constructed with over 100,000 instances, and includes data since 2002. The multiagent system has automatically incorporated over 20,000 instances during these three months and eliminated 13% of the initial ones. While the CBR-BDI Modelling Agent generates results on a daily basis without any human intervention, the Casix manual modelling techniques require the work of one researcher processing data during at least four working days. Although the system proposed requires further improvements and more work the initial results are very promising. The VCBP CBR systems embedded within the Vessel agent has provided relatively accurate results in the routes generated in the North Atlantic Waters [10]. The framework generated facilitates the incorporation of new agents using different modelling techniques and learning strategies so that further experiments will allow us to compare these initial results with the ones obtained by other techniques.

References

1. Bauer, B. and Huget, M. P. (2003) FIPA Modeling: Agent Class Diagrams.
2. Bellifime, F. Poggi, A. and Rimasa, G. (2001) JADE: a FIPA2000 compliant agent development environement. Proceedings of the 5th international conference on autonomous agents (ACM).

3. Bratman M.E., Israel D., and Pollack M.E. (1988). Plans and resource-bounded practical reasoning. Computational Intelligence, 4. pages 349-355.
4. Bratman, M.E. (1987). Intentions, Plans and Practical Reason. Harvard University Press, Cambridge, M.A.
5. Corchado J. M. and Laza R. (2003). Constructing Deliberative Agents with Case-based Reasoning Technology, International Journal of Intelligent Systems. Vol 18, No. 12, December. pp.: 1227-1241
6. Corchado J. M. and Lees B. (2001). A Hybrid Case-based Model for Forecasting. Applied Artificial Intelligence. Vol 15, no. 2, pp.105-127.
7. Corchado J. M., Pavón J., Corchado E. and Castillo L. F. (2005) Development of CBR-BDI Agents: A Tourist Guide Application. 7th European Conference on Case-based Reasoning 2004. Lecture Notes in Artificial Intelligence 3155, Springer Verlag. pp. 547-559.
8. DeLoach, S. (2001) Anlysis and Design using MaSE and AgentTool. Proceedings of the 12th Midwest Artificial Intelligence and Cognitive Science Conference (MAICS).
9. EURESCOM (2001) MESSAGE: Methodology for engineering systems of software agents. Technical report P907-TI1, EURESCOM.
10. Glez-Bedia M. and Corchado J. M. (2002) A planning strategy based on variational calculus for deliberative agents. Computing and Information Systems Journal. Vol 10, No 1, 2002. ISBN: 1352-9404, pp. 2-14.
11. Glez-Bedia M., Corchado J. M., Corchado E. S. and Fyfe C. (2002) Analytical Model for Constructing Deliberative Agents, Engineering Intelligent Systems, Vol 3: pp. 173-185.
12. Iglesias, C., Garijo, M., Gonzalez J.C. and Velasco J. R. (1998) Analysis and Design using MAS-CommonKADS. Inteligent Agents IV LNAI Volume 1365 Springer Verlag.
13. Lefevre N., Aiken J., Rutllant J., Daneri G., Lavender S. and Smyth T. (2002) Observations of pCO2 in the coastal upwelling off Chile: Sapatial and temporal extrapolation using satellite data. Journal of Geophysical research. Vol. 107, no. 0.
14. Nwana H.S., Ndumu, D. T., Lee, L. C. and Collins J. C. (1999) ZEUS: A Toolkit for Building Distributed Multi-Agent Systems. Applied Artificial Intelligence Jounal, vol 1, n°13, pp. 129-185.
15. Odell, J., Levy R., and Nodine M. (2004) FIPA Modeling TC: Agent Class Superstructure Metamodel. FIPA meeting and interim work.
16. Odell, J. and Huget, M. P. (2003) FIPA Modeling: Interaction Diagrams.
17. Pokahr, A., Braubach, L. and Lamersdorf W. (2003) Jadex: Implementing a BDI-Infrastructure for JADE Agents, in: EXP - In Search of Innovation (Special Issue on JADE), Vol 3, Nr. 3, Telecom Italia Lab, Turin, Italy, September 2003, pp. 76-85.
18. Santamaría J. and Nieto J. (2000) Los agujeros del cambio climático. World Watch no. 12. pp 62-65.
19. Sarmiento J. L. and Dender M. (1994) Carbon biogeochemistry and climate change. Photosynthesis Research, Vol. 39, 209-234.
20. Takahashi T., Olafsson J., Goddard J. G., Chipman D. W. and Sutherland S. C. (1993) Seasonal Variation of CO2 and nutrients in the High-latitude surface oceans: a comparative study. Global biochemical Cycles. Vol. 7, no. 4. pp 843-878.
21. Wooldridge, M. and Jennings, N. R. (1995) Agent Theories, Architectures, and Languages: a Survey. In: Wooldridge and Jennings, editors, Intelligent Agents, Springer-Verlag, pp. 1-22.
22. Wooldridge, M. and Jennings, N. R. and Kinny, D. (2000) The Gaia Methodology for Agent-Oriented Analysis and Design. Journal of Autonomous Agents and Multi-Agent Systems, 3 (3). pp. 285-312.

Music Knowledge Analysis: Towards an Efficient Representation for Composition

Jesus L. Alvaro[1,2], Eduardo R. Miranda[2], and Beatriz Barros[1]

[1] Departamento de Lenguajes y Sistemas Informáticos, UNED
JesusLAlvaro@gmail.com
bbarros@lsi.uned.es
[2] Interdisciplinary Centre for Computer Music Research, University of Plymouth, UK
eduardo.miranda@plymouth.ac.uk

Abstract. This document presents an analysis of Music Knowledge as a first step towards music representation for composition. After an introductory review of music computing evolution, several approaches to music knowledge are described: the system levels context, music theory and disciplines, dimensions in music, and finally the creative process. Then, the composition knowledge is analyzed at the symbolic level, dissecting its sub-level structure, and concluding with some requirements for an efficient representation. EV meta-model is presented as a multilevel representation tool for event based systems as music. Its structure and unique features are described within the analyzed level context. Three musical application examples of EV modeling are shown in the field of sound synthesis and music composition. These examples test representation, extension and development features.

1 Computer Music

The use of computers in music composition was introduced fifty years ago. Since then, computers have played an active role in several aspects of music creation from complex sound synthesis to automatic generation of musical material. Computer music systems, in this period, have been greatly influenced by the paradigms in which they have been developed, conditioning their creative capabilities.

As a first paradigm, the traditional conception of the *score*, as a representation for performance, has influenced several systems. They have been supported by the *score-orchestra metaphor*. Systems like *Music V* and its successors *Csound* [7], *CLM* [4], clearly reveal such dissociation by separately defining score and orchestra. The score approach lacks some representativity in two aspects. Composition elements are absent, and the sonic final result depends excessively on the specific performer.

Many of these same systems, and some others like *PD* [3] were also influenced by the *UG paradigm,* or the architecture of the first analog synthesizers. The approach is really efficient in the sound synthesis field and real time performance but not so efficient representing music for human performance. Its signal processing approach imposes a rigid lineal time conception without allowing flexible multi-temporal structures.

Some systems like *Symbolic Composer* [5] and *CM* [6] have been based on the MIDI representation. The MIDI specification was a great advance for computer music in the eighties, because it simplifies the score as a piano keystroke sequence. This

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 331 – 341, 2006.

simplification reduces the computation requirements, but, on the other hand, it lacks some performance information such as articulations and dynamics. It is also a score type representation for an undefined orchestra.

As we have seen, the underlying Knowledge Representation will determine and limit the creative capabilities of a computer music system, so it is worth focusing on *Musical Knowledge* as the base steps of our computer music research.

2 Music Knowledge Approaches

A first step in approaching Music Knowledge could be trying to define what we understand by the term “Music”. The simple query "define:Music" at an internet searcher can show up to forty definitions for the term. Different approaches and descriptions can be found: music as the sound itself, the organization of sounds, a human activity, an art, a communication, something impossible to define... Let's analyze some of these approaches.

2.1 Music System Levels

In *"The Knowledge Level"* [2], Alen Newell describes the level structure of a system. One of the interesting properties of every level, as defined in the paper, is its independence from lower levels, so it is possible to work at one level without knowing the details of those below it. Newell also introduces the knowledge level as a zone which is immediately above the symbolic level with knowledge as its medium that can be defined or represented at the symbolic level. Figure 1 represents a possible translation of that level structure to the music domain.

KNOWLEDGE LEVEL	Emotions, Evocations Equilibrium, Continuity, Unity, Symmetry Development, Contrast, Surprise, Liveliness, Dynamism, Metaphor
SYMBOLIC LEVEL	Outlines, Elements, Procedures Composer Style, Music Theory, Harmony, Notation
IMPLEMENTATION LEVEL	Sound, Scores

Fig. 1. Levels in Music

At the implementation level, we could situate the resulting product of the composition process, that is the sound itself, or the music sheet ready for the performer.

The music message content is found at the knowledge level. It comprises what is transported by the music, and what is received by the listener. In this category, we can find components from the composer which are both conscious and sub-conscious. We can find animi states, emotions, evocations and other human components. But also more objective qualities can be found like equilibrium, continuity, development, contrast, surprise, unity, symmetry, liveliness, dynamism, metaphors...

The remaining music components should be placed at the symbolic level. That is everything that can be implemented in a music score, and can represent or transport the knowledge level. Some of the components at this important level are shown in Figure 1. A deeper analysis of the symbolic level will be presented in the next section.

2.2 Music Theory

According to Music Theory, music knowledge is something different than the knowledge level. It comprises disciplines like notation, harmony, counterpoint, music form, composition techniques like development or variation, orchestration, acoustics... In the level structure above, this music knowledge should be placed still at the symbolic level. The placement of this music knowledge or "composition knowledge" is a difficult task, a matter of philosophy and aesthetics discussion. In that sense, music is sometimes recognized as "pure music", an "always abstract art", some kind of artistic expression without represented content, and without material substance. According to this, *music would represent itself*, or in other words, the knowledge level of a music composition is the composition itself. This can be realized when analyzing some music compositions like Webern op.27, where equilibrium and beauty of the score architecture could be enough artistic content for the piece.

2.3 Music Dimensions

From the definition of music above, it can be deduced that pitch and time are essential dimensions of music. Other involved magnitudes are timber, dynamics, tempo and articulation. Undoubtedly, the most important dimension in music is the *time*. Every element in music is arranged in time, even sound has nothing to do without time. We can consider time as the horizontal dimension along which any other dimension is distributed. It can be easily understood looking at a music score as a cartesian representation with time as its horizontal axis. Rhythm, as the most elemental form of music, consists of the organizations of events in time. The importance of time is even more notorious when we realize that there are many time scales in music. From a microscopic point of view we can see the wave-shape of every sound note. From a macro view, the global musical form of the piece is perceived. Every intermediate depth has its significance in the music form such as notes, motives, phrases, sections, movements... An interesting property of the temporal multi-scale is the similarity of structures across scales; they share some kind of fractal recurrence. It is also important to observe, that the score deals with time in a discrete way. Time positions and durations are constrained to a grid of bars, beats and sub-beats.

Pitch is probably the second music dimension in order of importance. Vertically arranged in the score staff, it has to do with the height of the sounds, the fundamental perceived frequency. Like time magnitude, pitch is not usually considered as a linear continuum, but as a quantized form. The piano keyboard is a clear example of this, where practicable pitch values are represented by its associated key, ignoring any intermediate pitch between adjacent keys.

Dynamics is the musical name for the sound intensity dimension. It is probably the other main dimension in music. In the score is notated by symbols like "*pianissimo"* or "*mezzo forte"* and opening or closing hairpins indicating *crescendo* or *diminuendo*,

respectively. Other music dimensions like articulation or tempo should be considered in a music knowledge representation. See the references section for a further study.

Upon observing music magnitudes from the level structure described above, it is important to note how the time dimension remains stable across every level. Alternatively, other dimensions like pitch, timber or dynamics appear at some level as components representing other qualities. This property also supports the preponderance of time dimension to be considered in music representation.

2.4 The Creative Process

Composition process is more complex than a mere element association. It comprises some subprocesses in the composer world such us conception, abstraction, imagination, implementation, analysis and correction. Figure 2 is a representation of an analysis model of composition processes, shown as a cyclic process of subprocesses[1]. From a knowledge representation point of view, it is important to observe that the creative process is unique. Fortunately for arts but unfortunately for AI, no composer follows the same procedures or the same scheme. It is quite difficult to draw the frontier between the generality of the creative process and the particularity of the composer's style, language and technique.

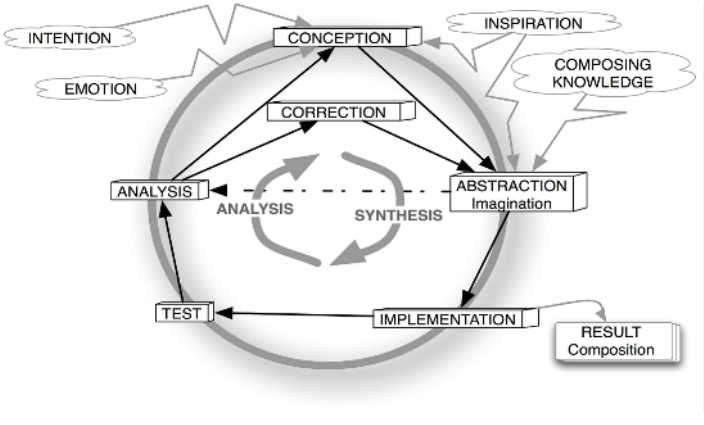


Fig. 2. Subprocesses cycle in composition

3 Composition Knowledge

Having analyzed the creation process, let's divide music knowledge into *music message* (at the knowledge level), and *composition knowledge* (at the symbolic level). It can be established that composition knowledge comprises elements, procedures, technique and strategies brought into play by the composer during the music creation process [1]. Not all elements have the same role in knowledge, but they should be ontologically organized into several layers. Elements, entities, and relationships occupy the bottom one; rules, constrains and axioms in the next one; procedures, rule-breaks, strategies, and some other meta-knowledge, occupy the next; and everything

equilibrated, at the top, by criteria and intentions[1]. For now, let's explore the level structure, in this search for an efficient representation.

3.1 The Symbolic Level

The symbolic level, hosts every representation of the composing knowledge. It is the level where the whole creative process takes place. Every component, from musical notes up to piece outlines, is placed inside this heterogeneous level. Under a detailed observation of this vast area, multiple sub-levels should be differentiated, in the sense of Newell's level definition. Every sub-level keeps its independence capability, and can be described or implemented by the sub-level immediately below.

Figure 3 shows a sub-level hierarchic structure of the *Music Symbolic Level.* At the top, and immediately below the knowledge level, there is a "*goal"* sub-level. This zone hosts top-level decisions about the piece, like composer objectives, intentions, global musical form, time schedule, climactic points, constrains from the both chosen music language and style, and also piece conception.

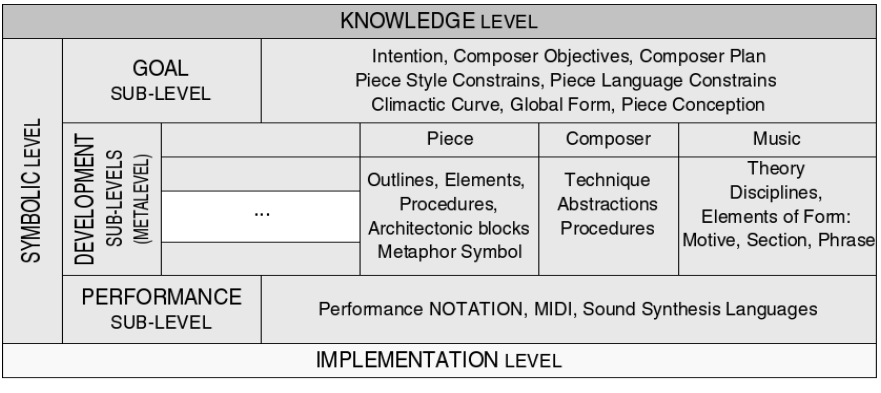


Fig. 3. Sub-levels in the Symbolic Level

At the bottom, and immediately above the implementation level, lies the "*performance"* sub-level. Performance representations, such as score notation, sound synthesis languages or even MIDI representation, can be placed in this sub-level. That is, any music representation ready to be interpreted by a performer. These representations are often recognized as the final result of the process, the composition, or the piece itself. The composition process can be seen as a trip inside the symbolic level, from the upper *goal* sub-level down to lower *performance* sub-level, passing across the region in between those extremes. This region is recognized in the *Metalevel* hypothesis [1]: "*Above performance level, there is a musical representation zone where composer usually works, and where it is possible to deal with efficient knowledge representations for computer music"*. Most of the abstraction of the composer, his technique resources, and the expertise of the music disciplines reside in the *Metalevel*. The metalevel could be also comprised of an undetermined number of sub-levels. All symbolic sub-levels share the following properties:

1. Every sub-level implies a higher degree of abstraction than the sub-level below.
2. Any representation at any sub-level can be translated into an equivalent representation at a lower sub-level. This compilation is also called *level development.*
3. Every sub-level shares the same time dimension.

3.2 Efficient Representation for Composition

At this point we are in a position for conjecturing about an efficient representation of Musical Knowledge for computer aided composition. -What features should it offer? -What design criteria should be applied? -What is the starting point? The following points of our *hypothesis for an efficient representation* for composition will try to answer these questions.

1. An efficient representation must be simple, but powerful enough to support the development of intelligent computer tools in a wide range of creativity, minimizing the limitations imposed by such representation.
2. Time dimension must be considered as the main magnitude, and must be flexibly managed at any level of depth, from micro-time to the music form.
3. It must be coherent and efficient, also at any level of representation, up from the performance level, and especially practicable at the *Metalevel,* close to *abstraction.*
4. It should contain an ontological substratum to accommodate conventional music entities, but flexible enough to easily incorporate new classes and relationships at higher sub-levels, when demanded by composer. Over this surface, it must be possible to represent higher layers of knowledge such as constrains, procedures and strategies, in a flexible manner.

4 EV: A Multilevel Representation

EV Meta-Model [1] is proposed as a basis for an efficient representation for composition. In the described context, EV tries to model the properties shared by every sub-level in the Symbolic Level. That is, EV is a multilevel approach whose goal is the modeling capability of time dimension based representations, at any level. It is constructed upon the principles of simplicity, recursion, flexibility and coherency.

4.1 MetaModel Structure

A general overview of the EV metamodel structure is shown in Figure 4. Three main areas are represented: the ontological core, auxiliary modules and extension slots. The core was described in detail in [1]. Let's resume three unique features of the core:

1. *Unified class approach*: every object in the system is a descendant of the same event class, so system properties can be defined in the event class definition. In addition, the event is also considered as an event container, so complex time structures can be recursively defined by single events.
2. *Liveliness character*: every parameter value in the system is, by default, a dynamic object with an evolving status. This "live" property is then transmitted to the represented music. Recursion is also present in dynamic objects in the sense that they are defined by means of simpler dynamic objects.

3. *Time relativity*: Events are provided with their own time management. That allows a time conception inside the "child-event", different from the time of its "mother-event".

Some auxiliary modules are also shown in Figure 4. The path module provides the interconnecting and referencing system. It allows any parameter of any event of the tree to be referenced by any other dynamic object. This feature expands the creative possibilities by representing relationships between any elements of the model.

The *dynamic object* module comprises the dynamic object defining system. It provides definition syntax, as well as an extendable base of predefined elementary dynamic objects.

The controlled random module provides support for the definition of dynamic objects with any random requirement. Several random distributions are provided. Every random cell status is recallable by the use of initialization seeds, so any pseudo-random behavior can be easily repeated.

Extension of the model is considered in three directions. They are represented by three slots at the top of Figure 4. Definition of new subclasses from the main event, allows customizations in the structure of events. Definition of behaviors by specializing methods extends the customizations of the level, and allows the representation of some knowledge in the procedural form. New definitions of either elementary or complex dynamic object types can be included in the base, expanding the creative possibilities.

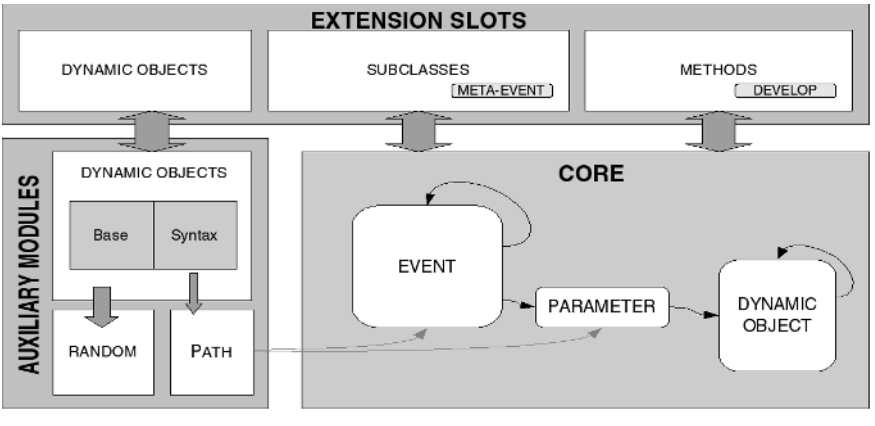


Fig. 4. EV MetaModel Structure

It is important to observe that extension can be carried out both in the metamodel or the modeled level itself. In other words, some extension could be incorporated to the metamodel, in order to be available for any other "EV model".

4.2 Level Development

As an example of a metamodel extension, the *MetaEvent* is defined as a subclass. It is the basis for the level development described below. The MetaEvent is characterized

by being an event with the capability of describing new events. It operates at a higher level, and again, it allows recursion. That is, the event defined by a metaevent, could be also a *metaevent*. The level development is achieved, by providing the metaevent with the "develop" method. Developing a metaevent means creating the events described by such *metaevent*. As a consequence, a translation from a higher level to a lower level is carried out. The development could be recursive if needed until the target lower level was reached.

There are two defined subclasses of *metaevent*: the *macro*, whose expansion is done at the definition time; and the *generator*, the standard metaevent whose expansion is performed at the event time.

An interesting type of generator for music is the *"sampler-generator"* subclass. Some parameters of this type of event are descriptions of the values for the parameters of generated events in the form of dynamic objects. These "child events" are generated by time sampling those dynamic objects, and collecting the samples for each parameter into a new event. There is a direct correspondence between the parameters of both generator and generated events. This way, a large quantity of events could be described with a high degree of control by a single event in a higher level.

5 EV Modeling

Most of the work in EV modeling is done by extension, which is by defining subclasses, methods and dynamic objects. In this section several cases are briefly reviewed, as practical examples of EV modeling. Some of them are explained, with more implementation detail, in reference [1]. They are reviewed here under the analytical approach described above. Three different examples are shown, each of them probing important aspects of *EV*. They all are music applied examples. Analytically, the first example is a level development, the second shows modeling of a preexisting level, and the third is a level development of a metaevent extended model, using the model of the second example, as its development target level.

5.1 Sound Synthesis from the Metalevel

Based on the sampler-generator subclass, *EVcsound* is an application demonstrating the level expansion capability of *EV*. Its purpose is to synthesize sound from high level expressions. Csound [7], both a sound synthesis language and a compiler, is used in this application at the lowest symbolic level. Its mission is sound synthesis, that is, the compilation of sound (at the implementation level), from a procedural description of a performer (.orc file) and a note event list (at the performance sub-level). Hence the target level of the experiment is the lowest sub-level in the symbolic level.

The music composition is an event-sequence instance with a list of child events of the subclass sampler-generator. Each sampler-generator develops into a list of new child events. Development ends when all child events are of the target sub-class csound-event. The compilation to the implementation level is then performed by csound. A practical example of the process is described in [1].

5.2 Notated Score Representation

- Could traditional notation be represented by EV? - Could such representation keep EV flexibility? - Could it be extended? - Could it flexibly support creative higher symbolic sub-levels? These are some guide questions for the design of *Evscore*, a traditional notation compatible representation. Built by defining subclasses from the main event class, it supports the representation of musical notes, articulations, text, dynamics, slurs, voices and bars in a flexible way. Every element in the score is a descendant from the event, so they inherit all properties and features of EV. As an example, although the notated time organization of bars and beats is represented, the time flexibility of EV is still available, allowing sequencer time management, real time spotting, or even a combination of them. The extendibility is applicable in order to represent complex pitch groups like trills, mordants, glissandi or arpeggios; and even some unconventional symbols used by composer. The EV approach of this representation also implies that scores can be written from definitions of requirements at a higher level.

EVscore is more than a test. One of the key-points of this representation and also the goal of its design, is to constitute the performance sub-level for other tools, a both solid and flexible basement over which higher levels of music representation can be successfully supported.

5.3 Extending the MetaModel: Modeling a Composing Procedure

The third example is built over *EVscore*, the notation representation described above. It constitutes a test for its design goal, and it also tests the metaevent extension capability of EV. *EVzone*, as the experiment is named, is a model for a composition procedure whose function is the compilation of melodies starting from the definition of some musical specifications for that melody. By using this model, the composer can try several melodies out directly from their musical specifications. In the context of the composition process model shown in Figure 2, the composer is liberated from the tedious way from abstraction to implementation, also simplifying both analysis and correction.

In *Evzone* implementation, an extension of the metamodel is carried out by defining two dynamic objects: tables and maps, and one special metaevent: the zone.

Table is an array of time sequenced value samples. Definition of a dynamic object by a set of snapshots is possible through this type of use. This is especially useful for the repeated use of either random or expensive calculation objects. In the example below, a table is used to store a random brownian shape. *Map* is used for representing recurrent structures by decomposing them into both constitutive elements and form pattern or symbolic map. This is a very musical feature based on how form is represented in the music community. The concept can be further extended by applying a grammar generator as a map, hence obtaining complex rich forms.

Zone is a metaevent subclass designed with musical form in mind. It intends to model form in a flexible manner, thus allowing multiple views of the same musical form. This is achieved by considering it not just a sequence of time zones, but a tree. The internal duration (trunk) of this metaevent is divided into zones (leaves) arranged

in a tree structure, the zone-tree. The rest of the parameters are expected to be defined also as a tree. A tree correspondence is then established between every parameter tree and the zone-tree, so every definition can address the corresponding zone. The same tree structure is not needed for parameters, just any tree structure allowing a correspondence is required. As an example, an atomic parameter definition would address the whole event. In addition to extra form flexibility, zone-tree provides the capability of a unified definition of several zones at the same time, thus allowing perception a refinement of zone relationships.

```
(deftable t1 256 (br 0 1 :seed .42723))          ;brownian shape in a table
(define-zone-event my-melody                  ;melody definition
    :zone-tree '((a b b c) ( a b c b c))         ;the form
    :ryth (mp zone '("+-++" "+-+-" "+---"))      ;rhythmic cells mapped into form
    :shape (mp '(a b) (list t1 (retrograde t1))) ; symmetry, map example
    :tessitura (tab t1 :min 0 :max 12)           ;step zone, scaled t1
    :ambitus (li i 5 8)                          ;melody span
    :pitchclass 'glydian                         ;the scale
    :step0pitch  'g4)
```

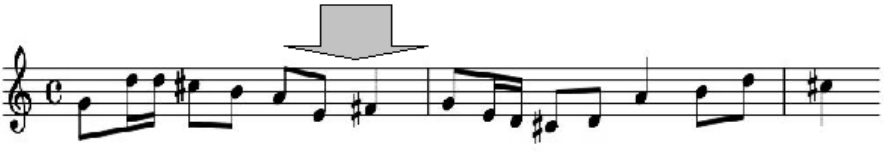


Fig. 5. Melody description example

Figure 5 shows both the musical definition, and the resultant notation of a simple melody generation example. The melody is represented by musical melodic properties like structural-form (zone), rhythm, contour (shape), tessitura, ambitus. Last two parameters, pitchclass and step0pitch, represent pitch constrains; in this case to the lydian scale rooted on G. Other constrains have been used at the implementation for optimizing scale step election. Deeper implementation details can be found in [1].

6 Conclusions

Because the creative potential of a music computer system is limited by the chosen representation, a previous deep analysis of music knowledge is a good starting point. The presented analysis is carried out in the context of system level concept. Music theory, music dimensions and the creative process, have been taken as reference points in this analysis.

Music knowledge is divided into *music message* at the knowledge level, and *composition knowledge* at the symbolic level. Composition knowledge is also comprised of several sub-levels spanning from *goal sub-level* down to *performance sub-level*. Composition process is a compilation or development of the *goal sub-level* into lower sub-levels. *Time*, the main dimension of music, remains present at every level. An efficient representation for composition should deal with time in a flexible way at any depth of scope. It must coherently support multiple levels.

EV meta-model is a representation of the multilevel nature of music composition. Constructed upon the principles of simplicity, recursion, flexibility and coherency, it sets the basis for music representation. EV modeling is performed by both extension and level development. As the given examples show, EV modeling has been found efficient in sound synthesis and music composition from high abstraction levels.

References

1. Alvaro, J.L. and Miranda, E.R. and Barros, B. (2005). *"EV: Multilevel Music Knowlegde Representation and Programming"*, in *Proceedings of SBCM, Belo Horizonte, Brazil.*
2. Newell, A. (1982). *"The Knowledge Level"*, Artificial Intelligence, vol. 18, pp. 87-127
3. Puckette, M. (1997). *"Pure Data"*, in *Proceedings of ICMC*, pp 224-227, Thessaloniki Greece
4. Schottstaedt, B. (2003). *"CLM Manual"*, Stanford, url: http://ccrma.stanford.edu/software/snd/snd/clm.html
5. Stone, P. (1997). *"Symbolic Composer"*, url: http://www.symboliccomposer.com
6. Taube, H.K. (1997). *"An Introduction to Common Music"*, in *Computer Music Journal*, Vol 21, No 1, pp 29-34
7. Vercoe, B.L. (1994). *"Csound: A Manual for the Audio-Processing System"*: MIT Media Lab

Mutual Information Based Measure for Image Content Characterization

Daniela Faur[1], Inge Gavat[1], and Mihai Datcu[2]

[1] Politehnica University Bucharest, 312 Splaiul Independentei 060042 Bucharest, Romania
dfaur@electro.masuri.pub.ro, igavat@alpha.imag.pub.ro
[2] German Aerospace Center DLR Oberpfaffenhofen, Germany
mihai.datcu@dlr.de

Abstract. An image can be decomposed into different elementary descriptors depending on the observer interest. Similar techniques as used to understand words, regarded as molecules, formed by combining atoms, are proposed to describe images based on their information content. In this paper, we use primitive feature extraction and clustering to code the image information content. Our purpose is to describe the complexity of the information based on the combinational profile of the clustered primitive features using entropic measures like mutual information and Kullback-Leibler divergence. The developed method is demonstrated to asses image complexity for further applications to improve Earth Observation image analysis for sustainable humanitarian crisis response in risk reduction.

1 Introduction

Anomaly detection in Earth Observation images refers to the detection of irregularity in the scene that appears unlikely according to a probabilistic or physical model of the scene, for example ammunition elements in a scene which is dominated by vegetation and soil. The image segmentation or classification into a set of structures or objects is a fundamental aspect for understanding the nature of the observed scene. Most of the work in the field is concentrating to solve the grouping of coherent or regular patterns in images. However, the existing methods do not cope with the heterogeneity of typical high resolution images of scenes subject of hazard. Thus, the present work will focus on the elaboration of models to asses anomalies and/or disorder in images. Image's anomaly regions can be regarded as very complex regions, different from the other regions or "regular" patterns composing the image.

An image can be partitioned into different objects, with different attributes depending on the observer interest and how carefully is observed. Similar techniques with the one used to locate words, regarded as molecules, formed by combining atoms, are proposed to describe images based on their information content. In this paper, we use primitive feature extraction, as spectral and texture signatures, and clustering to code the image information content. Another

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 342–349, 2006.

approach to describe image complexity is a method based on rate distortion analysis of the cluster space [1].

Our purpose is to describe the complexity of the information based on the combinational profile of the clustered primitive features using entropic measures. Using the Kolmogorov interpretation of entropy, a measure of image "complexity" is presented and evaluated to rank the degree of "disorder" of the image structures.

In order to prove the algorithms efficiency we use a remote sensing image presenting different spectral signature and a broad variety of structural information.

2 Primitive Image Features Extraction

In our approach we suggest an image content description with the use of spectral signatures and texture. The feature extraction split the image content into two different information channels. An unsupervised clustering is done for each channel as an information encoding and data reduction operation [2].

The image features reflect the physical parameters of the imaged scene. Spectral and image texture carry information about the structure of object surfaces. In the case of modeling high complexity signals, a large number of sources coexist within the same system. Thus, we use two models to describe the information source in the image. The models are likely to be analyzed hierarchically [3]. The hierarchical information representation is further presented and depicted in Fig.1:

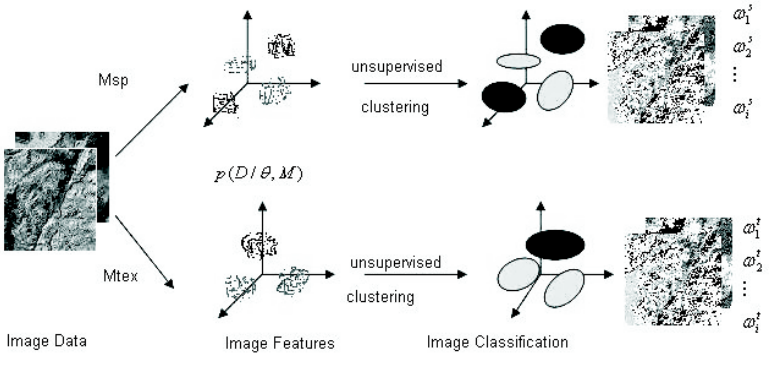


Fig. 1. Hierarchical modeling of image content. First primitive features θ are extracted from image data D based on two parametric signal models, e.g.spectral s and texture t. Using an unsupervised clustering across all images in the archive we obtain significant signal classes ω_i^s (spectral) and ω_i^t (texture) as content index.

At the level of image data information is content in raw data, the lowest level of information representation. For a quasi-complete characterization of the image content, information is extracted in the forms of parameters characterizing

spectral and texture as interactions among spatially distributed samples. The clustering is applied for each feature space, apart for all images in the archive. Thus, from image space I we obtain a space of clusters. Further, we use the dependencies between image space and clusters's space to determine image complexity in the archive.

2.1 Spectral Features

To form our data set we split a co-registered Daedalus image, of 12 spectral bands in visible and infrared, of size of 2752 x 883, and 1 m resolution, covering the land of Oberpfaffenhofen, Germany, in 16 images, of 300 x 300. The images indicate different spectral signatures and a broad variety of structural information (Fig.2)

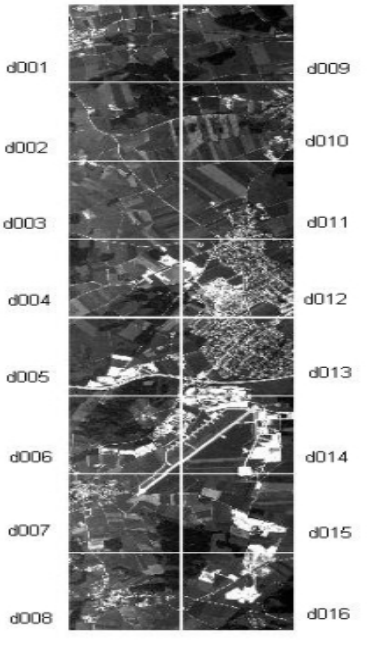


Fig. 2. Sixteen sub-images extracted from a Dadealus super-spectral image (Band 1)

2.2 Texture Features

To extract the image texture primitive features auto-binomial Gibbs Random Filed is used the as data model [4], [5], [11].

The Gibbs auto-binomial Random Field family of stochastic models assume that the statistics of the grey level of a pixel in the image depends only on the grey levels of the pixels belonging to a finite dimension neighborhood. The

probability of the grey level of the pixel x_s conditioned by its neighbourhood ∂x is:

$$p(x_s|\partial x_s, \theta) = \frac{1}{Z_s} \exp(-H(x_s|\partial x, \theta) \tag{1}$$

where Z_s is a normalization factor given by the sum of all possible states for the pixel x_s and θ is the model of the parameter vector. Assumptions have to be made for the functional form of the energy function H. In this study an auto-binomial model has been used, in which the energy function is:

$$H(x_s \mid \partial x_s, \theta) = -\ln(\frac{G}{x_s}) - x_s\eta \tag{2}$$

$$\eta = a + \sum_{i,j} b_{ij} \frac{x_{ij} + x_{ij}^{'}}{G}$$

where G represent the number of images's gray levels, each b_{ij} parameter describes the interaction between the pixel x_s and the pair x_{ij}, $x_{ij}^{'}$ (Fig.3) and the parameter a represents a sort of auto interaction.

X_{22}	X_{12}	X_{21}'
X_{11}	X_s	X_{11}'
X_{21}	X_{12}'	X_{22}'

Fig. 3. Image's pixel neighborhood

A fitting of the model on the image is performed, in order to obtain the best fitting parameters; for the estimation a conditional least square estimator was used:

$$\theta_{CLS} = argmin_\theta (x_s - \sum_{x_s=0}^{G} x_s p(x_s|\partial x_s, \theta))^2 \tag{3}$$

The evidence of the model, i.e. the probability of the model given the data, can be calculated by:

$$p(M|D) = \frac{p(D|M)p(M)}{p(D)} \tag{4}$$

where the probability of the data can be obtained via integration:

$$p(D|M) = \int p(D|\theta, M)p(\theta|M)d\theta \tag{5}$$

From the estimated parameters,we derive several features to describe the image content: the norm of the estimated parameters $|\widehat{\theta}|$ as the strength of texture,the estimate of the variance $\widehat{\sigma}_M^2$ and the evidence of the model M [6].

3 Coding Image Content and Assessment of Complexity

The image coding is effectuated by extracting its primitive features, i.e. color or spectral information, and texture parameters. Based on information theory, sustaining that data processing cannot increase information, each level in the hierarchical scheme is associated with a certain loss of information.

Image information content is reduced, compressed and the clusters are the ones who provide information. Proceeding on this idea and on the fact that pattern recognition requires clusters recognition in different ways, this paper proposes a complexity valuation method at cluster level based on Kolmogorov entropy. Kolmogorov generalized the notion of mutual information from information theory for the case of absolutely arbitrary continuous messages and signals [13]. Starting with the mutual information he defines the ε - entropy of a random object. His problem formulation was: "Suppose source information yields a magnitude X and another signal $X^{'}$ such that their joint probability density function is restricted to a certain family, dependent on the parameter ε". Kolmogorov gave an alternative interpretation to information theory from the algorithmic point of view. The Kolmogorov entropy is a measure of complexity of an object; an object being a natural number or other constructive objects which are computable feasible by natural numbers. In order to rank our images by their information content we use a clustering algorithm like K-means on our data set to obtain a general number of clusters on each particular image. Then we apply Kolmogorov entropy on clustered images for complexity valuation at cluster level.

From the generated content-index the mutual information between image space I and space of clusters ω can be computed as:

$$I(I,\omega) = p(\omega_i|I_k)p(I_k)\log\frac{p(\omega_i|I_k)}{p(\omega_i)} \tag{6}$$

where $p(\omega_i|I_k)$ indicates the posterior probabilities of signal classes given a certain image I_k from the archive. Prior probabilities for signal classes and images are given $p(\omega_i)$ and $p(I_k)$, respectively. For spectral model it is used ω_i^s and for texture ω_i^t. The measures indicate the information transmitted from image data through feature extraction and unsupervised content-index generation (clustering) to the space of clusters (Tabel 1). Note that the minimum information is given for spectral model.

The association between image space and class space can further be used to measure the complexity of images in the archive. Since the query performance of content-based image retrieval systems depends on the complexity of the data, analyzing the image database that is used for testing is rather important for

Table 1. The mutual information between image space I and space of clusters ω computed for the two models.The structural (texture) information dominates the spectral information.

Image	Mutual information $I(I,\omega)$
Spectral model	0.2230
GRF model	0.8804

evaluation. Similar to the method of [7] that applies information theory to determine the complexity of image databases, we measure the information between image space and class space based on Kullback -Leibler divergence. Compared to mutual information, Kullback -Leibler divergence can be applied to determine the complexity of a single image in the entire archive, being an objective measure. Kullback -Leibler divergence is given according to:

$$D(p,q) = \sum_{ji} p(\omega_j^s|I_k)p(\omega_i^t|I_k) \log \frac{p(\omega_j^s|I_k)p(\omega_i^t|I_k)}{q(\omega_j^s)q(\omega_i^t)} \tag{7}$$

and can be interpreted as how much a single image I_k is a typical mixture of the complete image content of the whole dataset.

Table 2. Kullback-Leibler divergence for image complexity determination in the data set. The complexity measure show that d_007 is an image containing much of the archive information content whereas the information offered by d_013 is quite different.

Image	KL divergence	Image	KL divergence
d_001	0.755	d_009	1.549
d_002	0.900	d_010	0.953
d_003	1.101	d_011	1.015
d_004	0.622	d_012	0.889
d_005	0.786	d_013	1.955
d_006	0.901	d_014	0.992
d_007	0.533	d_015	0.925
d_008	0.784	d_016	0.775

An unsupervised clustering algorithm like K-means was applied for each feature space, spectral and texture, across al images in the archive. Then, we use posterior probabilities of signal classes ω_j^s and ω_i^t given a certain image I_k in order to compute $D(p,q)$, a "distance measure" between two probabilities distributions. The results denote how much a single image I_k is a mixture of the entire archive information content. The most complex images from the sixteen studied are d_013 and d_009 because they have the highest value for KL divergence, denoting the facts that are really different from the information content of the data set. Of course, the image complexity depends on the ability of the

applied signal model to describe the image content and to capture characteristic image structures. In this case the results indicate that fine textures can influence the complexity.

4 Results and Discussions

This paper presents an assessment of information theory methods for image complexity characterization used in analysis of earth observation data for humanitarian crisis management. At the beginning information measures settled in information theory namely Shannon's entropy, mutual information and Kullback Leibler divergence are presented and then we applied them to evaluate information content in a sixteen image dataset.

One relative approach for image content characterization is to assume a particular source model generating an image and compute entropy of the image based on that model. It has been established that entropy can solve the problem of extracting a relevant summary of data, a compressed description that captures only relevant or meaningful information from image content. Mutual information between image space and class space can be regarded as an objective measure and indicates system's ability to describe the image content. The composition of image structures by combining clusters in the primitive image feature space is further used to consider the Kolmogorov approach. This approach might offer interesting applications in content based archives since it allows a ranking of the image regions according to the user subjective concept of complexity regarding color and texture.

Acknowledgements. Work in the frame of FP6-EC project -"Technology to Support Sustainable Humanitarian Crisis Management STREAM".

Daedalus superspectral image was provided for research purposes by Dr. Peter Strobl from the German Aerospace Center, DLR.

References

1. Iancu, C., Gavat., I., Datcu , M. : Image disorder characterization based on rate distortion, Proceedings of 11th Conference of the Spanish Association for Artificial Intelligence CAEPIA 2005, Santiago de Compostela, Spain, November, 16-18, 2005,
2. Daschiel, H., Datcu, M.: Image information mining - Exploration of Earth Observation archives, Geographica Helvetica 58, pp. 154-168, 2003
3. Datcu, M.,Daschiel, H.,Pelizzari, A., Quartulli, M., Galoppo, A., Colapicchioni, A., Pastori, M.,Seidel, K., Marchetti, P.G., D'Elia, S. : Information mining in remote sensing image archives -Part A: System Concepts, IEEE Trans on Geosciences and Remote Sensing , pp. 2923-2936, 2003
4. Datcu, M., Stoichescu, D.A., Seidel, K., Iorga, C.: Model fitting and model evidence for multiscale image texture analysis, American Institute of Physics, AIP Conference Proceedings 735, pp. 35-42, 2004
5. M. Schrder, M. Walessa, H. Rehrauer, K. Seidel and M. Datcu : Gibbs random field models: a toolbox for spatial information extraction, Computers and Geosciences 26, pp. 423-432, 2000

6. M. Datcu, K. Seidel, S. D'Elia and P. G. Marchetti, : Knowledge-driven Information Mining in Remote-Sensing Image Archives, ESA Bulletin 110, pp. 26-33, 2002
7. Schröder, M., Rehrauer, H., Seidel, K. and Datcu, M. : Interactive learning and probabilistic retrieval in remote sensing image archives, IEEE Trans. on Geosciences and Remote Sensing, vol. 38, no. 5, pp. 2288-2298, 2000
8. Gonzalez, R., Woods, R. : Digital Image Processing, Prentice Hall, 2002
9. Jain, R. : Unified access to universal knowledge: Next generation search experience, Ramesh Jain-White papers,2004.
10. Rao, A., Srihari, R. K. , Zhu, L. and Zhang, A. : A method for measuring the complexity of image databases, IEEE Trans. on Multimedia, vol. 40, no. 2, pp. 160-173, 2002.
11. Schröder, M., Rehrauer, H., Seidel, K. and Datcu, M : Spatial information retrieval from remote sensing images: Part II Gibbs Markov Random Field, IEEE Trans. on Geosciences and Remote Sensing, vol. 36, pp. 1446-1455, 1998
12. Shannon, L.E. : A mathematical theory of communication, Bell Systems Technical Journal, vol. 27, 1948
13. Shiryayev A.M. : Selected works of A.M. Kolmogorov, Academic Publishers, 1993
14. Smeulders A., Worring M, Santini S.Gupta, A. Jain : Content based image retrieval at the end of early years, IEEE Trans Pattern Anal Machine Intell vol. 22, no. 12, 2000
15. Spataru A. : Fondements de la thorie de la transmission de l'information. Lausanne: Presses Polytechniques Romandes,1987

Nonlinear Mappings with Cellular Neural Networks

J. Álvaro Fernández-Muñoz[1], Víctor M. Preciado-Díaz[2], and Miguel A. Jaramillo-Morán[1]

[1] Dpto. Electrónica e Ing. Electromecánica, Escuela de Ingenierías Industriales, Universidad de Extremadura, Avda. de Elvas, s/n, 06071, Badajoz (Spain)
{jalvarof, miguel}@unex.es, http://eii.unex.es/profesores
[2] Laboratory for Electromagnetic and Electronic Systems (LEES), Massachusetts Institute of Technology, Room 10-171, 77 Massachusetts Avenue, Cambridge, MA, 02139, USA
vmp@mit.edu, http://lees.mit.edu/lees/lees_students.htm

Abstract. In this paper, a general technique for automatically defining multi-layer Cellular Neural Networks to perform Chebyshev optimal piecewise linear approximations of nonlinear functions is proposed. First, a novel CNN cell output function is proposed. Its main goal is to control input and output dynamic ranges. Afterwards, this 2-layer CNN is further programmed to achieve generic piecewise Chevyshev polynomials that approximate a nonlinear function.

1 Introduction

Since their introduction in 1988, Cellular Neural Networks (CNN) [1] [2] have attracted research community's attention, mainly because their ability of integrating Digital Image Processing (DIP) tasks into compact, real-time programmable VLSI circuits [3]. In the past decade, a wide variety of digital processes over digital binary images have been developed, most of them making profit of the CNN's inherent nonlinear behaviour [2] [4] [5].

In the Grey-scale Image Processing context, CNN have participated in processes that include or are based on linear spatial filtering (i.e. discrete convolution) [6] [7]. However, in many cases it is necessary to perform nonlinear operations over images, enabling the redistribution of intensity content of each pixel within the image by means of a relation or function known as mapping. This is the case of a series of widely used techniques including image thresholding or binarization, automatic dynamic range expansion and compression (contrast stretching) and histogram equalization, to name a few [8] [9]. In addition, there are many situations in which a nonlinear (e.g. trigonometric, exponential, logarithmic) calculation is needed, e.g. to calculate image statistics that are associated to one pixel and its surroundings [10].

In this work, a method for automatically defining multi-layer CNN that approximate nonlinear mappings by means of piecewise linear functions with user-controlled accuracy is proposed. In the next paragraphs, an introduction of the main features of CNN and some of the common DIP tasks that can be defined through nonlinear mappings is performed. Then, the definition of optimal Chebyshev polynomials is covered in detail.

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 350 – 359, 2006.

In Section 2, the basic dynamic range control CNN is introduced. This network is the basis for the design of multi-layer CNN that performs the Chebyshev piecewise linear approximation of nonlinear functions, as detailed in Section 3. An application example of these two new CNN structures is presented in Section 4. Finally, a brief results summary and conclusion remarks are outlined in Section 5.

1.1 Cellular Neural Networks (CNN)

The original model introduced by Chua & Yang [1] is based upon the neural definition of a 2D discrete neural layer that is connected to the input plane (e. g. an image X with the same dimensions) and to the output plane, also of the same dimensions. This connection is performed by two templates, B and A, respectively. The B feed-forward template performs a spatial discrete convolution over the input X, while the feedback template A operates in the same way over the output Y. The outcomes of these two convolution operations are then stored in the so-called neural state matrix Z, whose value varies in time by means of the following differential equation:

$$C\frac{dz_{ij}(t)}{dt} = -\frac{1}{R}z_{ij}(t) + \sum_{C(k,l)\in N_r(i,j)}\left[A(i,j;k,l)\cdot y_{kl}(t) + B(i,j;k,l)\cdot x_{kl}\right] + I\ , \tag{1}$$

where C and R are values that control the transient response of the neuron (just like an RC filter), I is a constant value that biases the state matrix Z, and N_r is the local neighbourhood of each pixel (i, j) defined for a positive integer radius r as

$$N_r(i,j) = \left\{C(k,l)\middle|\max\left\{|k-i|,|l-j|\right\}\le r\right\}\ . \tag{2}$$

Finally, the relation between the state matrix Z and the output image Y can be defined through the following piecewise linear function

$$y_{ij} = \frac{1}{2}\left(|z_{ij}(t)+1| - |z_{ij}(t)-1|\right)\ . \tag{3}$$

1.2 Linear and Nonlinear Mappings in DIP

One of the basic criteria for classifying DIP techniques is the domain where they operate, namely the image or spatial domain and the transform domain (e. g. the Fourier domain). Spatial domain techniques are those who operate directly over the pixels within an image (e.g. its intensity level). A generic spatial operator can be defined by

$$Y(i,j) = T\left[X(i,j)\right]_{N_r}\ , \tag{4}$$

where X and Y are the input and output images, respectively, and T is a spatial operator defined over a neighbourhood N_r around each pixel (i, j). Based on this neighbourhood, spatial operators can be grouped in two types: Single Point Processing Operators, also known as Mapping Operators, and Local Processing Operators, which can be defined by a spatial filter (i. e. 2D-discrete convolution) mask [9].

The simplest form of T is obtained when N_r is 1 pixel size. In this case, Y only depends of the intensity value of X for every pixel and T becomes an intensity level transformation function, or mapping, of the form

$$s = T(r), \tag{5}$$

where, for the sake of simplicity in the notation, r and s are variables that represents grey level in X and Y, respectively, for each pixel coordinate (i, j).

According to this formulation, mappings can be achieved by direct application of a function over a range of input intensity levels. By properly choosing the form of T, a number of effects can be obtained, as the grey-level inversion, the compression or expansion of the dynamic range (i. e. contrast enhancement), and threshold binarization for obtaining binary masks used in analysis and morphologic image processing.

A mapping is linear if its mapping function T is also linear. Otherwise, T is not linear and the mapping is also nonlinear. An example of nonlinear mapping is the CNN output function (**3**). This mapping consists of three linear segments: two saturated levels -1 and +1, and the central linear segment with unitary slope that connects them. This function is known as a piecewise linear function. It performs a mapping of intensity values stored in Z in the [-1,+1] range. The bias I controls the average point of the input range, where the output function gives a zero-valued outcome.

Other well-known nonlinear mappings are the absolute value function, the unit step, the exponential or power and logarithmic or root, and any combination of these functions. It is worth remarking that the main research approaches to the definition of nonlinear and complex piecewise linear functions in CNN need the use of nonlinear templates that are cyclically implemented over the CNN Universal Machine (CNN-UM) [11] [12]. The method proposed in this work is based on a more simple and modular network that performs Chebysev piecewise linear approximations.

1.3 Chebyshev Linear Approximations

Chevyshev's method is one of the most powerful techniques for defining piecewise linear approximations of nonlinear mappings $f(x)$. It is based on the division of a nonlinear function in connected linear segments, i.e. a piecewise linear function, that are restricted to some error ε. In the infinite norm (i.e. maximum absolute deviation) case, this error function takes the following form:

$$\varepsilon = \max\left|f(x) - f_{PW}(x)\right|, \tag{6}$$

where $f(x)$ is the nonlinear function and $f_{PW}(x)$ is the piecewise linear function that approximates it. This function can be defined as a combination of first order polynomials with coefficients $\{p_i, c_i\}$, i=1,2,…,K, which represent each of the K linear segments that are needed to approximate $f(x)$ with a maximum error (**6**) in the $[x_0, x_K]$ range. Thus, $f_{PW}(x)$ can be expressed by the following equation:

$$f_{PW}(x) = \begin{cases} p_1 x + c_1 & x_0 \le x \le x_1 \\ p_2 x + c_2 & x_1 \le x \le x_2 \\ \vdots & \vdots \\ p_K x + c_K & x_{K-1} \le x \le x_K \end{cases}. \tag{7}$$

The first order polynomial that minimizes the error ε in a range of x is called a Chebyshev polynomial, and can be established by means of the error property theorem [13]. This theorem states that a n-order polynomial P_n minimizes the infinite norm with regard to a function $f(x)$ in the range $[x_a, x_b]$ if $|f(x)-P_n|$ takes the value of the infinite norm in at least $n+2$ points within the range $[x_a, x_b]$ with alternate sign.

For the special case of linear approximation of a function $f(x)$, in order to found an interval in which three points x_a, x_b and $x_i \in [x_a, x_b]$ that meet the maximum error criteria ε, with the additional constraint of the second derivative of $f(x)$ maintaining its sign over the full interval $[x_a, x_b]$, the following procedure can be applied:

Starting from a known x_a, x_b will be such that if

$$p = \frac{f(x_b) - f(x_a)}{x_b - x_a} \tag{8}$$

is the slope of the straight line segment defined by the polynomial

$$P_1(x) = px + c\,, \tag{9}$$

with

$$c = f(x_a) - px_a\,, \tag{10}$$

then, x_i will be such that

$$f'(x_i) = p\,, \tag{11}$$

meeting the following condition:

$$\left|P_1(x_i) - f(x_i)\right| = 2\varepsilon\ . \tag{12}$$

The Chebyshev polynomial that approximates $f(x)$ in the interval $[x_a, x_b]$ will be

$$Q_1(x) = \begin{cases} P_1(x) - \varepsilon & f''(x) > 0; \forall x \in [x_a, x_b] \\ P_1(x) + \varepsilon & f''(x) < 0; \forall x \in [x_a, x_b] \end{cases}. \tag{13}$$

The suitable sign for the definition of $Q_1(x)$ is determined by the sign of the second derivative of $f(x)$ in the closed interval $[x_a, x_b]$. The polynomial $Q_1(x)$ guarantees a maximum error approximation ε for both limit points x_a and x_b and the intermediate point x_i derived from (11).

Thus, the set of polynomials $Q_i(x)$ with coefficients $\{p_i, c_i\}$, $i=1,2,\ldots,K$, that approximate a nonlinear function $f(x)$ by means of the piecewise linear function $f_{PW}(x)$ defined in (7) can be obtained by successively applying the proposed algorithm.

2 Multi-layer CNN for Dynamic Range Control

Starting from the original CNN cell or neuron, it is needed to obtain a piecewise linear function, with input range $[m-d, m+d]$ and output range $[a, b]$. In other words, a modification of the generic CNN dynamic input and output ranges is demanded in

order to be able to implement any of the linear stages defined in Section 1.3 for the Chebyshev polynomial approximation.

Furthermore, this piecewise linear mapping function must be of the form

$$T[X(i,j)] = \begin{cases} a & -\infty < X(i,j) \le m-d \\ \dfrac{b-a}{2d}\left(X(i,j)-m\right)+\dfrac{b+a}{2} & m-d < X(i,j) \le m+d \\ b & m+d < X(i,j) < +\infty \end{cases} . \tag{14}$$

To be able to implement this function in a multi-layer CNN, the following constraints must be met:

$$|b-a| \le 2\,,\ d \le 1\ . \tag{15}$$

A CNN cell which controls the desired input range can be defined with the following parameters:

$$B_1 = \frac{1}{d}\,,\ A_1 = 0\,,\ I_1 = -\frac{m}{d}\ . \tag{16}$$

This network performs a linear mapping between [*m-d*, *m+d*] and [-1, +1], with zero average value. The output of the network is the input for a second CNN whose parameters are:

$$B_2 = \frac{b-a}{2}\,,\ A_2 = 0\,,\ I_2 = \frac{b+a}{2}\ . \tag{17}$$

The output of this second network is exactly the mapping *T* defined in (14) limited by the constraint (15). In the next section, this basic 2-layer CNN will be employed for defining Chebyshev piecewise linear mapping multi-layer CNN.

3 Multi-layer CNN for Chebyshev Linear Approximation

For achieving a CNN-based Chebyshev piecewise linear approximation, it is more suitable to transform the polynomial definition of $f_{PW}(x)$ into the following expression:

$$Q_i(x) = y_i(x) = y_i\left(x_a^i\right) + p_i\left(x - x_a^i\right), \tag{18}$$

Here, $y_i(x)$ is the straight line that corresponds to the Chebyshev polynomial $Q_i(x)$ in the *xy*-plane, x^i_a is the lower limit of the *i*-th interval and p_i is the slope of this interval, as defined in (8).

Using this new formulation, a 2-layer CNN with piecewise output function (3) is defined for each interval *i*, in order to achieve the mapping function *T* of (14). The CNN parameters m_i, d_i, a_i and b_i that establish this function for the *i*-th interval can be obtained by the following relations:

$$a_i = Q_i\left(x_a^i\right) = y_a^i\,,\ b_i = Q_i\left(x_b^i\right) = y_b^i\,,\ m_i = \frac{x_b^i + x_a^i}{2}\,,\ d_i = \frac{x_b^i - x_a^i}{2}\ . \tag{19}$$

Using these parameters, the definition of the 2-layer CNN for dynamic range control introduced in the previous section by (16) and (17) follows. In this case, these equations become:

$$B_1^i = \frac{1}{d_i}\,,\ A_1^i = 0\,,\ I_1^i = -\frac{m_i}{d_i}\,, \tag{20}$$

$$B_2^i = \frac{b_i - a_i}{2}\,,\ A_2^i = 0\,,\ I_2^i = \frac{b_i + a_i}{2} - I_{\text{var}}^i\,, \tag{21}$$

for the i-th interval, where

$$I_{\text{var}}^i = \begin{cases} 0 & i = 1 \\ y_a^i & 1 < i < K \end{cases} \tag{22}$$

takes into account the average point or bias shift needed for the output function to correct the accumulative addition of the saturation outputs from the i-1 previous CNN pairs.

Now, recalling that the output function (3) of each CNN is a piecewise linear function with maximum dynamic range [-1, +1] and that the final output must be the pixel by pixel addition of the K double CNN (20) and (21), this multi-layer CNN will achieve the linear approximation of $f(x)$ if the constraint defined by (15) is met for each layer. This condition now can be expressed by

$$\left|y_b^i - y_a^i\right| \le 2\,,\ \frac{x_b^i - x_a^i}{2} \le 1\,, \tag{23}$$

for i=1,2,…,K, where K is the number of intervals. In addition, it is also necessary that

$$\left|y_a^1\right| \le 1\,. \tag{24}$$

If this constraint is not fulfilled, the first layer will unbalance the average point of the combined network output function. If this happens, it will be necessary to define n additional networks, where n is the minimum integer such that

$$\left|y_a^1\right| < n\,, \tag{25}$$

with the following network parameters:

$$B_j = 1\,,\ A_j = 0\,,\ I_j = x_a^1 + j\,, \tag{26}$$

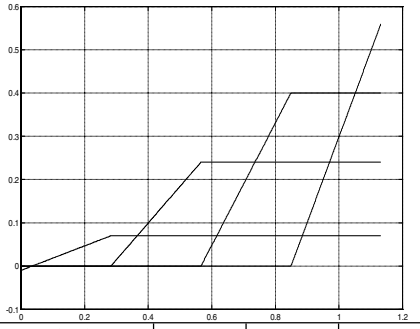
with j=1,2,…,n. The aim of these additional networks is to prepare the beginning point of the first linear stage of the approximation for meeting the constraint defined

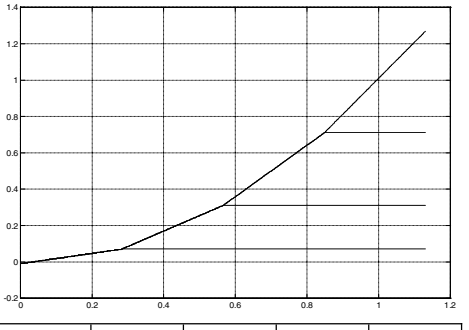
in (24). To do so, the next modification in the bias parameter of the second layer of the K approximating networks must be performed:

$$I_{\text{var}}^{i} = \begin{cases} n & i = 1 \\ y_a^i + n & 1 < i < K \end{cases} . \tag{27}$$

The final output of the multi-layer network will be the addition of each of the K piecewise linear output functions that operate over every pixel within the input image. Thus, this network performs piecewise linear mappings $f_{PW}(x)$ over intensity images.

As an example of this method, the square mapping $f(x) = x^2$ in the interval [0, 1] with maximum error $\varepsilon = 0.01$ is shown in Fig. 1. The Chebyshev polynomial coefficients $\{p_i, c_i\}$ and the resultant eight CNN parameters are also included. Note that there is no need for additional networks, provided that condition (24) is fulfilled.





$[x_a^i, x_b^i]$	p_i	c_i	y_a^i	y_b^i	B_1^i	I_1^i	B_2^i	I_2^i
[0.000 , 0.283]	0.283	-0.010	-0.010	0.070	7.067	-1.000	0.040	0.030
[0.283 , 0.566]	0.849	0.170	0.070	0.310	7.067	-3.000	0.120	0.120
[0.566 , 0.849]	1.415	-0.490	0.310	0.710	7.067	-5.000	0.200	0.200
[0.849 , 1.132]	1.981	-0.971	0.710	1.271	7.067	-7.000	0.280	0.280

Fig. 1. Chebyshev piecewise linear approximation of $f(x) = x^2$ in the interval [0, 1] obtained by an 8-layer CNN, with maximum error $\varepsilon = 0.01$. From top to bottom: outputs of each of the 4 dynamic range control CNN; combined output piecewise linear mapping $f_{PW}(x)$, and data table with intervals and coefficients for $f_{PW}(x)$ and subsequent network parameters.

4 Contrast Enhancement with Multi-layer CNN

One of the simplest techniques used in grey-scale image contrast enhancement is contrast stretching or normalization. This technique maximizes the dynamic range of the intensity levels within the image from suitable estimates of the maximum and minimum intensity values [9]. Thus, in the case of normalized grey-scale images, where the minimum (i.e. black) and maximum (i.e. white) intensity levels are represented by 0 and 1 values, respectively; if such an image with dynamic intensity range $[f, g] \subseteq [0,1]$ is fed in the input of the 2-layer CNN defined by (16) and (17), the following parameters will achieve the desired linear dynamic range maximization:

$$a = 0\,,\ b = 1\,,\ m = \frac{g+f}{2}\,,\ d = \frac{g-f}{2}\ . \tag{28}$$

Another more sophisticated contrast enhancement technique is the so-called histogram equalization. This method can be modelled by means of a nonlinear strictly growing mapping (i.e. the cumulative normalized histogram) which reallocates the intensity levels within the image in such a way that the output image features an approximately flat histogram [8].

In the discrete version of this technique, the output image only contains $L < N$ different intensity levels. This generally results in an output image with less quality than the input. However, a better equalization will be obtained if a greater number of output levels L are used in the calculations, but the procedure will always consume more CPU time [9].

A more favourable alternative may be to construct a piecewise linear mapping $f_{PW}(x)$ which interpolates each one of the intervals considered for the L-level normalized cumulative histogram. Using the line segment notation of (18), the definition of the L+1 segments in the interval [0, 1] that determine $f_{PW}(x)$ can be achieved by

$$y_1(x) = 2Lc_1 \cdot x \qquad 0 \le x < \frac{1}{2L} \tag{29}$$

$$y_k(x) = c_{k-1} + L(c_k - c_{k-1}) \cdot \left(x - \frac{2k-3}{2L} \right) \qquad \frac{2k-3}{2L} \le x < \frac{2k-1}{2L} \tag{30}$$

$$y_{L+1}(x) = c_L + 2L(1-c_L) \cdot \left(x - \frac{2L-1}{2L} \right) \qquad \frac{2L-1}{2L} \le x < 1 \tag{31}$$

where k=2,…,L and

$$c_k = \sum_{j=1}^{k} h_j\ , \tag{32}$$

for the normalized histogram $H_L = \{h_j\}$ and j=1,…,L.

These line segments can be obtained by a multi-layer CNN using the relations developed in Section 3 for the Chebyshev mappings, by properly assigning each of the L+1 segments to a dynamic range 2-layer CNN. The output image will be obtained as a combination (i. e. addition) of the outputs of each CNN pair.

An example of these two contrast enhancement techniques is shown in Fig. 2 for the normalized image *Model*, with 459x404 pixels and N=256 intensity levels. From its histogram, the global minimum f and maximum g intensity levels are obtained. Then, the image is processed by a 2-layer CNN for linear dynamic range maximization (i. e. contrast stretching) with the parameters defined in (28).

Comparing the result of this technique with that obtained from the interpolation of the cumulative normalized 16-level histogram, it can be noticed that the second technique performs a rougher, nonlinear mapping over the intensity levels than the linear

one. However, the output image reveals texture details that are not visible in the input neither in the contrast-stretched image. In addition, the image quality is higher compared with that obtained from the digital classic algorithm, because it contains in this example 131 distinct grey levels instead of standard L=16 levels.

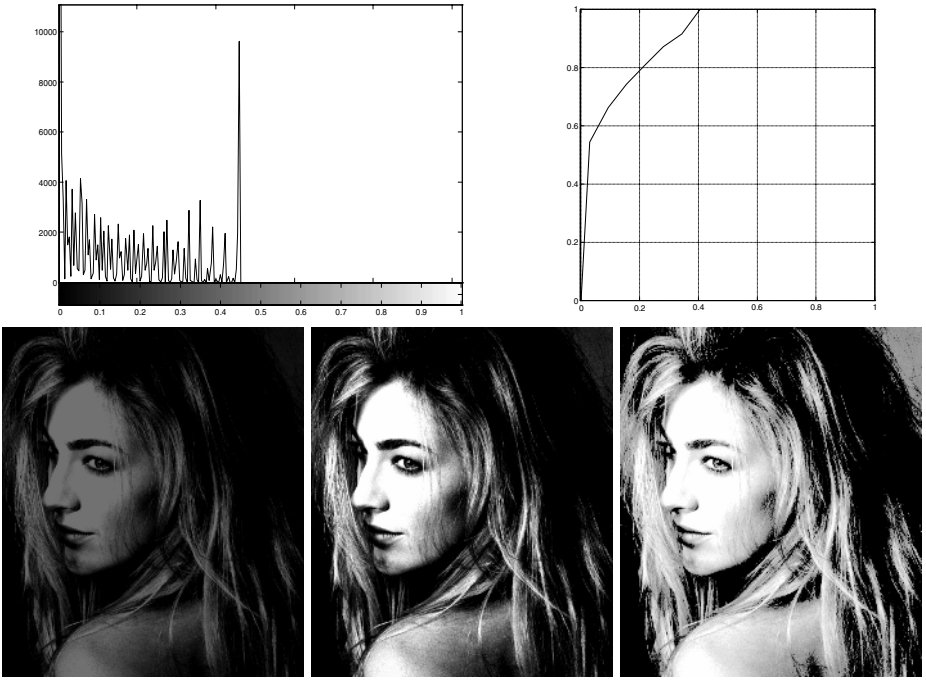


Fig. 2. Contrast enhancement over image *Model*, 459x404 (8 bits). From left to right, and top to bottom: interpolated histogram and cumulative normalized histogram with L=16 levels; original image, normalized output image from a dynamic range CNN with global maximum and minimum values; and output image from multi-layer Chebyshev CNN for histogram equalization by means of linear interpolation of the discrete cumulative normalized histogram.

5 Conclusion

A new CNN structure for approximating nonlinear functions with definable maximum error has been introduced. This network is composed of several 2-layer CNN blocks that perform a dynamic range control in both input and output, allowing the definition of Chebyshev optimal piecewise linear functions. In addition, its usefulness in contrast enhancement image processing has been demonstrated, both in linear (contrast stretching) and nonlinear processing (histogram equalization).

The proposed method overcomes the CNN inherent limitations found in literature when nonlinear calculations were about to be made on a parallel image processing structure [11] [12].

Because of its generic kind, it allows a wide variety of applications other than DIP, being its main virtue an efficient use of neurons and connections, which entails an easy implementation in programmable circuits based on the CNN-UM.

References

1. Chua, L.O., Yang, L.: Cellular Neural Networks: Theory. IEEE Trans. Circ. Syst., Vol. 35, nº 10 (1988) 1257–1272
2. Chua, L.O., Yang, L.: Cellular Neural Networks: Applications. IEEE Trans. Circ. Syst., Vol. 35, nº 10 (1988) 1273–1290
3. Rodríguez, A., Linán, G., Carranza, L., Roca, E., Carmona, R., Jiménez, F., Domínguez, R., Espejo, S.: ACE16k: The Third Generation of Mixed-Signal SIMD-CNN ACE Chips Toward VSoCs. IEEE Trans. Circ. Syst. I: Reg. Papers, Vol. 51, nº 5 (2004) 851-863
4. Matsumoto, T., Chua, L.O., Furukawa, R.: CNN Cloning Template: Hole-Filler. IEEE Trans. Circ. Syst., Vol. 37, nº 5 (1990) 635-638
5. Martinelli, G., Perfetti, R.: Generalized Cellular Neural Network for Novelty Detection. IEEE Trans. Circ. Syst. I: Fund. Theory Appl., Vol. 41, nº 2 (1994) 187-190
6. Chua, L.O., Roska, T.: The CNN Paradigm. IEEE Trans. Circ. Syst. I: Fund. Theory Appl., Vol. 40, nº 3 (1993) 147-156
7. Jaramillo, M.A., Fernández, J.A.: Adaptive Adjustment of the CNN Output Function to Obtain Contrast Enhancement. IWANN ’99 Proc. LNCS 1607 (II), Springer Verlag (1999) 412-421
8. Jain, A.K.: Fundamentals of Digital Image Processing. Prentice-Hall, Englewood Cliffs, NJ, (1989)
9. Fisher, R., Perkins, S., Walker, A., Wolfart, E.: Hypermedia Image Processing Reference (HIPR2). Web: *http://www.dai.ed.ac.uk/HIPR2*, University of Edinburgh, UK (2004)
10. Glaseby, C.A.: An Analysis of Histogram Based Thresholding Algorithms. Graph. Models Imag. Proc., Vol. 55 (1993) 532-537
11. Ter Brugge, M.H., Nijhuis, J.A.G., Spaanenburg, L.: Efficient DT-CNN Implementations for Large-neighborhood Functions. Proc. CNNA ‘98, London, UK (1998) 88-93.
12. Kék, L., Zarándy, A.: Implementation of Large Neighborhood Non-Linear Templates on the CNN Universal Machine. Intl. J. Circuit Theory Appl., Vol. 26 (1998) 551-566
13. Burden, R.L., Douglas, J.: Numerical Analysis. PWS Publishing Co., New York (1993)

On the Use of Entropy Series for Fade Detection

José San Pedro[1], Sergio Domínguez[1], and Nicolas Denis[2]

[1] DISAM - ETS Ingenieros Industriales - Universidad Politécnica de Madrid
C/José Gutierrez Abascal, 2 - 28006 Madrid, Spain
http://www.disam.upm.es/vision
[2] Omnividea Multimedia
http://www.omnividea.com

Abstract. Accurate shot boundary detection techniques have been an important research topic in the last decade. Such interest is motivated by the fact that segmentation of a video stream is the first step towards video content analysis and content-based video browsing and retrieval. In this paper, we present a new algorithm mainly focused on the detection of fades using a non-common feature in previous work: entropy, a scalar representation of the amount of information conveyed by each video frame. A survey of the properties of this feature is first provided, where authors show that the pattern this series exhibits when fades occur is clear and consistent. It does not depend on the monochrome color used to fade and, in addition, it is able to deal with on-screen text that sometimes remain in the monochrome stage. A statistical model-based algorithm to detect fades is proposed. Due to the clear pattern shown by fades in the entropy series and the accurate mathematical model used, motion and illumination changes do not severely affect precision as it normally happens with algorithms dealing with the detection of gradual transitions.

1 Introduction

Video analysis algorithms commonly rely in results from a series of stages that try to extract structure and meaning from the huge amount of data contained in a clip. Shot boundary detection is often the first of such stages. Its goal is to split the video stream into a series of segments captured in a single camera's "record and stop" operation. These segments, also known as shots, are joined together during the edition of the video using different types of effects: from simple sharp cuts to gradual transitions, as fades or dissolves. This pre-processing stage, if performed accurately, helps to reach high level video interpretation by examining the relationship between shots and transitions.

The detection of transition effects between shots requires specialized algorithms, depending on their type, to achieve good performance and precision. Hard cuts represent normally over 80% of the effects in a video. They are characterized by an intense and short deviation in the time series of most low level features in the video, which makes its detection easy and fast. A big number of

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 360–369, 2006.

works take advantage of the analysis of different features in the time series to detect hard cuts [1, 2].

Gradual transitions, such as dissolves and fades, produce patterns sometimes similar to those of moderate motion or lighting changes, which makes their accurate detection harder to achieve.

Meng *et al.* [3] present an algorithm where gradual transitions are ascertained from peaks and spikes in luminance variance series, using a threshold based on the assumption that neighboring scenes are independent. However, a certain correlation often exists between different scenes, thereby affecting the detection of peaks and spikes. In [4], an adaptive dissolve detection method based on the analysis of a dissolve modeling error is proposed. After candidates are extracted, they are verified based on a dissolve modeling error of the parabolic variance pattern. Enhancements to gradual transition detection algorithms are proposed by Truong *et al.* in [5]. Instead of selecting thresholds based on the traditional trial-and-error approach, adaptive thresholds are derived analytically from the mathematical model of transitions. Large negative spikes in the second derivative curve of luminance variance should appear at the beginning of a fade-out and at the end of a fade-in. These spikes will be taken as the boundaries of possible fades, and the first derivative curve of the mean luminance is studied in the values between the spike and the monochrome frames, to discard false detections.

Campisi *et al.* [6] make use of statistical methods to detect transitions. They present a computational inexpensive correlation-based algorithm for dissolve and fade detection. Fades and dissolves are gradual transitions that change linearly: differences between frames are a constant during the effect. The difference between consecutive frames is computed and subdivided in W blocks. A set of W correlation coefficients is then computed by comparing appropriate blocks of consecutive frame differences. A global correlation value is evaluated and compared to a threshold to decide whether it is a transition or not. A similar method is used by Han *et al.* using classical statistics to find the correlation between two subsequent frame differences [7].

In this paper, we use frame entropy as a measure to improve fade detection. Entropy is a measurement of the average number of bits needed to code some information. Previous work using entropy in video segmentation algorithms is not common. Cernekova *et al.* uses mutual information and joint entropy series to segment video and extract key frames [8]. Entropy is extracted from frame differences giving a measurement of the amount of information changing between successive frames. In this paper we consider entropy in the scope of the color distribution of each frame, which provides a scalar value representing the amount of information contained in every frame. Time series produced by mean luminance or variance tend to be more difficult to analyze and are more likely to produce false detections than entropy time series. In section 2, we introduce entropy as a generic measure of information and also as a measure of color information in images. A theoretical study of the behavior of frame entropy in the time series of a video is also presented, focusing in the pattern produced by fades. Section 3 describes a new algorithm to detect fades using frame entropy

series. Mathematical models introduced in Section 2 are used to create a model-based approach to detection using correlation. Section 4 introduces the results obtained by the detector along with an analysis of the effects that could decrease recall and precision. In Section 5, we present some conclusions about the quality of this new detector.

2 Entropy

2.1 Mathematical Background and Relation to Image Analysis

Given a random variable, X, which takes on a finite set of values according to a probability distribution $p(X)$, its entropy is defined as:

$$H(X) = -\sum_{i=1}^{n} p(x_i) \log p(x_i) \tag{1}$$

Shannon's entropy is the average amount of information contained in random variable X. It is also the uncertainty removed after the actual outcome of X is revealed. The Shannon's function is based on the concept that the information gain from an event is inversely related to its probability of occurrence.

This paper is focused in the analysis of images. Equation 1 can be used to compute the entropy of an image, I. Consider X a random variable representing the luminance intensity of I. Consider also I_k, the number of pixels in I with a luminance level of k, where $0 \leq k \leq N-1$ and N the number of bins. We can then define

$$p_I(k) = \frac{I_k}{\sum_{j=0}^{N-1} I_j} \tag{2}$$

as the probability of X being color k. This probability can be easily obtained from the histogram of I. Considering $h_I(k)$ to be the histogram value for luminance level k, then $h_I(k) = I_k$, and it follows:

$$p_I(k) = \frac{h_I(k)}{\sum_{j=0}^{N-1} h_I(j)} \tag{3}$$

$$H(X) = -\sum_{k=0}^{N-1} p_I(k) \log(p_I(k)) \tag{4}$$

As entropy can be seen as the uncertainty removed after the actual outcome of X is revealed, it follows that its value increases along with the width of the histogram. Thus, maximum entropy is reached when $p_I(c_i) = \frac{1}{N}$ while minimum is reached when all the pixels are within bin k, so $p_I(c_k) = 1$. This relation of the entropy with the color distribution of the image lacks an spatial component; its value is the same after applying any transformation to the image that keeps its histogram invariant. This behavior can be use to extract information from a sequence of images or frames by observing the time series of the entropy values.

2.2 Video Sequence Entropy Time Series

Equation 4 provides a measurement of the luminance complexity of a frame in a sequence. Computing this value for the complete sequence of frames gives us a time series that can be analyzed to search for certain patterns. Figure 1(a) shows a typical scenario in an entropy time series. It can be seen how frames from the same shot produce a smooth curve (sometimes a constant value, if few motion is present) in the series (frames 0 to 22). When a hard-cut occurs, the series will normally change abruptly (Frame 23). This is caused by the different luminance distribution between frames in different shots. Note that, although sometimes it is very clear in the series where a cut exists, histogram differences are a more reliable method to detect them: different histograms can lead to the same entropy value, causing a missed boundary.

When the frame sequence includes a fade, a well defined pattern appears in the entropy time series. A fade can be modeled with the following equations:

$$T_{f_i} = \alpha(t)M + (1 - \alpha(t))I(t) \tag{5}$$

$$T_{f_o} = \alpha(t)I(t) + (1 - \alpha(t))M \tag{6}$$

where $\alpha(t)$ is a transformation function that is usually a linear function and $t \in [0, Time-1]$ where $Time$ represents the duration of the fade. M is a monochrome frame and $I(t)$ is the frame at time t.

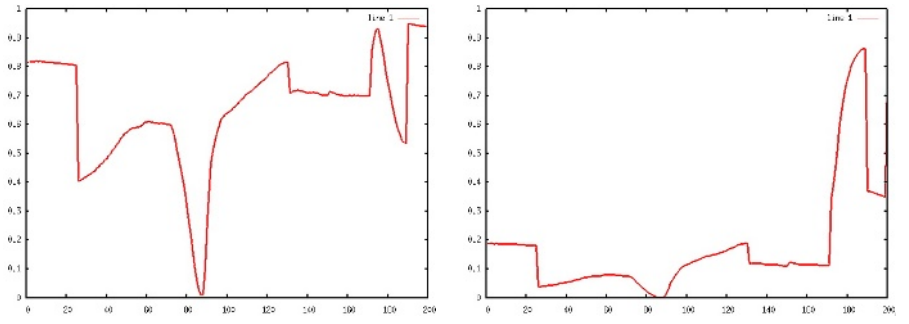
This linear function, $\alpha(t)$, causes a linear decrease in the width of the histogram and thus a logarithmic decrease in the value of the entropy time series. A simple detector can be built to look for logarithmic curves close to monochrome frame sequences, as we will show in the next section.

3 Fade Detection

Entropy time series pattern in the presence of fades is depicted in figure 1. The figure shows a nearly logarithmic curve in the presence of fades (in the $[88, 110]$ range). Motion, lighting changes and irregularities in the luminance histogram produce variations from the expected logarithmic values in the series.

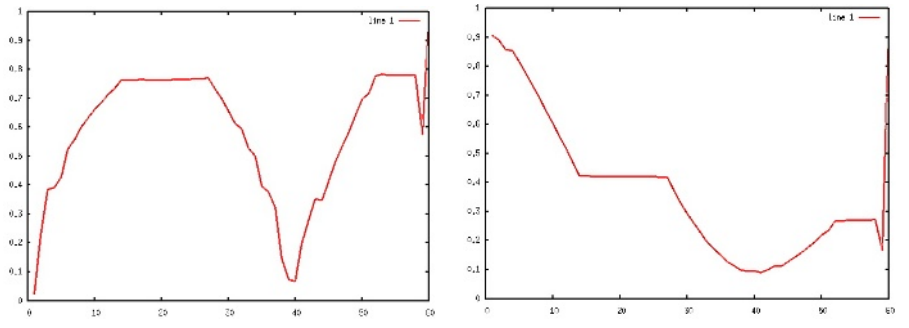
Figures 1 and 2 show the differences between entropy and mean luminance series, where it is shown that entropy is a better alternative to detect fades. Firstly, entropy is able to filter low information values, so monochrome frames can be detected more accurately (avoiding artifacts such as on screen text, etc). Near frame 20, figure 1 shows a sudden fall in both series, caused by a hard-cut; while the entropy series stays clearly above the monochrome threshold, luminance falls to values that could be considered as monochrome frames. Furthermore, if we consider figure 2(b), near frame 40 a fade exists where there is some on-screen text. The monochrome sequence of this fade have luminance values above the non-monochrome frame luminance of the previously mentioned cut in figure 1(b).

In addition to those advantages, entropy series are not affected by the color the fade goes to (black, white, ...). Entropy is a measurement of the width of



(a) Entropy Series (b) Mean luminance series

Fig. 1. Frame Sequence A - Entropy and mean luminance series. The sequence has 3 cuts (frames 23, 130 and 190); a fade-out ([63-88]) and a fade-in ([88-110]).



(a) Entropy series (b) Mean luminance series

Fig. 2. Frame Sequence B- Entropy and mean luminance series. The sequence has two fade-ins ([0, 15], [40, 53]) and a fade out([30-40]).

the histogram, and not its value, so if the scene is mainly composed of one single luminance value, the entropy will be low. Compare the entropy and luminance series in figure 2. The FadeIn - FadeOut - FadeIn sequence is much clearer in 2(a) because the first fade-in comes from a white (not black) series of monochrome frames.

Note also that, as it usually happens, the fade-in between frames 90 and 130 does not increase linearly. This is caused by lighting changes and motion, and can produce missing detections when using quite selective filters. Detecting such irregular fades causes a decrease in precision.

Knowing the pattern that a fade exhibits in the entropy time series, authors have developed a detector based on the correlation of the real series and the theoretical logarithmic series.

3.1 Fade Detection Algorithm Introduction

Using a similar procedure to the one introduced in [9], the first stage of the fade detection will be the location of the monochrome frame sequences of the

video, characterized by very low entropy values even when the color is not black and even in the presence of on-screen text or images. A frame sequence $M^i = \{f_0^i, f_1^i, \ldots, f_{k-1}^i\}, k \geq 1$ is monochromatic if all its frames are monochromatic, $H(f_j^i) < \theta_{mono}$, being $H(f_j^i)$ the entropy of frame f_j^i and θ_{mono} the monochrome threshold. Fades can only happen just before or after a monochrome frame sequence. Locating monochrome sequences allows to discard large portions of the entropy series. For every sequence M^i, we will analyze the preceding and succeeding series to detect fade-outs and fade-ins respectively.

To avoid false detections, we use a correlation based comparison approach using a theoretical model of a fade in the entropy series. Some effects other than fades can decrease entropy gradually to the monochrome region; consider a light spot aiming progressively the camera. When the light comes directly to the camera, low entropy frames are produced. However, the pattern this effect produces does not resemble the one produced by a fade in the entropy series. Therefore, potential fades are compared to a theoretical model by means of correlation. Correlation between two random variables, X and Y, is defined as:

$$\rho(X,Y) = \frac{Cov(X,Y)}{\sigma_X \sigma_Y}, \quad \text{where } -1 \leq \rho(X,Y) \leq 1 \tag{7}$$

$$Cov(X,Y) = E[(X - E[X])(Y - E[Y])] \tag{8}$$

where $Cov(X,Y)$ is the covariance between X and Y, and $Var[X] = \sigma_X^2$ is the variance of X. When variables are independent, correlation takes values close to 0. In the opposite case, the absolute value is close to 1. When correlation between the real series and the model is high, $\rho(X,Y) > \theta_{cov}$, a fade is detected.

This approach relies in the fact that fades will always produce such a pattern in the time series. Unfortunately, fades do not show such a regular behavior. We can find different production functions, $\alpha(t)$. Consider also the noise in the series caused by motion or illumination changes. We cannot expect to detect all the fades using a high threshold, θ_{cov}, and a single model. However, decreasing θ_{cov} could lead to false hits and, consequently, to lower precision. These facts lead us to build adapted models for each possible fade.

3.2 Model-Based Approach

Let us consider a potential fade-out in the set of frames $F^i = \{f_0^i, \ldots, f_{n-1}^i\}$.

Assuming a constant distribution of pixels in the histogram, we can simplify the computation of the entropy of a frame, $I \equiv f_j^i$, using

$$\begin{aligned} H(I) &= -\sum_{k=0}^{N-1} p(k) log(p(k)) = -\sum_{k=0}^{N-1} \frac{h_I(k)}{\sum_{c=0}^{N-1} h_I(c)} log(\frac{h_I(k)}{\sum_{c=0}^{N-1} h_I(c)}) = \\ &= -\sum_{k=0}^{N-1} \frac{1}{N} log(\frac{1}{N}) \quad = log(N) \end{aligned} \tag{9}$$

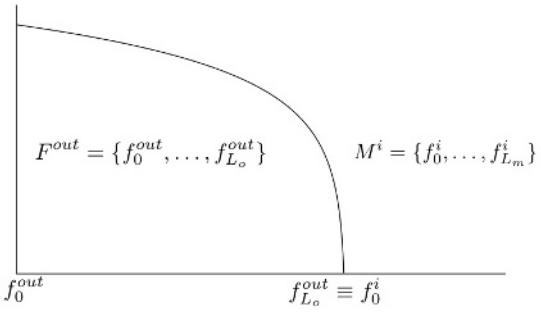


Fig. 3. Entropy model for a fade out

During a fade-out, and assuming a linear production function $\alpha(t)$, the histogram narrows linearly. The number of histogram bins that contain pixels, K, decreases. The entropy of these frames can be computed as:

$$H(I) = -\sum_{k=0}^{K-1} p(k)log(p(k)) - \sum_{k=K}^{N-1} 0\, log(0) =$$
$$= -\sum_{k=0}^{K-1} \frac{1}{K} log(\frac{1}{K}) \;=\; log(K) \tag{10}$$

This result means that a fade will be shown in the time-series as a logarithmic curve, which we would model taking the begin and end values as well as the fade length. Using these three parameters, we build a logarithmic model for each fade, and then compute correlation between the series.

This simple model performs really good in empirical tests, mainly because entropy is a good low-level feature to detect fades. Furthermore, using this model makes the algorithm automatically suitable to detect fades produced by other $\alpha(t)$ functions. If we consider the function x^2, histogram width will decrease quadratically. Thanks to the properties of the logarithm, $log(x^2) = 2log(x)$, the exponent is transformed into a factor that is ignored by the correlation value. Thus, using just one model we are able to detect fades produced using multiple production functions($x^\beta, \beta \in \Re$), as those in figure 2(b).

3.3 Fade Detection Algorithm

Using all these elements, the detection algorithm locates the next sequence of monochrome frames $M^i = \{f_0^i, f_1^i, \ldots, f_{k-1}^i\}$ and then study the surrounding values. The procedure to detect fade-outs and fade-ins is analogous; only fade-out detection is, thus, discussed. Figure 3 shows a fade-out sample labeled with the notation used in the algorithm explanation below.

If a fade-out exists, the series values before the beginning of the monochrome region, f_0^i, must decrease, as shown in the range $[f_0^{out}, f_{L_0}^{out}]$ of figure 3. Using the

first derivative of the series, the algorithm quickly finds this maximum fade-out range, $[f_0^{out}, f_{L_0}^{out}]$, where $f_{L_0}^{out} \equiv f_0^i$. For the range of series $[f_j^{out}, f_{L_o}^{out}], j \in [0, L_o]$ the algorithm:

- Computes correlation for $A = [f_j^{out}, f_{L_o}^{out}]$ and $B = [f_{j+1}^{out}, f_{L_o}^{out}]$ with the model.
- if $length(A) < l_{min}$ there is no fade-out
- else if $\rho(A, Model) < \theta_{cov}$ or $\rho(A, Model) < \rho(B, Model)$ then we consider $j = j + 1$.
- else a fade-out exists in the range $[f_j^{out}, f_{L_o}^{out}]$.

The algorithm will select larger fades before shorter and will completely discard fades shorter than l_{min} to avoid false detections. The final fade range is selected as the one that maximizes correlation.

4 Results

This section presents an evaluation of the proposed detector focusing on: features of entropy as a fade detector and correlation analysis of our mathematical model. These two points are introduced in this paper as an improvement over previous work related to the detection of gradual transitions, fades in particular. The video test set used has been selected to provide information in the worst case, including very different fade patterns and motion/lighting effects that are known to cause errors (false and missed detections). A first set of test videos are action movie sequences (from commercial DVDs such as Final Fantasy: the spirits within) featuring explosions, zooms, fast camera panning and high motion activity. The second set of videos includes segments (coming mainly from movie trailers available at quicktime.com) with features that affect entropy-based detection: low entropy of the shots surrounding the transitions (e.g text over a plain background), sudden lighting changes, too dark and too bright scenes, etc.

4.1 Performance of Entropy as a Fade Detector

The results presented in table 1 illustrate high average precision (94%) and recall (96%) values obtained using our detector. That are the best values and were obtained using a configuration with $\theta_{mono} = 0.15$ and $\theta_{cov} = 0.875$. The θ_{mono} value does not affect the quality of the detection if it stays in the $[0.15, 0.4]$ range (that can be expected just by looking at figures 1(a) and 2(a)). Setting θcov correctly is more relevant. While results are similar in the $[0.86, 0.91]$ range, getting beyond these limits makes either precission or recall fall below 90%.

Table 1. Results - $\theta_{cov} = 0.875$. Average Recall=96%. Average Precision=94%.

Video Set	Total Duration	N. of fades	Detected	False	Recall	Precision
Set 1	2021 sg	427	415	23	97%	95%
Set 2	3210 sg	573	524	15	91%	97%

The algorithm is therefore very accurate and features good tolerance to motion and lighting effects and independence from the fade length and monochrome color. The noise present in the first set of videos do not affect the recall value, but mainly precision (false detecting events that exhibit the same pattern as fades: e.g. a zoom into a dark object or objects appearing over a low-entropy background). Recall values are affected by low entropy sequences in which the model is not very clear (e.g. shots with text over plain background).

4.2 Analysis of Mathematical Model

One of the most common problems of shot detection is to get the highest recall value without increasing the number of false hits. Threshold based algorithms, as the one we have introduced, can be tuned to get the optimum combination of recall and precision changing the threshold. However, each video has its own optimum threshold. Furthermore, this technique always decreases precision to increase recall. The mathematical model presented has shown to be valid; recall and precision are above 90% for a reasonably wide range of θ_{cov} values. Figure 4 shows the theoretical and real entropy series for a fade-out. It can be seen both series differ slightly; that is caused by the presence of motion and light changes in the video sequence. The correlation between series is above 99%, which shows the good behaviour of the model to detect fades even with noise in the series. Table 2 shows our model good behavior. Average, minimum and maximum correlation values are shown for the three cases that the algorithm can come across: a real fade, a false fade incorrectly detected and a false fade correctly discarded. For real fades, the minimum value obtained with our method is above 0.91 while

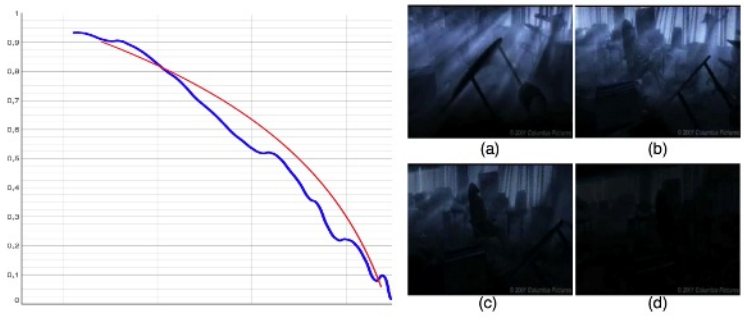


Fig. 4. Theoretical and real entropy series. $\rho = 0.99038$.

Table 2. Correlation Results - Between real&false fades, fade-like effects and our model

Kind	Avg Cor.	Max Cor.	Min Cor.
Accepted Fades	0,97	0,99	0,91
Accepted False Fades	0,979	0,99	0,901
Discarded Fades	0.77023	0,85	0,66

the maximum value that fade-like effects obtain is 0.85. We used the arithmetic mean value, 0.875, as a the threshold to obtain table 1 results. Any other value in this range will provide also good results. If precision is preferred over recall, a higher threshold, $\theta_{cov} \approx 0.91$, will be used. In other case, lower threshold values, $\theta_{cov} \approx 0.85$, should be considered.

5 Conclusions

In this paper we have successfully used entropy for fade detection and obtained satisfactory results. Entropy series present features that make it appropriate to detect fades. Firstly, fade produces an easy to recognize entropy pattern. Fades that use colors other than black produce the same pattern, so entropy provides independence from the monochrome color. In addition, motion and illumination do not produce significant changes in the series that can result in a missed or false detection. To conclude, in the presence of artifacts and noise (such as on-screen text) entropy values stay below θ_{mono}, so fades can be recognized.

Most existing approaches for shot-boundary detection are based on given thresholds, which determine the detection performance. The problem of specifying such a precise threshold and the sensitivity of this method make its validity questionable. In our detector, thanks to the selective and regular behavior of entropy series and the use of a mathematical model, we get a very discriminative estimator, avoiding being too sensitive to the threshold value, with independence of the video being analyzed.

References

1. Boreczky, J., Rowe, L.: Comparison of video shot boundary detection techniques, SPIE (1996) 122–128
2. Guimaraes, S., Leite, N., Araújo, A.: A method for cut detection based on visual rhythm. In: IEEE SIBGRAPI, IEEE (2001) 297–304
3. Meng, J., Juan, Y., Chang, S.F.: Scene change detection in a mpeg comrpessed video sequence. In: IS&T SPIE Symposium for Video Technology. (1995)
4. Won, J.U., Chung, Y.S., Kim, I.S., Choi, J.G., Park, K.H.: Correlation based video-dissolve detection. In: IEEE International Conference on Information Technology: Research and Education. (2003) 104–107
5. Truong, B.T., Dorai, C., Venkatesh, S.: Improved fade and dissolve detection for reliable video segmentation. In: IEEE Int. Conf. Image Processing. (2000)
6. Campisi, P., Neri, A., Sorgi, L.: Automatic dissolve and fade detection for video sequences. In: 14th International Conference on Digital Signal Processing. (2002)
7. Han, S.H., Kweon, I.S.: Detecting cuts and dissolves through linear regression analysis. IEEE Electronic Letters **39** (2003) 1579–1581
8. Z.Cernekova;, Nikou, C., Pitas, I.: Entropy metrics used for video summarization. In: Proceedings 18th conference on Computer graphics, ACM Press (2002) 73–82
9. Lienhart, R., abd W. Effelsberg, C.K.: On the Detection and Recognition of Television Commercials. In: Proceedings of the International Conference on Multimedia Computing and Systems. (1997)

Order of Magnitude Qualitative Reasoning with Bidirectional Negligibility*

A. Burrieza[1], E. Muñoz[2], and M. Ojeda-Aciego[2]

[1] Dept. Filosofía. Universidad de Málaga. Spain
burrieza@uma.es
[2] Dept. Matemática Aplicada. Universidad de Málaga. Spain
{emilio, aciego}@ctima.uma.es

Abstract. In this paper, we enrich the logic of order of magnitude qualitative reasoning by means of a new notion of negligibility which has very useful properties with respect to operations of real numbers. A complete axiom system is presented for the proposed logic, and the new negligibility relation is compared with previous ones and its advantages are presented on the basis of an example.

1 Introduction

Qualitative reasoning is an emergent area of AI. It is an adequate tool for dealing with situations in which information is not sufficiently precise (e.g., numerical values are not available).

A form of qualitative reasoning is to manage numerical data in terms of orders of magnitude (OM) (see, for example, [7,8,9,11,13,15]). There are two approaches for OM that can be combined: Absolute Order of Magnitude (AOM), which is represented by a partition of the real line $\mathbb{R}$ and each element of $\mathbb{R}$ belongs to a qualitative class and Relative Order of Magnitude (ROM), introducing a family of binary order of magnitude relations which establishes different comparison relations in $\mathbb{R}$ (e.g., *comparability*, *negligibility* and *closeness* [13]).

Several logics have been defined to deal with qualitative reasoning, such as the Region Connection Calculus [2, 14] for managing qualitative spatial reasoning; or the multimodal logics used in [3, 17] to deal with qualitative spatio-temporal representations, and the use of branching temporal logics to describe the possible solutions of ordinary differential equations when we have a lack of complete information about a system [12]: however, an analogous development of order-of-magnitude reasoning from a logical approach standpoint has received little attention: to the best of our knowledge, the only logics dealing with order-of-magnitude reasoning have been developed in [5, 6]. The present paper continues the line of research of a logical approach to order-of-magnitude reasoning.

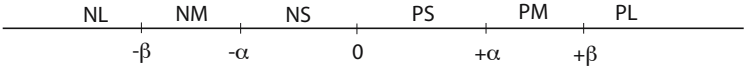
Regarding the underlying representation model, it seems natural to consider an absolute order of magnitude model with a small number of landmarks, so that the size of the proof system obtained is reasonable. Following usual practice, we

* Partially supported by Spanish project TIC2003-9001-C02-01.

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 370–378, 2006.

will divide the real line into seven equivalence classes using five landmarks chosen depending on the context [10, 16].

The system considered corresponds to the schematic representation shown in the picture below:



were α, β are two positive real numbers such that $\alpha <_{\mathbb{R}} \beta$, and $\leq_{\mathbb{R}}$ the usual order in $\mathbb{R}$. Moreover, in this work we consider that:

$$\mathrm{NL} = (-\infty, -\beta), \qquad \mathrm{NM} = [-\beta, -\alpha), \qquad \mathrm{NS} = [-\alpha, 0), \qquad [0] = \{0\},$$
$$\mathrm{PS} = (0, \alpha], \qquad \mathrm{PM} = (\alpha, \beta], \qquad \mathrm{PL} = (\beta, +\infty)$$

The labels correspond to "negative large", "negative medium", "negative small", "zero", "positive small", "positive medium" and "positive large", respectively.

The main result of this paper is the introduction of a new bidirectional negligibility relation in the logic for qualitative reasoning with orders of magnitude introduced in [6]. This new negligibility relation is "more quantitative" than the original one and, thus, is closer to the one presented in [11] but less complex. As a consequence, our approach (between purely qualitative and purely quantitative) allows us to have the expressive power of the logic and a lot of useful properties of the negligibility relation.

The notion of negligibility that will be used in the rest of the paper is given in the following definition:

Definition 1. *Given $\alpha, \beta, x, y \in \mathbb{R}$, such that $0 <_{\mathbb{R}} \alpha <_{\mathbb{R}} \beta$, we say that x is* **negligible** *with respect to (wrt from now on) y, in symbols $x\, N_{\mathbb{R}}\, y$, if and only if, we have one of the following possibilities:*

$$(i) \quad x = 0 \qquad (ii) \quad x \in \mathrm{NS} \cup \mathrm{PS} \text{ and } y \in \mathrm{NL} \cup \mathrm{PL}$$

Note that item (i) above corresponds to the intuitive idea that 0 is negligible wrt any real number and item (ii) corresponds to the intuitive idea that a number *sufficiently small* is negligible wrt any number *sufficiently large*, independently of the sign of these numbers. Thus, we have that $N_{\mathbb{R}}$ is not a restriction of $<_{\mathbb{R}}$. In this way, we say that $N_{\mathbb{R}}$ is *bidirectional*.

The paper is organized as follows: In Section 2, the syntax and semantics of the proposed logic is introduced; in Section 3, the axiom system for our language is presented; in Section 4, a comparison with alternative definitions of negligibility is done, and an example is presented on which some features of the proposed relation are shown. Finally, some conclusions and future work are presented.

2 Syntax and Semantics of the Language $\mathcal{L}(MQ)^{N_{\mathbb{R}}}$

In a similar way to [6], we define the connectives $\overrightarrow{\Box}$, $\overleftarrow{\Box}$, $\Box_N$ and $\overline{\Box}_N$ to deal with the relations $<$ and N, respectively. The intuitive meanings of each modal connective is as follows:

- $\overrightarrow{\Box}\, A$ means A *is true for all element greater than the current one.*
- $\overleftarrow{\Box}\, A$ means A *is true for all number less than the current one.*
- $\Box_N A$ is read A *is true for all number from which the current one is negligible.*
- $\overline{\Box}_N A$ is read A *is true for all number which is negligible from the current one.*

Now, we define the language $\mathcal{L}(MQ)^{N_{\mathbb{R}}}$ of our logic. The alphabet of $\mathcal{L}(MQ)^{N_{\mathbb{R}}}$ is defined by using:

- A stock of atoms or propositional variables, $\mathcal{V}$.
- The classical connectives $\neg, \wedge, \vee, \rightarrow$ and the constants $\top$ and $\bot$.
- The unary modal connectives $\overrightarrow{\Box}$, $\overleftarrow{\Box}$, $\Box_N$ and $\overline{\Box}_N$.
- The finite set of specific constants $\mathcal{C}$ defined by $\mathcal{C} = \{\beta^-, \alpha^-, \overline{0}, \alpha^+, \beta^+\}$.
- The auxiliary symbols $(,)$.

Formulae of $\mathcal{L}(MQ)^{N_{\mathbb{R}}}$ are generated from $\mathcal{V} \cup \mathcal{C}$ by the construction rules of classical propositional logic adding the following rule:

If A is a formula, then so are $\overrightarrow{\Box}\, A$, $\overleftarrow{\Box}\, A$, $\Box_N A$ and $\overline{\Box}_N A$. The *mirror image* of A is the result of replacing in A the occurrence of $\overrightarrow{\Box}, \overleftarrow{\Box}, \Box_N, \overline{\Box}_N, \beta^-, \alpha^-, \overline{0}, \alpha^+, \beta^+$ by $\overleftarrow{\Box}, \overrightarrow{\Box}, \overline{\Box}_N, \Box_N, \beta^+, \alpha^+, \overline{0}, \alpha^-, \beta^-$ respectively. As usual in modal logic, we use $\overrightarrow{\Diamond}$, $\overleftarrow{\Diamond}$, $\Diamond_N$, and $\overline{\Diamond}_N$ as abbreviations respectively of $\neg\, \overrightarrow{\Box}\, \neg$, $\neg\, \overleftarrow{\Box}\, \neg$, $\neg \Box_N \neg$ and $\neg \overline{\Box}_N \neg$.

Definition 2. *A* **qualitative frame** *for $\mathcal{L}(MQ)^{N_{\mathbb{R}}}$ is a tuple $\Sigma = (\mathbb{S}, C, <, N)$, where:*

1. *$\mathbb{S}$ is a nonempty set of real numbers.*
2. *$C \subseteq \mathbb{S}$ where $C = \{-\beta, -\alpha, 0, \alpha, \beta\}$.*
3. *$<$ and N are, respectively, the restriction to $\mathbb{S}$ of the relations $<_{\mathbb{R}}$ and $N_{\mathbb{R}}$ above.*

If condition 2 it not fulfilled, we say that $(\mathbb{S}, <, N)$ is a **pre-frame**.

Notation: We will sometimes assume the following notations, in order to simplify the presentation of results:

- $c_1 = -\beta, c_2 = -\alpha, c_3 = 0, c_4 = +\alpha, c_5 = +\beta$
- $\overline{c}_1 = \beta^-, \overline{c}_2 = \alpha^-, \overline{c}_3 = \overline{0}, \overline{c}_4 = \alpha^+, \overline{c}_5 = \beta^+$
- If $X \subseteq \mathbb{R}$, R is a relation in X and $a \in X$:

$$R(a) = \{a' \in \mathbb{R} \mid aRa'\}$$
$$R^{-1}(a) = \{a' \in \mathbb{R} \mid a'Ra\}$$

Definition 3. *Let $\Sigma = (\mathbb{S}, C, <, N)$ be a qualitative frame for $\mathcal{L}(MQ)^{N_{\mathbb{R}}}$, a* **qualitative model** *for Σ (or, simply Σ-model) is an ordered pair $\mathcal{M} = (\Sigma, h)$ where $h : \mathcal{V} \rightarrow 2^{\mathbb{S}}$ is a function called* **interpretation**. *Any interpretation can be uniquely extended to the set of all formulae in $\mathcal{L}(MQ)^{N_{\mathbb{R}}}$ (also denoted by h)*

by means of the usual conditions for the classical boolean connectives and for $\top$, $\bot$, *and the following conditions* [1]*:*

- $h(\overrightarrow{\Box}\, A) = \{x \in \mathbb{S} \mid (x, +\infty) \subseteq h(A)\}$.
- $h(\overleftarrow{\Box}\, A) = \{x \in \mathbb{S} \mid (-\infty, x) \subseteq h(A)\}$.
- $h(\Box_N A) = \{x \in \mathbb{S} \mid N(x) \subseteq h(A)\}$.
- $h(\overline{\Box}_N A) = \{x \in \mathbb{S} \mid N^{-1}(x) \subseteq h(A)\}$.
- $h(\overline{c}_i) = \{c_i\}$ *for all* $i \in \{1, 2, 3, 4, 5\}$.

The concepts of truth and validity are defined in a standard way.

3 Axiom System for $\mathcal{L}(MQ)^{N_{\mathbb{R}}}$

We will denote $MQ^{N_{\mathbb{R}}}$ the axiom system with all the tautologies of classical propositional logic together with the following axiom schemata:

Axiom Schemata for Modal Connectives:

K1 $\overrightarrow{\Box}\,(A \to B) \to (\overrightarrow{\Box}\, A \to \overrightarrow{\Box}\, B)$
K2 $A \to \overrightarrow{\Box}\overleftarrow{\Diamond}\, A$
K3 $\overrightarrow{\Box}\, A \to \overrightarrow{\Box}\overrightarrow{\Box}\, A$
K4 $\big(\overrightarrow{\Box}\,(A \vee B) \wedge \overrightarrow{\Box}\,(\overrightarrow{\Box}\, A \vee B) \wedge \overrightarrow{\Box}\,(A \vee \overrightarrow{\Box}\, B)\big) \to (\overrightarrow{\Box}\, A \vee \overrightarrow{\Box}\, B)$

Axiom Schemata for Constants:

C1 $\overleftarrow{\Diamond}\, \overline{c}_i \vee \overline{c}_i \vee \overrightarrow{\Diamond}\, \overline{c}_i$, where $i \in \{1, 2, 3, 4, 5\}$
C2 $\overline{c}_i \to (\overleftarrow{\Box}\, \neg\overline{c}_i \wedge \overrightarrow{\Box}\, \neg\overline{c}_i)$, being $i \in \{1, 2, 3, 4, 5\}$
C3 $\overline{c}_i \to \overrightarrow{\Diamond}\, \overline{c}_{i+1}$, where $i \in \{1, 2, 3, 4\}$.

Axiom Schemata for Negligibility Connectives:

N1 $\Box_N(A \to B) \to (\Box_N A \to \Box_N B)$
N2 $A \to \Box_N \overline{\Diamond}_N A$
N3 $(\overleftarrow{\Box}\, A \wedge A \wedge \overrightarrow{\Box}\, A) \to \Box_N A$
N4 $(\overrightarrow{\Diamond}\, \alpha^- \vee \overleftarrow{\Diamond}\, \alpha^+) \to \Box_N \bot$
N5 $\overline{0} \to (\Box_N A \to (\overleftarrow{\Box}\, A \wedge A \wedge \overrightarrow{\Box}\, A))$
N6 $(\neg\overline{0} \wedge (\alpha^- \vee (\overleftarrow{\Diamond}\, \alpha^- \wedge \overrightarrow{\Diamond}\, \alpha^+) \vee \alpha^+)) \to \Box_N(\overrightarrow{\Diamond}\, \beta^- \vee \overleftarrow{\Diamond}\, \beta^+)$
N7 $\big(\neg\overline{0} \wedge (\alpha^- \vee (\overleftarrow{\Diamond}\, \alpha^- \wedge \overrightarrow{\Diamond}\, \alpha^+) \vee \alpha^+)\big) \to$
$\big(\Box_N A \to (\overleftarrow{\Box}\,(\overrightarrow{\Diamond}\, \beta^- \to A) \wedge \overrightarrow{\Box}\,(\overleftarrow{\Diamond}\, \beta^+ \to A))\big)$

We also consider as axioms the corresponding mirror images of axioms K1-K4, and axioms N1-N3.

[1] These algebraic conditions for modal connectives are based on the intuitive meanings presented above.

Rules of Inference:

(MP) Modus Ponens for $\rightarrow$
(R$\overrightarrow{\Box}$) If $\vdash A$ then $\vdash \overrightarrow{\Box} A$
(R$\overleftarrow{\Box}$) If $\vdash A$ then $\vdash \overleftarrow{\Box} A$

Informal reading of specific axioms is the following:

- Axioms C1–C3 formalize, respectively, the existence, uniqueness and ordering of the constants.
- Axioms N1-N2 are standard in modal logic.
- Axiom N3 means that the bidirectional relation N is a restriction of $< \cup >$.
- Axiom N4 expresses that neither large nor medium elements are negligible with respect to any number.
- Axiom N5 means that 0 is negligible wrt every number.
- Axiom N6 means that all elements wrt which small elements are negligible are large. Axiom N7 means that any small number is negligible wrt any large number. Summarising, axioms N6 and N7 mean that $x \neq 0$ is negligible wrt y if and only if x is small and y is large (as expressed in Definition 1).

Theorem 1 (Soundness and Completeness)

- *Every theorem of* $MQ^{N_{\mathbb{R}}}$ *is a valid formula of* $\mathcal{L}(MQ)^{N_{\mathbb{R}}}$.
- *Every valid formula of* $\mathcal{L}(MQ)^{N_{\mathbb{R}}}$ *is a theorem of* $MQ^{N_{\mathbb{R}}}$.

The soundness of the axiom system is straightforward.
Regarding completeness, a *step-by-step* proof (see, for example, [1] and [4]) can be given in the following terms:

Given any consistent formula A , we have to prove that A is satisfiable. With this purpose, the step-by-step method defines a qualitative frame $\Sigma = (\mathbb{S}, C, <, N)$ and a function f_Σ which assigns maximal consistent sets to any element of $\mathbb{S}$, such that $A \in f_\Sigma(x)$ for some $x \in \mathbb{S}$. The process to build such a frame is recursive, and follows the ideas of [6]: firstly, a pre-frame is generated which is later completed to an initial finite frame; later, successive extensions of this initial frame are defined until Σ is obtained.

Although the method of proof is the same, the technical problems which arise from the use of this more complex language need special attention. Due to lack of space, the formal details are omitted.

4 Comparison with Other Definitions of Negligibility

In the definition of our negligibility relation $N_{\mathbb{R}}$, we pursue a balance between the applicability to different types of problems and the complexity of the logic that we want to construct. For this reason, this section is devoted to compare the properties of $N_{\mathbb{R}}$ with other definitions of negligibility, in particular, the ones presented in [6, 11].

The negligibility relation presented in [6], denoted by $\prec$ is defined as a restriction of $<_{\mathbb{R}}$ satisfying the following properties: (i) If $x \prec y <_{\mathbb{R}} z$, then $x \prec z$; (ii) If $x <_{\mathbb{R}} y \prec z$, then $x \prec z$; (iii) If $x \prec y$, then either $x \notin INF$ or $y \notin INF$, being $\alpha \in \mathbb{R}$ and considering the real line divided in three equivalence classes using two landmarks α and $-\alpha$ as follows:

$$\begin{aligned} OBS^- &= \{x \in \mathbb{R} \mid x \leq_{\mathbb{R}} -\alpha\}; \\ INF &= \{x \in \mathbb{R} \mid -\alpha \leq_{\mathbb{R}} x \leq_{\mathbb{R}} \alpha\}; \\ OBS^+ &= \{x \in \mathbb{R} \mid \alpha \leq_{\mathbb{R}} x\} \end{aligned}$$

The main differences between $N_{\mathbb{R}}$ and $\prec$ are that $\prec$ is a restriction of $<_{\mathbb{R}}$ and $N_{\mathbb{R}}$ is not. Indeed $N_{\mathbb{R}}$ is bidirectional and this fact allows us to compare positive and negative numbers. Another difference is that $N_{\mathbb{R}}$ uses the standard division of the real line in seven classes while $\prec$ only uses three. Thus, $N_{\mathbb{R}}$ verifies a version of property (iii) in the definition of $\prec$, taking into account that we have a different number of intervals and substituting INF by $NS \cup PS$. $N_{\mathbb{R}}$ satisfies neither (i) nor (ii) because, as commented above, $N_{\mathbb{R}}$ is not a restriction of $<_{\mathbb{R}}$. However, if we work just with positive numbers, then the two properties are fulfilled. $N_{\mathbb{R}}$ verifies one of the properties of quasi-density studied in [6], i.e. if $x\, N_{\mathbb{R}}\, y$, then there exists $z \in \mathbb{R}$ such that $x\, N_{\mathbb{R}}\, z <_{\mathbb{R}} y$, for all $x \in \mathbb{R}$ but does not satisfy the other one, i.e. if $x\, N_{\mathbb{R}}\, y$, then there exists $z \in \mathbb{R}$ such that $x <_{\mathbb{R}} z\, N_{\mathbb{R}}\, y$, because it fails when $x = 0$ or $x = \alpha$. However, none of these properties is obtained from the definition of $\prec$.

On the other hand, the negligibility relation (also bidirectional) presented in [11], denoted by *Ne*, is defined as follows: Given $\alpha, \beta \in \mathbb{R}$ such that $0 <_{\mathbb{R}} \alpha <_{\mathbb{R}} \beta$, then $x\ Ne\ y$ if and only if, $|\frac{x}{y}| <_{\mathbb{R}} \frac{\alpha}{\beta}$. The main difference wrt our relation is that *Ne* is not supported by a logic whereas $N_{\mathbb{R}}$ is; in addition, $N_{\mathbb{R}}$ maintains the majority of the properties of *Ne* as we can see below.

Proposition 1. *Consider $x, y, z, t \in \mathbb{R}$, then we have*

1. $0\, N_{\mathbb{R}}\, x$
2. *If $y \neq 0$ and $x\, N_{\mathbb{R}}\, y$, then $|x| < |y|$.*
3. *If $x \neq 0$ and $x\, N_{\mathbb{R}}\, y$, then it is not the case that $y\, N_{\mathbb{R}}\, x$*
4. *If $x\, N_{\mathbb{R}}\, y$ and $y\, N_{\mathbb{R}}\, z$, then $x\, N_{\mathbb{R}}\, z$*
5. *If $x\, N_{\mathbb{R}}\, y$, then $(\pm x)\, N_{\mathbb{R}}\, (\pm y)$*
6. *If $x\, N_{\mathbb{R}}\, y$, $|z| \leq |x|$ and $|y| \leq |t|$, then $z\, N_{\mathbb{R}}\, t$*
7. *If $x\, N_{\mathbb{R}}\, y$, then $sign(x + y) = sign(y)$*
8. *If $sign(x+y) = +$ and $sign(x) = -$, then $sign(y) = +$ and it is not the case that $y\, N_{\mathbb{R}}\, x$*
9. *If $x\, N_{\mathbb{R}}\, y$ and $sign(y) = sign(z)$, then $x\, N_{\mathbb{R}}\, (y + z)$*
10. *If $y \neq 0$ and $(x - y)\, N_{\mathbb{R}}\, y$, then it is not the case that $y\, N_{\mathbb{R}}\, x$*
11. *If $\beta \geq 2\alpha$, $y \in PL$ and $x\, N_{\mathbb{R}}\, y$, then $x + y \in PM \cup PL$.*
12. *If $\beta \geq 2\alpha$, $y \in NL$ and $x\, N_{\mathbb{R}}\, y$, then $x + y \in NM \cup NL$.*
13. *If $x\, N_{\mathbb{R}}\, y$, $|z| \leq 1 \leq |t|$, then $zx\, N_{\mathbb{R}}\, ty$*

Summarising, we can say that $N_{\mathbb{R}}$ presents different features than $\prec$ which make it closer to Ne and, consequently, $N_{\mathbb{R}}$ is somewhere in between the other two relations, sharing with them some qualitative and some quantitative interesting properties, apart from the fact that both $\prec$ and $N_{\mathbb{R}}$ are founded on a logic.

We finish this section with the next example which is devoted to specify a small use of our logic. We want to emphasize the use of the bidirectionallity and properties of our negligibility relation. Moreover, the use of the axiom system allows us to obtain some interesting properties from the specification of the example.

Example 1. Let us suppose that we want to specify the behaviour of a device to automatically control the temperature, for example, in a museum, where we need to have some specific conditions. If we have to maintain the temperature close to some limit T, for practical purposes any value of the interval $[T-\epsilon, T+\epsilon]$ for small ϵ is admissible. Then the extreme points of this interval can be considered as the milestones $-\alpha$ and α, respectively. Moreover, assume that if the temperature is out of this interval (for example, because the number of people within the museum is changing), it is necessary to put into operation some *heating* or *cooling* system. In addition, we have another interval $[T-\lambda, T+\lambda]$, such that if the temperature does not belong to this interval, we need to use an extra system of *cooling* or *heating*, because the normal system is not enough. Now, the extreme points of this interval are the milestones $-\beta$ and β, respectively. We also assume that when the normal system of *cooling* or *heating* is operating, a system to maintain the humidity is needed, and when the extra system is operating, we also need an extra system of humidification.

The intervals $\mathrm{NL}, \mathrm{NM}, \mathrm{NS} \cup [0] \cup \mathrm{PS}, \mathrm{PM}$ and PL can be interpreted by Very_Cold, Cold, Ok, Hot and Very_Hot, respectively. The proper axioms that specify the general behaviour of the system are stated below:

$$\textsc{Ok} \rightarrow \mathit{off} \qquad \textsc{Very\_Cold} \rightarrow \mathit{X\text{-}heating}$$
$$\textsc{Cold} \rightarrow \mathit{heating} \qquad \textsc{Hot} \rightarrow \mathit{cooling}$$
$$\textsc{Very\_Hot} \rightarrow \mathit{X\text{-}cooling} \qquad (\textsc{Cold} \vee \textsc{Hot}) \rightarrow \mathit{humidifier}$$
$$(\textsc{Very\_Cold} \vee \textsc{Very\_Hot}) \rightarrow \mathit{X\text{-}humidifier}$$

The following formulae introduce relations among actions:

$$\mathit{X\text{-}heating} \rightarrow (\neg \mathit{heating} \wedge \neg \mathit{off} \wedge \neg \mathit{cooling} \wedge \neg \mathit{X\text{-}cooling} \wedge \mathit{X\text{-}humidifier})$$
$$\mathit{heating} \rightarrow (\mathit{humidifier} \wedge \neg \mathit{extra\text{-}cooling} \wedge \neg \mathit{cooling} \wedge \neg \mathit{off})$$
$$\mathit{off} \rightarrow (\neg \mathit{X\text{-}cooling} \wedge \neg \mathit{cooling} \wedge \neg \mathit{humidifier} \wedge \neg \mathit{X\text{-}humidifier})$$
$$\mathit{cooling} \rightarrow (\neg \mathit{X\text{-}cooling} \wedge \mathit{humidifier})$$
$$\mathit{X\text{-}cooling} \rightarrow \mathit{X\text{-}humidifier}$$
$$\mathit{humidifier} \rightarrow (\mathit{cooling} \vee \mathit{heating})$$
$$\mathit{X\text{-}humidifier} \rightarrow \neg \mathit{humidifier}$$

where *off* means that the system is *off*, *cooling* means that we use the normal system of *cooling* and *X-cooling* means that we need to use an extra cooling system. Analogously, we have the meaning of *heating*, *X-heating*, *humidifier* and *X-humidifier*.

Some consequences of the previous specification that are obtained by using the proposed axiom system are the following:

1. The conditionals in the proper axioms turn out to be bi-conditionals, that is, we also have: $\mathit{off} \rightarrow \textsc{Ok}$, $\mathit{cooling} \rightarrow \textsc{Hot}$, etc.
2. $\neg \mathit{off} \rightarrow \Box_N \bot$
3. $(\mathit{cooling} \vee \mathit{X\text{-}cooling}) \rightarrow (\mathit{humidifier} \vee \mathit{X\text{-}humidifier})$
4. $\mathit{cooling} \rightarrow \overrightarrow{\Box} (\neg \mathit{X\text{-}cooling} \rightarrow \mathit{humidifier})$
5. $(\mathit{off} \wedge \neg \overline{0}) \rightarrow \Box_N \mathit{X\text{-}humidifier}$
6. $\mathit{humidifier} \rightarrow \overline{\Box}_N (\neg \overline{0} \rightarrow \bot)$
7. $\mathit{X\text{-}humidifier} \rightarrow \overline{\Box}_N \mathit{off}$
8. $(\mathit{X\text{-}cooling} \vee \mathit{X\text{-}heating}) \rightarrow \overline{\Box}_N (\neg \mathit{humidifier} \wedge \neg \mathit{X\text{-}humidifier})$

If we assume the intuitive hypothesis $\beta \geq 2\alpha$, the properties 11 and 12 of Proposition 1 can be applied in this example as follows, where p represents *the temperature is incremented (positively or negatively) in the actual value*:

$$(\textsc{Ok} \wedge \neg \bar{0} \wedge \Diamond_N p) \rightarrow (\mathit{humidifier} \vee \mathit{X\text{-}humidifier})$$

This formula can be read in this way: if the temperature is Ok but nonzero and is incremented in a value from which the actual value is negligible, then we have to use the *humidifier* or extra *humidifier* system because the *cooling* or *heating* systems have been put into operation.

5 Conclusions and Future Work

A sound and complete system of qualitative order-of-magnitude reasoning has been introduced with a new notion of negligibility. This new negligibility relation is between those presented in [11] and [6], thus sharing convenient properties from both the qualitative and the quantitative sides. As a result, we have been able to construct a logic with a reasonable number of axioms, which is useful to work in situations where we need to use different properties of the negligibility relation.

As future work, our plan is to investigate a logic with modal operators which represent, in some way, the sum and product of real numbers. Thus, it would be very interesting for the applications (see, for example [13]) to study the behaviour of the negligibility relation $N_{\mathbb{R}}$ wrt the sum and product of real numbers as we have commented in Section 4. It is easy to see that if $x \, N_{\mathbb{R}} \, y$ and $z \in \mathbb{R}$, neither $(x + z) \, N_{\mathbb{R}} \, (y + z)$ nor $xz \, N_{\mathbb{R}} \, yz$ are true in general; thus, we are interested in finding new relations which guarantee such kind of properties under certain conditions. Specifically, we will investigate the definition of new relations to obtain the definability of those properties of sum and product introduced in Proposition 1, in order to extend the current logic with the corresponding new modal connectives.

Last but not least, we are planning to develop theorem provers for our system based either on tableaux or on resolution.

References

1. Blackburn, P., de Rijke. M. and Venema, Y. *Modal Logic*, Cambridge University Press, Cambridge, 2001.
2. Bennett, B. Modal logics for qualitative spatial reasoning. *Bull. of the IGPL*, 3:1–22, 1995.
3. Bennett, B., Cohn, A.G., Wolter, F. and Zakharyaschev, M. Multi-Dimensional Modal Logic as a Framework for Spatio-Temporal Reasoning. *Applied Intelligence*, 17(3): 239–251, 2002
4. Burgess, J.P. Basic tense logic. *Handbook of Philosophical Logic, vol 2: Extensions of Classical Logic*, edited by D. Gabbay and F. Guenthner: 89-133. Reidel, Dorchecht, 1984.
5. Burrieza, A. and Ojeda-Aciego, M. A multimodal logic approach to order of magnitude qualitative reasoning. In *Spanish Conference on Artificial Intelligence*, pages 66-75. Lect. Notes in Artificial Intelligence 3040, 2003.
6. Burrieza, A. and Ojeda-Aciego, M. A multimodal logic approach to order of magnitude qualitative reasoning with comparability and negligibility relations. *Fundamenta Informaticae* 68:21–46, 2005.
7. Dague, P. Numeric reasoning with relative orders of magnitude. In *Proc. 11th National Conference on Artificial Intelligence*, pages 541–547. The AAAI Press/The MIT Press, 1993.
8. Dague, P. Symbolic reasoning with relative orders of magnitude. In *Proc. 13th Intl. Joint Conference on Artificial Intelligence*, pages 1509–1515. Morgan Kaufmann, 1993.
9. Mavrovouniotis, M.L. and Stephanopoulos, G. Reasoning with orders of magnitude and approximate relations. In *Proc. 6th National Conference on Artificial Intelligence*. The AAAI Press/The MIT Press, 1987.
10. Missier, A., Piera, N. and Travé, L. Order of Magnitude Algebras: a Survey. *Revue d'Intelligence Artificielle* 3(4):95–109, 1989
11. Sanchez, M. Prats, F. and Piera, N. Una formalización de relaciones de comparabilidad en modelos cualitativos *Boletín de la AEPIA (Bulletin of the Spanish Association for AI)*. 6: 15-22, 1996.
12. Shults, B. and Kuipers, B.J. Proving properties of continuous systems: qualitative simulation and temporal logic. *Artificial Intelligence*, 92:91–129, 1997.
13. Raiman, O. Order of magnitude reasoning *Artificial Intelligence*. 51: 11-38, 1991.
14. Randell, D., Cui, Z. and Cohn, A. A spatial logic based on regions and connections. *Proc. of the 3rd Intl Conf on Principles of Knowledge Representation and Reasoning* (KR'92), pg 165–176, 1992.
15. Travé-Massuyès, L., Prats, F., Sánchez, M. and Agell, N. Consistent relative and absolute order-of-magnitude models. In *Proc. Qualitative Reasoning 2002 Conference*, 2002.
16. Travé-Massuyès, L., Ironi, L. and Dague, P. Mathematical foundations of qualitative reasoning. *AI magazine* 24(3):91–106, 2003.
17. Wolter, F. and Zakharyaschev, M. Qualitative spatio-temporal representation and reasoning: a computational perspective. In G. Lakemeyer and B. Nebel, editors, *Exploring Artificial Intelligence in the New Millenium*. Morgan Kaufmann, 2002.

Propagating Updates in Real-Time Search: HLRTA*(k)*

Carlos Hernández and Pedro Meseguer

Institut d'Investigació en Intel.ligència Artificial
Consejo Superior de Investigaciones Científicas
Campus UAB, 08193 Bellaterra, Spain
{chernan, pedro}@iiia.csic.es

Abstract. We enhance real-time search algorithms with bounded propagation of heuristic changes. When the heuristic of the current state is updated, this change is propagated consistently up to k states. Applying this idea to HLRTA*, we have developed the new HLRTA*(k) algorithm, which shows a clear performance improvement over HLRTA*. Experimentally, HLRTA*(k) converges in less trials than LRTA*(k), while the contrary was true for these algorithms without propagation. We provide empirical results showing the benefits of our approach.

1 Introduction

Real-time search interleaves planning and action execution in an on-line manner. This allows to face search problems in domains with limited information, where it is impossible to perform the classical search approach: planning first the whole solution off-line, and then executing such solution. For these domains, interleaving planning and action execution is needed: planning explores some local search space selecting the best action from that limited exploration, and action execution carries out that action, causing changes in the world. The process repeats until achieving a solution. To assure completeness, real-time search has to record in a hash table those actions that have been executed. Often, there is an upper bound on the computing time that the planning phase can take.

Real-time search is also useful for domains with complete information whose search spaces are too large to be explored in practice. In such cases, real-time search is a suitable alternative to the impractical off-line search [7].

A number of algorithms have been proposed to perform real-time search. They are based on the following strategy. Assuming an initial finite heuristic values, the algorithm explores a search space local to the current state and updates its heuristic accordingly, backing up costs. Then, the algorithm moves to the next state where exploration/updating is repeated. The process is iterated until finding a goal state. Some of these algorithms can find optimal solutions: solving repeatedly the same problem instance performance improves and they converge eventually to optimal paths.

* Supported by the Spanish REPLI project TIC-2002-04470-C03-03.

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 379–388, 2006.

In the author's knowledge the first proposed real-time search algorithms were RTA* and LRTA* in the seminal work by Korf [9]. While RTA* performs reasonably well in the first trial, it does not converge to optimal paths. On the contrary, LRTA* converges to optimal paths with a worse performance in the first trial. Both approaches are combined in the HLRTA* algorithm [11].

Instead of updating the heuristic of one (the current) state only, we have proposed to propagate the heuristic change consistently up to k states [6]. We call this idea bounded propagation, and we have applied it to LRTA* [6,5], producing the LRTA*(k) algorithm. Experimental results show that LRTA*(k) causes large benefits on the first solution, convergence and solution stability with respect to LRTA*, at the extra cost of longer planning steps. In this paper, we apply bounded propagation to HLRTA*, producing the new HLRTA*(k) algorithm. This is not a direct extension of the work done in LRTA*(k): HLRTA* deals with two heuristics h_1 (always admissible) and h_2 (admissible under some circumstances) that require a careful handling, since propagation is done on admissible heuristic values. Experimental results show that bounded propagation causes the same kind of benefits on the first solution, convergence and solution stability. These results show that, for $k > 1$, HLRTA*(k) converges in less trials than LRTA*(k), while the contrary was true for the original algorithms.

The paper structure is as follows. In Section 2 we revise existing work on real-time search, with special emphasis on HLRTA*. In Section 3 we present the idea of bounded propagation. In Section 4, we describe the new HLRTA*(k) algorithm, showing its correctness and completeness. In Section 5, we provide experimental results of HLRTA*(k) on classical real-time benchmarks. Finally, in Section 6 we extract some conclusions from this work.

2 Real-Time Search: RTA*, LRTA* and HLRTA*

The state space is defined as (X, A, c, s, G), where (X, A) is a finite graph, $c : A \mapsto [0, \infty)$ is a cost function that associates each arc with a finite cost, s is the start state, and $G \subset X$ is the set of goal states. X is a finite set of states, and $A \subset X \times X - \{(x, x) | x \in X\}$ is a finite set of arcs. Each arc (v, w) represents an action whose execution causes the agent to move from v to w. The state space is undirected: for any action $(x, y) \in A$ there exists its inverse $(y, x) \in A$ with the same cost $c(x, y) = c(y, x)$. The successors of a state x are $Succ(x) = \{y | (x, y) \in A\}$. A path $(x_0, x_1, x_2, \ldots)$ is a sequence of states such that every pair $(x_i, x_{i+1}) \in A$. The cost of a path is the sum of costs of the actions in that path. A heuristic function $h : X \mapsto [0, \infty)$ associates with each state x an approximation $h(x)$ of the cost of a path from x to a goal. $h^*(x)$ is the minimum cost to go from x to a goal. h is said to be admissible iff $h(x) \leq h^*(x)$, $\forall x \in X$. A path $(x_0, x_1, \ldots, x_n)$ with $h(x_i) = h^*(x_i)$, $0 \leq i \leq n$ is *optimal*.

RTA* works as follows. From the current state x, it performs lookahead at depth d, and updates $h(x)$ to the max $\{h(x)$, 2nd min $[k(x, v) + h(v)]\}$, where v is a frontier state and $k(x, v)$ is the cost of the path from x to v. Then, it moves to y, successor of x, with minimum $c(x, y) + h(y)$. This state becomes

the current state and the process iterates, until eventually finding a goal. This process is called a trial.

RTA* is a correct and complete algorithm in finite state spaces with positive edge costs, finite heuristic values and where a goal state is reachable from every state [9]. However, it is unable to improve its performance when solving repeatedly the same instance. This is due to the 2nd min updating strategy. To solve this issue, the LRTA* algorithm was proposed [9]. It behaves like RTA*, except that $h(x)$ is updated to the max $\{h(x)$, min $[k(x,v) + h(v)]\}$. This updating rule assures admissibility, provided the original heuristic was admissible. Then, the updated heuristic can be reused for the next trial. LRTA* is a correct and complete algorithm, that converges to optimal paths when solving repeatedly the same instance, keeping the heuristic estimates of the previous trial.

RTA* works fine in the first trial but there is no guarantee to convergence after successive trials to optimal paths. LRTA* converges but it performs worse than RTA* in the first trial. Would it not be possible to combine them? The answer is the HLRTA* algorithm [11]. HLRTA* keeps for each visited node two heuristic values, h_1 and h_2, which correspond to the heuristic updating of LRTA* and RTA* respectively. In addition, it keeps in $d(x)$ the next current node from x. The interesting result here is that when search has passed through x, and it backtracks to x from $d(x)$ (that is, when it goes back to x through the same arc it used to leave) then h_2 estimate is admissible and it can be used instead of h_1 [11]. Since HLRTA* always keeps admissible heuristic estimates, they are stored between trials and it converges to optimal paths, in the same way that LRTA* does. Experimentally, HLRTA* requires more trials than LRTA* to converge [4].

The HLRTA* algorithm with lookahead at depth 1 and with h admissible appears in Figure 1. Like in [9], we assume the existence of $Succ$ and h_0 functions, which when applied to a state x generate its set of successors and its initial heuristic estimate, respectively. Procedure `HLRTA*` initializes the heuristic estimates h_1 and h_2 of every state to h_0 and 0 respectively. It also initializes $d(x)$ with $null$. Then it repeats the execution of `HLRTA-trial` until convergence of the heuristic function, i.e., until h_1 does not change anymore. At this point, an optimal path has been found. Procedure `HLRTA-trial` performs a solving trial on the problem instance. It initializes the current state x with the start s, and executes the following loop until finding a goal. First, it performs lookahead from x at depth 1, updating its heuristic estimates accordingly (call to function `HLRTA-LookaheadUpdate1`). Second, it selects state y of $Succ(x)$ with minimum value of $c(x,y) + h(y)$ as next state (breaking ties randomly). Third, it executes an action that passes from x to y. At this point, y is the new current state and the loop iterates. Note that the heuristic estimators computed in a trial are used as initial values in the next trial.

Function `HLRTA-LookaheadUpdate1` performs lookahead from x at depth 1. If it happens that search moves back to a state $v \in Succ(x)$ through the same arc it moved forward (that is, $d(v) = x$), then $h_2(v)$ is admissible and $H(v)$ takes it. Otherwise, $H(v)$ takes $h_1(v)$. Then, $h_1(x)$ and $h_2(x)$ are updated accordingly. If $h_1(x)$ changes, the function returns $true$, otherwise it returns $false$.

```
procedure HLRTA*(X, A, c, s, G)
  for each x ∈ X do h1(x) ← h0(x); h2(x) ← 0; d(x) ← null;
  repeat
    HLRTA-trial(X, A, c, s, G);
    until h1 does not change;

procedure HLRTA-trial(X, A, c, s, G)
  x ← s;
  while x ∉ G do
    dummy ← HLRTA-LookaheadUpdate1(x);
    y ← argmin_{w∈Succ(x)}[c(x, w) + H(w)];
    execute(a ∈ A such that a = (x, y));
    d(x) ← y; x ← y;

function HLRTA-LookaheadUpdate1(x): boolean;
  for each v ∈ Succ(x) do
    if d(v) = x then H(v) = h2(v);
    else H(v) = h1(v);
  y ← argmin_{v∈Succ(x)}[c(x, v) + H(v)];
  z ← arg 2nd min_{v∈Succ(x)}[c(x, v) + H(v)];
  if h2(x) < c(x, z) + H(z) then h2(x) ← c(x, z) + H(z);
  if h1(x) < c(x, y) + H(y) then h1(x) ← c(x, y) + H(y); return true;
  else return false;
```

Fig. 1. The HLRTA* algorithm

In addition to the mentioned RTA*, LRTA* and HLRTA*, there are more algorithms for real-time search. The weighted and bounded versions of LRTA* [10]; FALCONS [3], that uses the classical heuristic $g(x) + h(x)$; eFALCONS [4], a hybrid between HLRTA* and FALCONS; a new version of LRTA* [8]; and $\gamma-$Trap [1], an algorithm that controls the exploration vs. exploitation trade-off.

3 Bounded Propagation

As search progresses, heuristic values of visited states are updated. Using the information obtained at the lookahead phase, we can better estimate the cost of reaching a goal from a visited state, and this new information is stored in the heuristic value of that state. This is a general strategy in real-time search algorithms. So far, most algorithms limit heuristic updating to the current state. Recently, we have proposed to propagate consistently the change of heuristic estimate of the current state up to k states, not necessarily distinct. We call this idea *bounded propagation*, since changes are *propagated* up to a *bound* or limit of k states per step. The propagation occurs on the successors of the state that changes its heuristic estimate. If one of these successors changes, this is again propagated on its own successors. This process is iterated with a limit of k considered states. Propagation improves the heuristic quality while keeping it

admissible, so search will find better heuristic estimates in the future states that will help find a solution sooner.

We have applied bounded propagation to LRTA*, producing the LRTA*(k) algorithm (in fact, LRTA* is just a particular case of LRTA*(k) with $k = 1$). LRTA*(k) performance improves greatly with respect to LRTA*, in terms of first solution quality, convergence and solution stability [6]. However, bounded propagation requires longer planning steps, since propagating to k states is computationally more expensive than propagating to one (the current) state. Nevertheless, benefits are important and the extra requirements on planning time are moderate, so if the application can accommodate longer planning steps, the use of bounded propagation is strongly recommended. Precise results depends on the parameter k (the higher k the more benefits at the cost of longer planning steps) and the specific problem considered. It is important to mention that these benefits are asymptotically limited.

Let x be the current state, and let us assume that $h(x)$ changes. Its new value is $h(x) = min_{v \in Succ(x)}[c(x, v) + h(v)]$. Some steps later, if it happens that $h(w)$ changes, $w \in Succ(x)$, then $h(x)$ might change again, so x should be reconsidered. But if w is not the state with $min_{v \in Succ(x)}[c(x, v) + h(v)]$, no matter the change in $h(w)$, $h(x)$ will not change, because the minimum of its successors has not changed its heuristic. That state is called a support for $h(x)$.

Formally, we say that state y is *support* of $h(x)$, denoted $y = supp(x)$, iff $y = argmin_{v \in Succ(x)}\ [c(x, v) + h(v)]$. The previous paragraph describes a simple property of bounded propagation: if state y changes its heuristic estimate, only those states x successors of y such that y is their support could change its heuristic estimate. A successor state z not supported by y will not change: z is supported by other state and as far this state does not change its heuristic, z will not change. Bounded propagation can benefit from this property, by propagating those states which are supported by the state that has changed its heuristic. The use of supports requires a table that records for each expanded state its corresponding support.

Since propagation is limited up to k states, it is meaningful to consider which states are the most adequate to be updated. Originally, we decided to limit propagation to states already expanded in the current trial. This and other alternatives are discussed in [5].

4 HLRTA*(k)

HLRTA*(k) is the algorithm that combines HLRTA* with bounded propagation. The rationale for this combination is as follows. First, it is reasonable to expect that HLRTA* would benefit from bounded propagation in the same way that LRTA* does, improving first solution quality, convergence and solution stability. Second, it has been observed experimentally that HLRTA* convergence is slower than LRTA* [4]. Since bounded propagation improves convergence, its combination with HLRTA* could be quite beneficial. Experimental results, reported in Section 5, clearly justify these assumptions.

The HLRTA*(k) algorithm appears in Figure 2. The main differences with HLRTA* appear in the `HLRTA-Trial` procedure: `HLRTA-LookaheadUpdate1` is replaced by `HLRTA-LookaheadUpdateK`. This procedure performs the updating of the current state plus bounded propagation. HLRTA* updating involves the first and second minima of heuristic values among successors of the current state, while propagation considers the first minimum only. These strategies are considered in `HLRTA-LookaheadUpdate2min` and `HLRTA-LookaheadUpdate1min`. This last function returns *true* when h_1 has changed as consequence of propagation, $false$ otherwise.

Bounded propagation is done in the `HLRTA-LookaheadUpdateK` procedure. If x is the current state of search, it initializes queue Q with capacity for k elements. After performing the 2nd minima update of x, it enters the following loop. While Q is not empty, it extracts the first state v from Q and performs 1st minima updating. If this updating has caused some change in $h_1(v)$, it has to be propagated, so the successors of v are entered in Q, provided the following conditions: there is room ($cont > 0$), a successor w has some chance to modify its heuristic estimator ($v = supp(w)$), and w belongs to the path of expanded states in the current or previous executions. As final remark, we stress the point that $d(x)$, the state that search moves from the current state, gets value *null* before the updating process. This is required to avoid that an old value of $d(x)$ could cause an erroneous propagation of heuristic estimates before it takes the next state to which search moves on.

It is not difficult to see that HLRTA*(k) is a complete algorithm. Theorem 2 of [11] on the completeness of HLRTA* remains valid, since stored heuristic values increase with the length of the path. To prove convergence to optimal paths it is required to assure that h_1 remains admissible after bounded propagation, with the following lemma.

Lemma 1(from [2], modified)**.** Let $x \in X - G$, h_1, h_2 and H as in HLRTA*. If $h_1(x) \leftarrow max(h_1(x), min_{v \in Succ(x)}(c(x,v) + H(v))$ then $h_1(x) \leq h^*(x)$.

Proof. If $h_1(x) \geq min_{v \in Succ(x)}(c(x,v) + H(v))$ there is nothing to prove. Otherwise, there is an optimal path from x to a goal that passes through a successor w. Then, $h^*(x) = c(x,w) + h^*(w) \geq c(x,w) + H(w)$. To see this, remember that $H(w)$ may be $h_1(w)$ (in which case $H(w)$ is obviously admissible) or $h_2(w)$. In general, $h_2(w)$ is not admissible, except through the path from x to w, provided that last time w was visited search moved to x as next state. In that particular case $h_2(w)$ is admissible [11]. But this is the case in which $H(w)$ can take value $h_2(w)$, when $d(w) = x$. Therefore, $H(w)$ is admissible through the path from x to w, so we can write $h^*(x) = c(x,w) + h^*(w) \geq c(x,w) + H(w)$. In particular, this is true for the minimum $c(x,v) + H(v)$ among successors of x, that is, $h^*(x) \geq min_{v \in Succ(x)}(c(x,v) + H(v))$. So $h_1(x) \leq h^*(x)$.

With this result, the proof of HLRTA* convergence to optimal paths (Theorem 5 of [11]) is also valid for HLRTA*(k), since it requires admissibility of h_1 stored heuristic. Therefore, HLRTA*(k) is a correct and complete algorithm that converges to optimal paths over repeated trials on the same problem instance.

```
procedure HLRTA*(k)(X, A, c, s, G)
  for each x ∈ X do h1(x) ← h0(x); h2(x) ← 0; supp(x) ← null;
  path ← ⟨s⟩;
  repeat
    HLRTA(k)-trial(X, A, c, s, G, k);
    until h1 does not change;

procedure HLRTA(k)-trial(X, A, c, s, G, k)
  x ← s;
  while x ∉ G do
    d(x) ← null;
    HLRTA-LookaheadUpdateK(x, k, path);
    y ← argmin_{w∈Succ(x)}[c(x, w) + H(w)];
    execute(a ∈ A such that a = (x, y));
    path ← add-last(path, y);
    d(x) ← y;
    x ← y;

procedure LookaheadUpdateK(x, k, path)
  Q ← ⟨x⟩;
  cont ← k − 1;
  HLRTA-LookaheadUpdate2min(x);
  while Q ≠ ∅ do
    v ← extract-first(Q);
    if HLRTA-LookaheadUpdate1min(v) then
      for each w ∈ Succ(v) do
        if w ∈ path ∧ cont > 0 ∧ v = supp(w) then
          Q ← add-last(Q, w);
          cont ← cont − 1;

procedure HLRTA-LookaheadUpdate2min(x)
  for each v ∈ Succ(x) do
    if d(v) = x then H(v) = max{h1(v), h2(v)};
    else H(v) = h1(v);
  z ← arg 2nd min_{v∈Succ(x)}[c(x, v) + H(v)];
  if h2(x) < c(x, z) + H(z) then h2(x) ← c(x, z) + H(z);

function HLRTA-LookaheadUpdate1min(x): boolean;
  for each v ∈ Succ(x) do
    if d(v) = x then H(v) = max{h1(v), h2(v)};
    else H(v) = h1(v);
  y ← argmin_{v∈Succ(x)}[c(x, v) + H(v)];
  supp(x) ← y;
  if h1(x) < c(x, y) + H(y) then h1(x) ← c(x, y) + H(y); return true;
  else return false;
```

Fig. 2. The HLRTA*(k) algorithm

5 Experimental Results

We compare the performance of HLRTA*(k), for different values of k, with RTA* (first trial only), HLRTA*, LRTA*, LRTA*(k) and FALCONS (first trial, convergence and stability). In HLRTA*(k) and LRTA*(k) we maintain the table of supports and heuristics values between trials [5]. As benchmarks we use these four-connected grids where an agent can move one cell north, south, east or west: Grid35, grids of size 301×301 with a 35% of obstacles placed randomly. In this type of grid heuristics tend to be only slightly misleading. Grid70, grids of size 301×301 with a 70% obstacles placed randomly. In this type of grid heuristics could be misleading. Maze, acyclic mazes of size 181×181 whose corridor structure was generated with depth-first search. Here heuristics could be very misleading.

Results are averaged over 1000 different instances. In grids of size 301×301 the start and goal state are chosen randomly assuring that there is a path from the start to the goal. In mazes, the start is (0,0), and the goal is (180,180). As initial heuristic we use the Manhattan distance. Results for HLRTA*(k), RTA*, HLRTA* and FALCONS appear in Table 1. For LRTA* and LRTA*(k) results appear in Table 2. The results are presented in terms of solution cost ($\times 10^3$), number of expanded states ($\times 10^3$) and time per step ($\times 10^{-6}$ seconds), for the first trial; convergence (trials $\times 10^3$ to converge); and stability indexes [10].

In the first trial, the effect of propagation is better in HLRTA*(k) than in LRTA*(k). For low values of k, the solution cost obtained with HLRTA*(k) is better than the solutions obtained by LRTA*(k) and the others algorithms for all benchmarks (except for Grid35 and $k = 6$). For HLRTA*(k) in Grid35 and Grid70, as the value of k increases the cost of the solution improves, but in Maze the solution cost is similar for all values of k. Therefore, in Maze it is better to use small values of k, since high values of k means more computation per step. For high values of k the cost of solution obtained with LRTA*(k) and HLRTA*(k) are similar. The smallest solution cost is obtained by HLRTA*(k) and LRTA*(k) with $k = \infty$. Comparing with RTA*, HLRTA*(k) algorithm finds better solutions from $k = 6$ on in Grid35, and $k = 1$ on in Grid70 and Maze. Comparing with FALCONS, HLRTA*(k) always produces better solutions in Grid35, Grid70 and Maze.

Considering convergence, HLRTA*(k) obtains optimal solutions with less cost than HLRTA*, LRTA* and FALCONS for all the values of k tested on the three benchmarks. The effect of propagation is better in HLRTA*(k) than in LRTA*(k); thus with low values of k, HLRTA*(k) obtains better results than LRTA*(k). With high values of k, the solution cost obtained with LRTA*(k) and HLRTA*(k) are similar. Solution cost decreases monotonically as k increases. We observe that the worse the heuristic information is, the better HLRTA*(k) behaves with respect to its competitors (results are better in Maze than in Grid70, and in Grid70 than in Grid35). Considering trials to convergence, HLRTA*(k) requires substantially less trials than HLRTA*. The number of trials decreases steadily as k increases, from approximately a third of the trials with $k = 6$. Comparing with FALCONS, it performs better for high values of k in Grid35,

Table 1. Results for the first trial (left), convergence (middle) and stability (right) for HLRTA*(k), HLRTA*, RTA* and FALCONS. Average over 1000 instances.

Grid35												
k	Cost%	Mem.%	T/Step%	Cost%	Trials%	Mem.%	T/Step%	IAE	ISE	ITAE	ITSE	SOD
100%	46.8	4.6	0.79	6202.2	5.0	45.1	0.8	$\times 10^6$	$\times 10^9$	$\times 10^8$	$\times 10^{11}$	$\times 10^5$
1 (HLRTA*)	100%	100%	100%	100%	100%	100%	100%	3.8	20.1	54.5	75.5	12.1
6	37%	87%	157%	39%	34%	100%	146%	1.6	6.1	8.4	13.3	5.0
15	24%	85%	193%	25%	20%	100%	186%	1.0	3.6	3.3	5.6	3.2
500	21%	123%	358%	5%	3%	99%	477%	0.3	1.1	0.1	0.2	0.7
∞	20%	126%	480%	1.3%	0.6%	93%	1355%	0.07	0.5	0.006	0.02	0.2
RTA*	66%	69%	39%	-	-	-	-	-	-	-	-	-
FALCONS	2058%	191%	54%	65%	13%	244%	55%	3.7	3898.1	6.3	88.3	10.6
Grid70												
100%	40.0	1.5	0.75	669.8	0.6	1.66	0.77	$\times 10^3$	$\times 10^6$	$\times 10^4$	$\times 10^7$	$\times 10^2$
1 (HLRTA*)	100%	100%	100%	100%	100%	100%	100%	125.4	2037.1	719.4	1598.1	512.2
6	51%	102%	167%	29%	25%	100%	126%	58.9	498.0	127.5	188.9	172.2
15	38%	103%	211%	18%	14%	100%	159%	43.6	289.4	62.9	92.4	118.8
500	10%	102%	574%	4%	2%	100%	450%	10.1	25.0	4.3	5.9	28.4
∞	7.4%	107%	3157%	1.0%	0.6%	100%	1723.3%	2.3	4.7	0.3	0.5	0.2
RTA*	106%	100%	43%	-	-	-	-	-	-	-	-	-
FALCONS	194%	102%	47%	96%	28%	104%	49%	480.9	17268.5	4088.3	46665.2	1244.5
Maze												
100%	11.4	7.2	0.71	11227.2	3.2	15.4	0.69	$\times 10^5$	$\times 10^9$	$\times 10^6$	$\times 10^1 0$	$\times 10^4$
1 (HLRTA*)	100%	100%	100%	100%	100%	100%	100%	0.3	0.3	0.3	0.2	1.3
6	102%	101%	159%	18%	18%	100%	113%	0.25	0.3	0.07	0.07	0.9
15	99%	99%	212%	8%	7%	101%	122%	0.25	0.4	0.05	0.07	1.0
500	98%	98%	745%	1%	0.4%	101%	717%	0.25	0.4	0.04	0.08	1.1
∞	100%	100%	1644%	0.3%	0.08%	101%	11506%	0.25	0.4	0.04	0.08	1.0
RTA*	309%	100%	47%	-	-	-	-	-	-	-	-	-
FALCONS	772%	147%	52%	193%	25%	120%	52%	189.4	1136.2	7072.6	43695.6	680.8

Table 2. Results for the first trial (left), convergence (middle) and stability (right) for LRTA* and LRTA*(k). Average over 1000 instances.

Grid35												
k	Cost%	Mem.%	T/Step%	Cost%	Trials%	Mem.%	T/Step%	IAE	ISE	ITAE	ITSE	SOD
100%	46.8	4.6	0.79	6202.2	5.0	45.1	0.8	$\times 10^6$	$\times 10^9$	$\times 10^8$	$\times 10^{11}$	$\times 10^5$
1 (LRTA*)	147%	104%	43%	105%	51%	100%	45%	5.2	56.5	38.1	130.9	15.9
6	35%	75%	87%	48%	34%	100%	82%	2.2	9.3	11.7	24.5	6.5
15	26%	79%	113%	29%	21%	99%	104%	1.3	5.1	4.4	8.8	3.9
500	21%	79%	249%	5%	3%	100%	308%	0.3	1.3	0.1	0.3	0.8
∞	20%	132%	336%	1.3%	0.6%	93%	998%	0.07	0.5	0.006	0.02	0.2
Grid70												
100%	40.0	1.5	0.75	669.8	0.6	1.66	0.77	$\times 10^3$	$\times 10^6$	$\times 10^4$	$\times 10^7$	$\times 10^2$
1 (LRTA*)	366%	99%	45%	118%	52%	100%	48%	509.1	24227.7	2029.2	9145.2	1174.8
6	125%	98%	76%	53%	29%	100%	75%	200.0	3110.0	600.0	1520.0	533.0
15	67%	100%	100%	31%	17%	100%	93%	100.0	1110.0	231.0	501.0	327.0
500	13%	103%	317%	5%	3%	100%	266%	20.0	64.0	8.1	16.4	51.2
∞	7%	107%	2093%	1%	0.6%	100%	1296%	2.0	4.7	0.3	0.5	0.2
Maze												
100%	11.4	7.2	0.71	11227.2	3.2	15.4	0.69	$\times 10^5$	$\times 10^9$	$\times 10^6$	$\times 10^1 0$	$\times 10^4$
1 (LRTA*)	5145%	114%	49%	247%	50%	90%	49%	221.6	5338.3	6952.1	120297.1	1331.1
6	1498%	106%	83%	101%	24%	96%	89%	87.3	1100.0	1580.0	15000.0	531.0
15	819%	102%	113%	57%	12%	99%	117%	50.5	485.0	556.0	4010.0	301.8
500	241%	101%	445%	9%	2%	100%	434%	8.0	43.1	14.3	57.6	42.6
∞	102%	101%	3743%	0.4%	0.1%	101%	10128%	0.3	0.4	0.05	0.08	1.1

and gets better results in Grid70 and Maze for all values of k. Comparing with LRTA*, it perform better for all values of k. Comparing with LRTA*(k) it performs equal or better for all values of k. The good results of HLRTA*(k) come at the cost of extra computation, that is, longer planning time. It is worth noticing that low values of k generate large improvements in solution cost and number of trials, with a limited effect in planning time per step. For instance, with $k = 6$, the total cost to converge to optimal path is divided by a factor of approximately 3, the number of trials is divided by a factor of approximately 4, at the cost of increasing the time per step by a factor around 1.3.

To measure solution stability we computed the indices IAE, ISE, ITAE, ITSE, and SOD [10]. Smaller values mean better stability of solutions. HLRTA*(k) outperforms HLRTA*, LRTA* and LRTA*(k) for all k values tested in all indices for the three benchmarks. For $k = \infty$ LRTA*(k) is similar to HLRTA*(k). Something similar happens when comparing with FALCONS, except for index ITAE on Grid35 for $k = 6$, where FALCONS obtains better results.

6 Conclusions

As in the case of LRTA*, bounded propagation of heuristic changes is quite beneficial when applied to HLRTA*, improving first solution, convergence and solution stability, at the extra cost of longer planning steps. Experimentally, for $k > 1$, HLRTA*(k) requires less trials than LRTA*(k) to converge to optimal paths, when the contrary happens for $k = 1$.

Acknowledgments

Authors thank anonymous reviewers for their detailed reviews and constructive comments.

References

1. V. Bulitko. Learning for adaptive real-time search. *The Computing Research Rep. (CoRR): cs.DC/0407017*, 2004.
2. S. Edelkamp and J. Eckerle. New strategies in learning real time heuristic search. In *Proc. AAAI Workshop on On-Line Search*, pages 30–35, 1997.
3. D. Furcy and S. Koenig. Speeding up the convergence of real-time search. In *Proc. AAAI*, pages 891–897, 2000.
4. D. Furcy and S. Koenig. Combining two fast-learning real-time search algorithms yields even faster learning. In *Proc. 6th European Conference on Planning*, 2001.
5. C. Hernandez and P. Meseguer. Improving convergence of lrta*(k). In *Proc. IJCAI Workshop on Planning and Learning in a Priori Unknown or Dynamic Domains*, pages 69–75, 2005.
6. C. Hernandez and P. Meseguer. Lrta*(k). In *Proc. IJCAI*, pages 1238–1243, 2005.
7. K. Knight. Are many reactive agents better than a few deliberative ones? In *Proc. 13th IJCAI*, pages 432–437, 1993.
8. S. Koenig. A comparison of fast search methods for real-time situated agents. In *Proc. 3rd AAMAS*, pages 864–871, 2004.
9. R. E. Korf. Real-time heuristic search. *Artificial Intelligence*, 42(2-3):189–211, 1990.
10. M. Shimbo and T. Ishida. Controlling the learning process of real-time heuristic search. *Artificial Intelligence*, 146(1):1–41, 2003.
11. P. E. Thorpe. A hybrid learning real-time search algorithm. Master's thesis, Computer Science Dep., UCLA, 1994.

Real Time Image Segmentation Using an Adaptive Thresholding Approach*

P. Arques, F. Aznar, M. Pujol, and R. Rizo

Department of Computer Science and Artificial Intelligence
University of Alicante
{arques, fidel, mar, rizo}@dccia.ua.es

Abstract. The aim of image segmentation is the partition of the image in homogeneous regions. In this paper we propose an approximation based on Markov Random Fields (MRF) able to perform correct segmentation in real time using colour information. In a first approximation a simulated annealing approach is used to obtain the optimal segmentation. This segmentation will be improved using an adaptive threshold algorithm, to achieve real time. The experiment results using the proposed segmentation prove its correctness, both for the obtained labelling and for the response time.

Keywords: Region Segmentation, Adaptive Thresholding, Simulated Annealing, Real Time, Markov Random Field.

1 Introduction

Image segmentation is the process that divides an image into a set of disjointed regions whose features, such as intensity, colour, texture, etc. are similar. Image segmentation can be considered as a labelling task with a set of homogeneous states and a discrete set of labels, therefore, the Markov Random Field theory can be applied to characterize the problem of trying to find an optimal solution.

In general, segmentation algorithms are based on two important criteria to be considered: the first is the homogeneity of the region and the second is the discontinuity between adjacent, disjointed regions. Although there are a wide variety of image interpretation techniques that are well treated in [1], [2], it is a very difficult task to satisfy all the properties for the optimal set of segmented regions. Normally, the resulting segmented image depends on a set of predetermined threshold values, so algorithms frequently fails to merge regions that must be separated or fails to split the regions that need to be together. These problems are due to the fact that the information about region uniformity or about discontinuity between different regions is not well integrated into segmentation algorithms. Therefore, it is interesting to incorporate features which are robust to some distortion level, so that solid results are produced.

* This work has been financed by the Generalitat Valenciana project GV04B685.

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 389–398, 2006.

Due to the large number of pixels on which the MRF is to be defined, it usually takes a long computional time to obtain optimal labels, which makes the MRF model difficult to apply to real scene domains. Most current approaches to segment colour images [3] [4] do not consider the processing time to apply it in real time systems. Image segmentation is a previous process to recognize objects. In our case, object recognition will be integrated in an autonomous agent to recognize objects, so real time in the segmentation process is needed.

The MPMT algorithm [5], [6], is used to obtain a preliminary segmentation and subsequently a MRF model in the segmented regions set is defined, so that the number of states that should be considered is significatively reduced.

To measure the region uniformity several first order statistics such as intensity average and variance are considered. Next, a border process will be introduced to reflect the measure of discontinuity along the common border between two adjacent regions. Before applying the optimal segmentation algorithms a process to remove isolated pixels is executed.

The process of searching for an optimal segmentation is time consuming so, it is a bottleneck for real time systems. In a first approximation, optimal labelling is proposed by minimizing an energy function using the simulated annealing algorithm.

In a second approximation, named Adaptive Thresholding, similar results were obtained for the optimal segmentation, but the processing time is much shorter. Therefore, this process could be used in real time systems.

2 Previous Segmentation

For the preliminary segmentation step, the Modified Partition Mode Test, MPMT [5], [6], has been chosen due to two main reasons: ease of implementation and it provides the necessary information to begin use of the MRF model. The segmentation algorithm based on modified partition mode test MPMT divides the input image into subareas (called windows). The MPMT algorithm is applied to assign a label to each window. A unique partition mode is assigned to each window, and also the windows are chosen so that they overlap. In this way, each pixel is covered by several different windows. The implemented algorithm uses overlapping 2x2 windows as basic subareas. The basic segmentation of subareas is done assigning one of the 12 partition modes to each window (See figure 1).

Nevertheless, the presegmentation MPMT algorithm provides an oversegmentation by nature (it only scans the image once, while the division into regions is

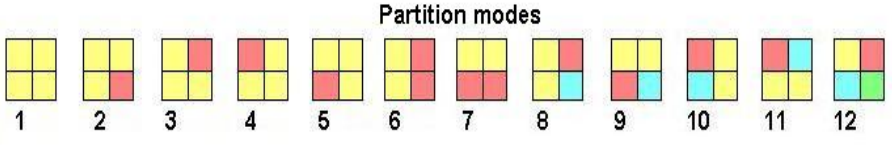


Fig. 1. Twelve partition modes used in the algorithm

done). Sometimes, this over-segmentation also produces some isolated points in the image. Isolated points add noise and complexity, due to the fact that they dramatically increase the number of regions.

Although in later segmentation processes these isolated point are labelled correctly, a new postprocess is performed (See table 1) to speed up the segmentation algorithm.

Table 1. MPMT Postprocess

```
ALGORITHM Postprocess MPMT
If Area(R_i)==1 then
   For all Neighbour(R_i)
      Calculate Distance(R_i ,R_j) ∀j
      Choose Minimum Distance
      If Minimum Distance < Threshold then
         Label(R_i)=Label(R_j)
         Calculate
         (Avg_r(R_j), Avg_g(R_j), Avg_b(R_j))
         Area(R_j)++
      else
         label(R_i)=label(R_j)
         Area(R_j)++
end ALGORITHM
```

This algorithm introduces a higher definition in the object characteristics in the image. The postprocess performs a single scan for the complete image, selecting those regions made up of 1 pixel, in other words, isolated points; next the distance between each isolated point to all its neighbours is calculated and the minimum distance is chosen. At this point two cases need to be considered:

- **Point with a wrong label.** The distance between this isolated point and one of its neighbouring regions is smaller than the threshold used to join regions. In this case, the isolated point is labelled using the neighbouring region label and the average intensity of the region is calculated taken into consideration the value of the isolated point.
- **Isolated point.** The minimum distance between this isolated point and one of its neighbouring regions is greater than the threshold used to join regions. In this case the isolated point is labelled again using the nearest region label. It is supposed that this point is just noise.

3 Obtaining Regions Using Simulated Annealing

Simulated annealing algorithm is based on randomization techniques and it is related to iterative improvement algorithms [6]. This algorithm is frequently used to segment regions to obtain an optimal labelling given an energy function, as can be seen in [7], [8], [9]. Next, an energy function for segmentation by simulated annealing is presented. It is suitable for colour images.

For merging or splitting regions, similar colour criteria are applied in the definition of the energy function. In RBG colour space, the difference between two colours is obtained calculating the euclidean distance.

In our segmentation model, the energy function is defined as follows:

$$U(CE/R, F) = \sum_{c} V_c(CE/R, F) \tag{1}$$

where CE is the current label configuration, R is the region process (measure of the regions homogeneity) and F is the boundary process (measure of the discontinuity in the limits of adjacent regions).

The initial energy function is calculated by adding the intensity differences between adjacent regions.

Let $F = \{F_{red}, F_{green}, F_{blue}\}$ the set of region processes.

where F_i = average intensity in region i.

The spectrum features in region R_i could be defined as:

$F_i\{F_{red}, F_{green}, F_{blue}\}$

To achieve an optimal image segmentation the following restrictions are imposed:

- A segmentation on simple regions is caused by uniform spectrum features.
- In the common border between two different segmented regions strong discontinuities of intensity must exist.

Given a system of neighbourhoods on a lattice, we define a clique C as either a single site, or a set of sites of the lattice, in such a way that all the sites that belong to C are neighbours of each other.

So, the clique function [6] referring to region c is defined as:

$$\sum_{R_i \in C} \sum_{\substack{R_j \in C \\ R_j \neq R_i}} |F_i - F_j| \tag{2}$$

Function (2) calculates the initial energy of the adjacency graph. The frontier process is added to this equation, that is, the impact caused by merging two regions, in the individual process of each region and in the global process of the adjacency graph. The resulting energy function, the consequence of merging two regions, is defined as:

$$\sum_{R_i \in C} \sum_{\substack{R_j \in C \\ R_j \neq R_i}} |F_i - F_j| + NAdj_{i,j}|F_i - F_j| \tag{3}$$

Where $NAdj_{i,j}$ is the maximum value of adjacency of regions R_i and R_j.

Two regions should be merged only if their union decreases the global energy function, in other cases they should remain as independent regions.

The algorithm used to minimize the energy function is the *Simulated Annealing* algorithm [10], a robust method which allow us to perform an optimal image segmentation, as shown in figure 2, where *frontier segmentation* represents

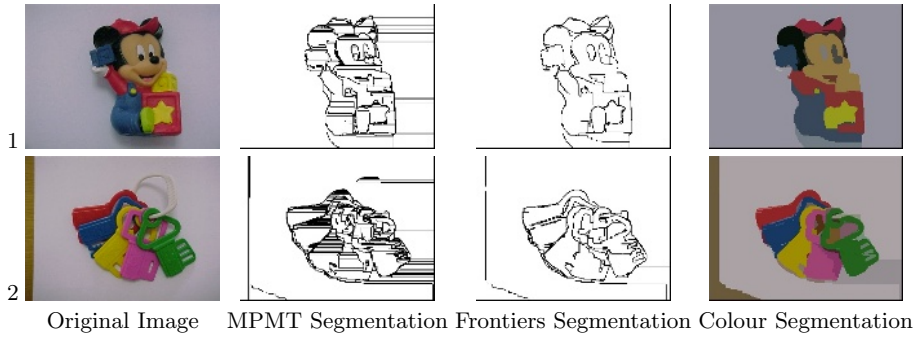


Fig. 2. Segmentation obtained using the proposed energy function

the object's frontiers when the algorithm is performed, and *colour segmentation* represents the same image as frontier segmentation but with the recuperation of colour of the objects detected by the algorithm.

4 Adaptive Thresholding

A natural way to segment an image into regions is using the thresholding technique, that is, dividing the image into bright and dark regions.

Thresholding generates binary images from grey level images, converting pixels which are below a certain threshold to 0 and those which are above the threshold to 1.

There are several approximations to automatically calculate an image threshold [12]. The *Otsu approximation* [13] assumes that the histogram is the addition of two gaussians. This threshold should minimize the weighted sum of the variance of each present object. It is supposed that the closer the gaussian to the real histogram, then the smaller the standard deviation.

The Otsu approximation has been modified to deal with RGB images. To obtain an optimal threshold, the average and the variance of each channel (red, green and blue) is used, among a set of images with similar illumination levels. In this way, the optimal threshold is the average value of standard deviation of each channel.

The threshold calculation consists of:

– Selecting a group of images (20 images have been used in our experiments) with the same luminosity as the image to be segmented. For each image the probabilities of each channel: red, green and blue, are calculated.

$$q_{channel}(t) = \sum_{i=0}^{nxm} P_{channel}(i) \tag{4}$$

where mxn are the image dimensions. The average of each channel probability is also calculated,

$$\mu_{channel} = \frac{1}{N} \sum_{t=0}^{N} q_{channel}(t) \tag{5}$$

where N is the maximum intensity value of each channel.

And finally, the standard deviation for each channel Red, Green and Blue is also calculated.

$$\sigma_{channel} = \frac{1}{N} \sqrt{\sum_{i=0}^{nxm} (P_{channel}(i) - \mu_{channel})^2} \tag{6}$$

– Once the data for all the samples are calculated, the threshold is obtained as the average of each channel standard deviation.

$$Threshold = \frac{1}{3 * p} \sum_{i=0}^{p} (\sigma_{red}^{i} + \sigma_{green}^{i} + \sigma_{blue}^{i}) \tag{7}$$

where p is the sample size and σ_{red}^{i} is the standard deviation for the red channel of the $i-$th image of the sample $i = 1..p$.

The value obtained after the process, supplies a suitable threshold to segment images in an adequate way.

Thanks to the use of real images the suitable threshold can be automatically calculated when the illumination conditions vary. As we are working with the standard deviation average of each channel, the actual property that is searched is the maximum difference of each channel.

4.1 Region Segmentation Using Adaptive Thresholding

The adjacency graph resulting from the MPMT algorithm and the isolated points removal, allow the system to obtain a certain label configuration and a neighbourhood definition. By using the adaptive thresholding algorithm a label configuration will be obtained using the previously defined threshold as a criteria for merging or dividing regions.

The *Adaptive Thresholding* algorithm consists of the following steps: taking the label configuration obtained from the MPMT pre-segmentation algorithm and the isolated points removal, then the adjacency graph must be scanned and the distance from one region to its neighbouring regions calculated. From the set of regions whose distance is lower than the calculated threshold, the smallest distance region is chosen. The adjacency graph is re-calculated and the process of region comparison continues. The process is repeated until all the regions with lower distance than the threshold are joined (See table 2).

Where CE is the label configuration obtained by the MPMT process, U is the calculated threshold for the images of this level of illumination and $Adj(R_i, R_j)$ is the next function:

$$Adj(R_i, R_j) = \begin{cases} 1 \text{ if regions } R_i \text{ y } R_j \text{ are adjacents.} \\ 0 \qquad\qquad\quad \text{in other case} \end{cases} \tag{8}$$

Table 2. Algorithm to merge regions

```
ALGORITHM MergingRegions
∀ R_i ∈ CE
∀ R_j ∈ CE so that R_j ≠ R_i
    If Adj(i, j) then
        Calculate Distance(R_i, R_j)
        Choose LowestDistance(R_i, R_j) so that Distance(R_i, R_j) < Threshold
            MergeRegions(R_i, R_j)
            Re − calculate CE
Repeat while CE is modified
EALGORITHM
```

Where $R = \{R(r), R(g), R(b)\}$ represents the regions processes set and R_i is the average value of colour of region i. The spectral features of region R_i should be defined as: $R_i = \{R_i(r), R_i(g), R_i(b)\}$

The distance between two adjacent regions is defined as:

$$Distance(R_i, R_j) = \sqrt{(R_i(r) - R_j(r))^2 + (R_i(g) - R_j(g))^2 + (R_i(b) - R_j(b))^2} \quad (9)$$

Where $LowestDistance(R_i, R_j)$ is the process to calculate the lowest distance between region i and all its neighbours, $MergeRegions(R_i, R_j)$ is the process to calculate the values of the new region features, $Re - calculate\, CE$ is the process which modifies the adjacent graph with the new region.

5 Experimentation

A comparison between the two approaches has been carried out. This comparison is based on the resulting number of regions and the computational time required to execute each proposed algorithm.

Several kind of images have been chosen, figures 2 and 3 show coloured objects with well-defined shapes, figure 4.1 shows a real outdoor image with a traffic sign

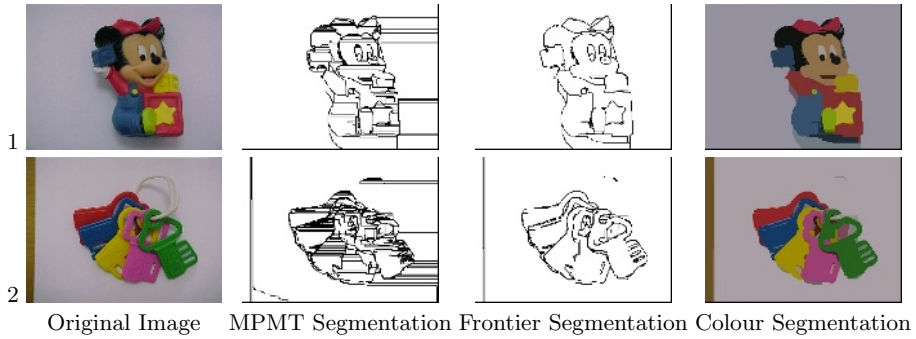


Fig. 3. Segmentation using the adaptive thresholding algorithm

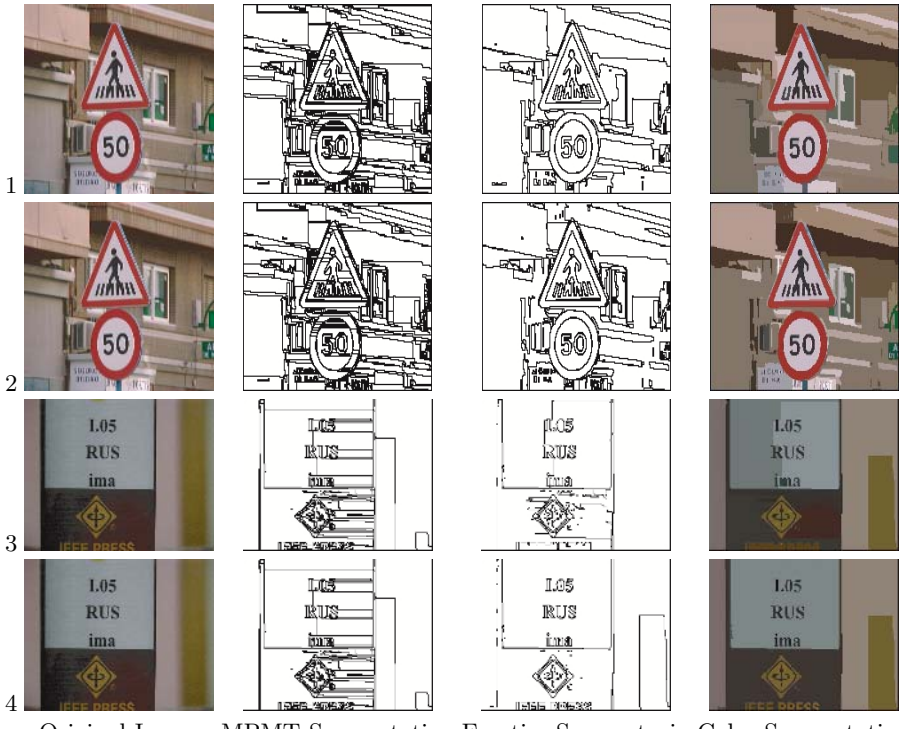


Fig. 4. Image Segmentation

Table 3. Two segmentation approximation comparison. The tests were conducted with a Pentium 4 Processor, with 512 RAM MB, and 2.4 GHz.

	MPMT		Simulated Annealing		Adaptive Thresholding	
Image	Labels	Time	Labels	Time	Labels	Time
2.1 and 3.1	280	209ms	62	1.285 ms	46	340 ms
2.2 and 3.2	488	247ms	97	4.517 ms	32	459 ms
4.1 and 4.2	1155	779ms	229	67.851 ms	67	2.393 ms
4.3 and 4.4	441	183ms	98	2.939 ms	54	398 ms

and figure 4.3 shows a book spine. Experimentation has been carried out using real images in JPEG format obtained by a digital camera.

Simulated annealing algorithm has been applied to figures 2.1, 2.2, 4.1 and 4.3 and adaptive thresholding algorithm has been applied to figures 3.1, 3.2, 4.2 and 4.4. Table 3 presents the comparisons in processor time and in the number of regions for each image. As shown, time needed is dependant on the complexity of the image.

As can be seen in the above images the optimal segmentation using the *simulated annealing* and the segmentation obtained by the *adaptive thresholding*

approximation are similar. However, temporal costs are quite different. The adaptive thresholding algorithm considerably reduces temporal costs in the segmentation process, decreasing by up to 30 times, the processing time to obtain an optimal segmentation. Therefore, an adaptive thresholding algorithm could be more suitable to use in systems which need real time segmentation.

6 Conclusions

Image segmentation could be defined as an MRF problem. Most of todays colour image segmentation approximations do not consider the processor time for real time applications. In our case, we have an autonomous agent for recognizing objects. So that, segmentation is a previous process for object recognition and it should be executed in real time.

Starting form MRF theory to carry out a robust image segmentation, defining a neighbourhood system and an adjacency graph for images, two algorithms have been applied to obtain an optimal labelling.

The first algorithm uses a *simulated annealing* approach to minimize a proposed energy function for colour image segmentation. Using *simulated annealing* an optimal segmentation is obtained although it requires a longer processor time. Nevertheless, in real time systems, the optimal solution if often relaxed to obtain shorter temporal costs. So, a second approach based on an *adaptive threshold* criteria has been proposed to merge or divide regions. Although , this second algorithm can not guarantee the return of an optimal segmentation, it provides satisfactory results. Moreover, it is possible to use it in a real time segmentation, due to its low temporal cost.

Satisfactory results have been obtained proving the proposed algorithm validity. Future works will focus on object recognition using the presented segmentation approach.

References

1. Pal, N., Pal, S.: A review on image segmentation techniques. Pattern Recognition **26** (1993) 1294–1993
2. Muoz, X., Freixenet, J., Cufi, X., Marti, J.: Strategies for image segmentation combining region and boundary information. Pattern Recognition Letters **24** (2003) 375–392
3. Martinez-Uso, A., Pla, F., Garcia-Sevilla, P.: Color image segmentation using energy minimization on a quadtree representation. International Conference on Image Analysis and Recognition ICIAR'04. Porto (2004)
4. Kim, B., Shim, J., Park, D.: Fast image segmentation based on multi-resolution analysis wavelets. Pattern Recognition Letters **24** (2003) 2995–3006
5. Suk, M., Chung, S.: A new image segmentation technique based on partition mode test. Pattern Recognition **16** (1983) 469 – 480
6. Pujol, M.: Incorporacion de caracteristicas en la funcion de energia para segmentacin de imagenes usando campos aleatorios de Markov. PhD thesis, Departamento de Ciencia de la Computacion e Inteligencia Artificial. Universidad de Alicante (2000)

7. Arques, P., Pujol, M., Rizo, R.: Robust segmentation of scenes with colour mark. Frontiers in Artificial Intelligence and Applications. Artificial Intelligence Research and Develop **100** (2003) 149–159
8. Lievin, M., Luthon, F.: Nonlinear color space and spatiotemporal mrf for hierarchical segmentation of face features in video. IEEE Transactions on Image Processing **13** (2004) 1–9
9. Luo, J., Guo, C.: Perceptual grouping of segmented regions in color images. Pattern Recognition **36** (2003) 2781–2792
10. Azencott, R.: Simulated Annealing. Parallelization Tecniques. John Wiley & Sons (1999)
11. Sahoo, P., Soltani, S., Wong, A., Chen, Y.C.: A survey of thresholding techniques. Computer Vision, Graphics, and Image Processing **41** (1988) 233–260
12. Weszka, J.S., Rosenfeld, A.: Threshold evaluation techniques. IEEE Transactions on Systems, Man and Cybernetics **SCM-8** (1978) 622–629
13. Otsu, N.: A threshold selection method from gray-level histograms. IEEE Transactions on Systems, Man and Cybernetics **SCM-9** (1979) 62–66

Scheduling a Plan with Delays in Time: A CSP Approach*

Eliseo Marzal[1], Eva Onaindia[1], Laura Sebastia[1], and Jose A. Alvarez[2]

[1] Universidad Politecnica de Valencia (Spain)
{emarzal, onaindia, lstarin}@dsic.upv.es
[2] Universidad de Guanajuato (Mexico)
alvarezj@salamanca.ugto.mx

Abstract. In many realistic planning domains, the exact duration of actions is only known at the instant of executing the action. This is the case, for instance, of temporal domains where it is common to find external factors that cause a delay in the execution of actions. In this paper we present an approach to obtain a plan for a temporal domain with delays. Our approach consists in combining a planning process, from which a temporal plan is obtained, and a scheduling process to allocate (instantiate) such a temporal plan over a time line.

1 Introduction

Research in AI planning is more and more concerned with the introduction of more expressive languages and development of new techniques to deal with more realistic problems. A crucial element in this approach to reality is time. In the last years, several extensions in the standard language PDDL have represented a step forward in the resolution of temporal planning problems, as the introduction of durative actions in PDDL2.1[1] or timed initial literals in the most recent PDDL2.2 [2].

Time also plays an important role because it is a source of imprecision and uncertainty.([3], [4]). This also occurs when plans are executed in a specific temporal setting and the execution of actions is affected by the typical delays due to external factors. Actions do not always take the same time to execute but real durations depend on the timing at which actions are actually executed, which may be affected by a longer or shorter delay. A lot of elements condition the real duration of actions: moving from one place to another is more costly at peak hours, loading objects in a container depends on the number of resources (hoists, humans) available when the load is carried out and a space operation might last differently whether it is carried out in daytime or at night.

In this paper we present an approach to obtain a plan for temporal domains with delays. The approach integrates a standard planning process, from which a temporal plan is obtained, with a CSP resolution to instantiate the temporal plan over a real time line. Our work brings two main contributions. First it introduces a model to handle delays in temporal planning domains, a concept that has not been previously dealt with in the literature. Second we present a different way of handling actions with variable durations through a scheduling process rather than within the planning process.

* This work has been partially funded by the Spanish government CICYT project TIN2005-08945-C06-06 (FEDER) and by the Valencian government project GVA06/096.

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 399–408, 2006.

2 Delays in a Temporal Setting

Actions or activities seldom last as initially planned (expected) in daily life. In principle, delays are produced by generic causes that affect action durations equally, no matter the timing at which actions are executed. For instance, driving with a heavy traffic flow implies to move slower, regardless if it happens at 9 a.m. or at 9 p.m..

However, many common causes of delay typically occur at particular times or time intervals. Peak-hours usually occur early in the morning or late in the afternoon after work. A lack of human resources may likely occur at lunch time. Therefore, an action may be affected by different delays along its execution depending on the time intervals over which the action is executed. If action `drive-truck T1 A B` starts at 8 a.m. and it takes two hours under normal conditions, the first driving hour may be affected by a heavier traffic flow than the second hour. Unloading a truck that usually takes one hour may be longer because the action starts at 11.30 a.m. and there are fewer workers from 12.00 a.m. on.

A delay is the extra-time to put in the action duration over a time interval due to an external factor. Let's define $\alpha_{<cause>}$ the delay caused by any reason. The value of $\alpha_{<cause>}$ can be specified as: a) a fixed value (i.e. 5 minutes), b) a value proportional to the standard action duration, (i.e. 5% of the standard duration or equivalently a fraction 5/100 in the interval [0,1]), c) a value given by a formula (i.e. $\mathsf{speed} * \mathsf{weight} - \mathsf{distance}$).

We will denote by $\alpha_{<cause,a_i>}$ the value that results from computing $\alpha_{<cause>}$ for a_i. The value $\alpha_{<cause,a_i>}$ represents the increase fraction that has to be applied on a_i when the $< cause >$ of delay is found during the execution of a_i. For each action a_i and time window $[t_b, t_e]$ (in our case $t_b = 0, t_e = 24$) we define a set of subintervals over $[t_b, t_e]$ and specify a delay associated to each of them. This set of subintervals along with their corresponding delays are called *timing intervals* of an action a_i. When a_i is executed over a timing interval, the actual duration of a_i is augmented with the delays indicated in the timing interval.

There can be more than one cause of delay over a timing interval. We denote by $\alpha_{c1[a_i,t_i,t_j]}$, $\alpha_{c2[a_i,t_i,t_j]}$, ..., $\alpha_{cn[a_i,t_i,t_j]}$ the values of the delays produced by causes $c_1, c_2, \ldots, c_n$ on action a_i over timimg interval $[t_i, t_j]$. That is, the result of $\alpha_{<c_k,a_i>}$ $\forall c_k \in < cause >$ over timing interval $[t_i, t_j]$. For the sake of simplicity, we will denote by $\alpha_{[a_i,t_i,t_j]}$ how long action a_i is totally delayed over $[t_i, t_j]$ due to all the delay causes produced over such a timing interval.

The definition of delays over timing intervals for an action a_i can be interpreted as a continuous function over the time line. This scheme would also allow to specify delays as a probability distribution. Figure 1 shows the timing intervals of the action `drive-truck T1 A B` for $\alpha_{heavy-traffic}$ and α_{night}. Thus, heavy traffic only affects timing interval $[8, 10]$ and the action is delayed $0.4 \times dur(a_i)$; and driving at night produces a retard over the timing interval $[20, 7]$ of $0.2 \times dur(a_i)$.

3 Planning in Temporal Domains with Delays

The straightforward consequence of handling temporal domains with delays is that the same action can have different durations depending on the timing intervals over which

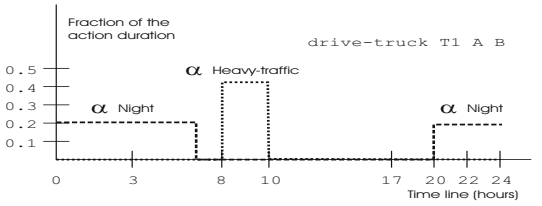


Fig. 1. Graphical representation of delays

it is executed, that is the duration will be different depending on its start execution timing[1]. One way to handle this would be to compute in advance the duration of each single action at each time point. However, although this can be done in polynomial time, the result would be a prohibitive number of different instantiated actions (one instance for each different possible duration) which would cause a blow up in the planning process. This also entails a greater difficult for computing accurate estimations of the plan duration to define heuristics.

Our proposal is not to consider the allocation (instantiation) of a plan in time into the temporal planning process but in a separate post-planning process (see figure 2a). First, we obtain a plan from an existing temporal planner and the start/end time of the plan is initially set to a value on the real time line, according to the user specifications; otherwise, the start time is set to any value. Then we calculate the extended duration of the plan when delays are applied. Once we have the plan allocated in time, we check if the user temporal constraints hold in the plan. If this is the case, a solution is returned. Otherwise, the plan is converted into a CSP and its resolution will provide new allocations of the plan in time until we find a solution that satisfies the user restrictions.

The post-planning process performs two main tasks:

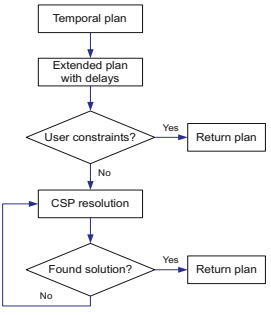
- **Apply delays to a temporal plan.** When the plan is set on a time line and, consequently, the timing intervals over which actions are executed are known, an algorithm to compute the extended duration of the plan is applied. This is explained below in more detail.
- **Scheduling a plan as a CSP.** The objective of converting a plan into a CSP is to find new allocations in time for the plan. Actions are reordered and re-allocated in time so as to minimize the overall delay and satisfy the user restrictions. This is explained in section *Plan scheduling as a CSP*

The extended (real) duration of a parallel plan P composed of several sequences of actions will be the extended duration of the longest sequence. We initially set the start/end time of the plan according to the user specifications. This gives us information about the timing intervals over which actions will be executed and we can then proceed to compute the extended duration of the plan. We show here how to compute the extended duration of a single action; the same process is then repeated for every action

[1] In this first aproximation we only consider the left extreme of timing intervals and we assume the action duration is the same regardless the specific time point within the corresponding timing interval. In a future work, we will extend this approach to work with the exact duration of each action.

and the final extended plan duration is calculated as indicated above. Figure 2b shows the algorithm for applying delays to a single action a_k.

Let a_k be a durative action which start time and standard duration are denoted by beg_k and d_k respectively. a_k starts at timing interval $[t_0, t_1]$ and finishes at timing interval $[t_{n-1}, t_n]$. The basic procedure of the algorithm works as follows: for action a_k and timing interval $[t_i, t_j]$, find out "how much" of the duration of a_k over $[t_i, t_j]$ actually corresponds to the action execution and "how much" corresponds to causes of delays in the timing interval; then substract the action duration from d_k and repeat again for the next timing interval.



(a) - `Post-planning process`

$p = 0$
do
if $(d_k + d_k \times \alpha_{[a_k,t_p,t_{p+1}]} \leq (t_{p+1} - \max(t_p, beg_k))$
then $end_k = \max(t_p, beg_k) + d_k + d_k \times \alpha_{[a_k,t_p,t_{p+1}]}$
else
$\Pi = (t_{p+1} - \max(t_p, beg_k))/(1 + \alpha_{[a_k,t_p,t_{p+1}]})$
$d_k = d_k - \Pi$
endif
$p = p + 1$
while $d_k > 0$

(b) - `Algorithm for delays application`

Fig. 2. Post-planning process to compute a plan in a temporal domain with delays

4 Plan Scheduling as a CSP

This step is executed when the scheduled plan P does not fulfil the user requirements. We build a CSP that encodes: a) information in plan P, b) all possible planning alternatives in P (different ways of achieving a literal with actions in P), c) the different durations actions in P can take on according to their timing intervals and d) user constraints.

It is important to remark that, unlike other approaches ([5], [6]) our CSP does not encode the planning problem but a scheduled plan in time plus all the necessary information to modify the plan and make it accomplish the user requirements.

4.1 CSP Specification

Let P be a partially-ordered set of actions $\{a_o, a_1, \ldots, a_n\}$; a_0 and a_n are the initial and final ficticious actions that represent the start and end time of P, respectively. We will use s_i and e_i to denote the start and end time of an action a_i, and t_i to refer either s_i or e_i indistinctly. Let X be the set of variables $X = \bigcup_{i=0}^{n} s_i \cup \bigcup_{i=0}^{n} e_i$. For the ficticious intial and final actions, $s_0 = e_0$ and $s_n = e_n$. Time is modelled by $\mathcal{R}^+$ and their chronological order. Particularly, the list of possible values for an end time point is $D_{e_i} = [0 \ldots 24) \in \mathcal{R}^+$ and for a start time point is $D_{s_i} = \{[0, v_1][v_1, v_2] \ldots [v_n, 24]\} \in \mathcal{R}^+$

where v_i denote the left/right extreme points of the timing intervals. We distinguish three types of constraints C: planning, timing interval and user constraints.

This specification gives rise to a TCSP [7] as the problem involves a set of variables X having continuous domains ($\mathcal{R}^+$), each variable represent a time point and the three types of constraints are encoded as a set of unary and binary constraints (see below).

Planning Constraints. We show here how to encode planning relationships into temporal constraints. $add(a_i)$ and $del(a_i)$ denote the positive and negative effects, respectively, generated by an action a_i.

- *Causal link*. We denote a causal link [8] between actions a_i and a_j as (a_i, a_j, p) where p is the literal that a_i produces for a_j. The translation of a causal link into a temporal constraint has four variants, depending whether literal p is produced at start or end of a_i and required at the start or end of a_j. These four variants are encoded as binary constraints: $s_i \leq s_j$, $s_i \leq e_j$, $e_i \leq s_j$ or $e_i \leq e_j$ $\forall i \leq j$. We will use the general temporal constraint $t_i \leq t_j$ to refer to any of the four variants. The different planning alternatives (ways of producing literal p with actions in P) are encoded as a disjunctive temporal constraint:

$$\bigvee_{\forall a_k / p \in add(a_k)} t_k \leq t_j \tag{1}$$

- *Threat*. Let's suppose a_k threatens causal link (a_i, a_j, p). The set of disjunctive constraints to model the demotion and promotion choices are:

$$\bigwedge_{\forall a_k / p \in del(a_k)} t_k < t_i \vee t_k > t_j \tag{2}$$

  In the case of an *overall* condition the disjunctive constraint would be $\bigwedge_{\forall a_k / p \in del(a_k)} t_k < t_i \vee t_k > e_j$.

We classify planning constraints into three different categories:

1. *Safe causal link*. When a_i is the only way to produce literal p and there is no action, except may be a_j, that deletes p. This implies a_i is the only producer so the causal link is unmodifiable and there are no threats over (a_i, a_j, p). Therefore, a safe causal link is encoded as temporal constraint that represents a causal link.
2. *Hard causal links*. When a_i is the only way to produce p and it exists at least one action that deletes p. This implies a_i is the only producer so the causal link is unmodifiable and there can be threats over (a_i, a_j, p). Therefore, a hard causal link is encoded as a temporal constraint that represents the causal link plus a set of disjuntive constraints to avoid threats over such a causal link.
3. *Weak causal links*. When there are several producer actions of p for a_j. A weak causal link is encoded by using the disjunctive temporal constraint of the planning alternatives plus a set of disjuntive constraints to avoid threats on each possible causal link. The combination of these contraints is encoded as:

$$\bigvee_{\forall a_i / p \in add(a_i)} t_i \leq t_j \wedge (\forall a_k / p \in del(a_k)(t_k < t_i \vee t_k > t_j)) \tag{3}$$

Timing Intervals Constraints. The constraints represent the different durations an action a_i can take depending on the timing interval where a_i starts its execution. For example, assuming we have $\alpha[a_i, 3, 5]$ and $\alpha[a_i, 5, 9]$, the final duration of a_i will be different if $s_i \in [3, 5]$ or $s_i \in [5, 9]$. The final duration also depends on the specific time point within the corresponding timing interval but we simplify the modelling by just assuming that the extended duration of a_i is the same when s_i falls at any point within the same timing interval[1]. The timing interval constraints are encoded as:

$$\forall a_i \in P \bigvee_{\forall [t_p, t_{p+1}] \in [t_b, t_e]} t_p \leq s_i \leq t_{p+1} \wedge e_i = s_i + d, \tag{4}$$

where d is calculated by applying the algorithm of figure 2b. We restrict the plan to be executed before $t_e = 24$.

User Constraints. Constraints on the start/end time of the plan are expressed as $s_0 \geq v_0$, $e_n \leq v_n$. The same specification is used to represent restrictions on the start/end time of the actions in the plan ($t_i \leq v_i$ $t_i \geq v_i$). A limit in the plan duration is encoded as $e_n = s_0 + d_k$.

4.2 Encoding the Current Plan into a TCSP

When we encode the current plan into a TCSP, we will obtain a representation of this plan together with all the possible alternative plans we can build. The different possibilities to encode weak causal links lead to a huge number of alternatives (see third column in table 1). Let us assume 2 actions that produce the literal p and 3 actions that delete it. This entails 2 (producer) * 3 (delete) * 2 (promotion/demotion) = 12 different alternatives to encode this weak causal link. The first step to encode the plan is to build a partial TCSP with information for safe and hard causal links (the latter with no consideration of actions that threat the link). This information is also registered in terms of actions, preconditions and effects so as to automatically apply a set of heuristics that allows reducing the number of alternatives while completing the TCSP. These heuristics are:

- *H0:* If there exists an action that needs a literal, which is already produced in the partial TCSP, this heuristic will reuse that action, instead of including a new one.
- *H1:* When there exists only one action a_i that deletes and needs the same literal, a_i can not be placed before an action a_j that produces such literal. Otherwise, it would be necessary to include a new action a_k to produce again that literal.
- *H2:* This heuristic creates several groups of actions based on the information of the partial TCSP (two actions belong to the same group if they have an ordering relationship). In the same group, two actions that need the same literal share the same producer action (for that literal) if only one of them deletes it.
- *H3:* If an action that produces a literal is placed after an action that needs it, this heuristic will not consider the first action as a producer.
- *H4:* Let us assume two actions in the same group. If an action a_i produces a literal and a_i is not placed after an action a_j that needs it, this heuristic will use a_i to solve that literal.

– *H5:* If an action a_i that deletes a literal is already placed after an action a_j that needs it, a_i will not be placed before the action that produces the literal for a_j.

These heuristics are applied in the same order as they appear listed above. After applying this set of heuristics we analyse the resulting alternatives and check out those ones that are consistent. This process is incrementally done, starting from the partial TCSP, adding successively restrictions and discarding those combinations that lead to an inconsistent solution.

4.3 Solving the CSP

Next step is to solve this TCSP in order to obtain a new plan, that is finding an alternative that satisfies the user requirements. As shown above, this is a disjunctive TCSP, with unary and binary constraints. A straightforward way of solving it is to decompose it into several non-disjunctive TCSP [9]. This is the approach we have adopted, but we need to distinguish between two levels of decomposition: the first level considers the disjunctive planning constraints (weak and hard) while the second level considers the disjuntive timing interval constraints. Once a non-disjuntive TCSP has been built, we apply an arc-consistency procedure in order to shrink interval domain for each s_i. The remainder of this section formalizes the algorithm to solve the general TCSP.

First, we need to define the intersection of intervals [9]. Let $T = \{I_1, \ldots, I_l\}$ and $S = \{J_1, \ldots, J_m\}$ be two constraints representing the domain of intervals of a temporal variable s_i or e_i. The *intersection* of T and S, denoted by $T \oplus S$, admits only values that are allowed by both T and S, that is, $T \oplus S = \{K_1, \ldots, K_n\}$ where $K_k = I_i \cap J_j$ for some i and j.

Our algorithm works in four stages. At any time, the algorithm can return that there is no solution when any of the variable domains becomes empty. These stages are:

1. **Dealing with the user requirements.** The first step of this algorithm consists in shrinking the given domains of the variables according to the user requirements. We distinguish three types of user requirements and each of them has a different process:
   – *Start and end of the plan:* Given a start and end of the plan constraints of the form $s_0 \geq v_0$ and $e_n \leq v_n$, we can shrink the domains of all the variables in the CSP as we implicitly know that $\forall a_i \in P, s_0 \leq s_i \wedge e_i \leq e_n$. Given that D_{t_i} represents the domain of a variable s_i or e_i, the new domains are computed as follows: $D_{t_i} = D_{t_i} \oplus [v_0, v_n]$. In case s_0 or e_n are not restricted, then it is assumed that $v_0 = -\infty$ and $v_n = \infty$.
   – *Start and end of one action:* Given two constraints indicating the start and end of an action of the form $s_i \geq v_0$ and $e_i \leq v_n$, we can shrink the domain of these variables in the same way as with the start and end of the plan.
   – *Duration of the plan:* This requirement cannot be used to shrink the variable domains, because it does not restrict at what time an action may or may not start. It only shrinks the makespan of the plan which is not represented in the variable domains.
2. **Selection of the planning disjunction.** We focus on those constraints which produce disjunctions: hard and weak causal link constraints. Let $CL = (a_i, a_j, p)$ be a causal link.

- Assuming that CL is a hard causal link, let c_j be a threat constraint related with CL on the form of (2). Therefore, the number of different TCSP we can build considering only CL is $NH_j = 2^{|a_k|}$, where $|a_k|$ is the number of actions that threaten CL.
- Assuming that CL is a weak causal link, let c_j be the constraint corresponding to CL on the form of (3). The number of different TCSP we can build considering only CL is $NW_j = |a_i| \times 2^{|a_k|}$, where $|a_i|$ is the number of actions that can solve p for a_j and $|a_k|$ is the number of actions that threaten CL.

The number of different TCSP grows exponentially as the number of hard and weak causal links increases. Namely, this number is: $\prod_{c_i \in HCL} c_i \times \prod_{c_i \in WCL} c_i$ where HCL and WCL are the set of hard and weak causal link constraints, respectively. We select one of the obtained TCSP randomly[2].

3. **Selection of the timing interval disjunction.** At this moment, we only consider the TCSP selected in the previous step. Let $c_i = \bigvee_{\forall [t_p, t_{p+1}]} (t_p \leq s_i \leq t_{p+1} \wedge e_i = s_i + d)$ be a timing interval constraint. Again, we can build a number of different TCSP, each of them considering a different execution interval for each action. This number is $\Pi_{\forall a_i \in P} |N_{Ia_i}|$, where $|N_{Ia_i}|$ is the number of timing intervals for each action. Fortunately, this number is an upper bound of the number of the TCSP we must actually consider, as some of these combinations rule out due to the domains restriction performed in previous steps. We select a TCSP which will be solved in the next step.
4. **Arc-Consistency.** An arc-consistency process is applied in order to shrink the domain for each variable, that is, we do not calculate the minimal domain because the start/end time points of the actions come conditioned by the start/end time points of the previous actions. The arc-consistency process we have implemented takes $O(n^2)$, where n is the number of variables.

5 Experimental Results

Our first experiment focuses on the process to encode a plan, initially generated by LPG [10], into a TCSP. We run different experiments from the `depots` and `driverlog` domains[3]. All experiments were run on a Pentium IV 3GHz with 512 Mb of memory and censored after 15 minutes. Table 1 shows the experimental results of encoding the plan. The first column represents the problem name, the second column the number of actions in the plan, the third column the initial alternatives, from the fourth to the ninth column the number of alternatives after applying each heuristic, the tenth one the number of consistent alternatives after encoding the plan. Finally, last column shows the CPU time in milliseconds. Note that, in general, H3 discards many alternatives, and after the applications of all the heuristics the number of consistent alternatives is very small. Therefore, we can conclude that encoding the plan in such a way shows very effective and vastly reduces the number of alternatives. Our second experiment

[2] At this moment, there is no reasoning about which of the obtained TCSP may lead to a solution with a higher probability. This will be discussed in the experiments section.

[3] Domains from the IPC 2002: http://planning.cis.strath.ac.uk/competition.

Table 1. Results of encoding a plan using H0 - H5 heuristics

Problem	Actions	IA	H0	H1	H2	H3	H4	H5	CA	Time
Depots_pfile1	12	2400	2400	2400	2400	9	4	4	1	125
Depots_pfile5	60	9.38E140	2.68E136	4.46E135	3.47E132	1.60E127	6.20E115	1.30E105	NO SOL	
Depots_pfile10	31	1.61E25	1.08E25	1.08E25	1.08E25	1.84E20	1.37E18	6.70E15	4	203
Depots_pfile16	34	2.66E23	7.00E21	4.67E21	2.33E21	1.23E12	1.15E10	88473600	10	140
Depots_pfile20	164	inf	inf	inf	inf	inf	inf	inf	NO SOL	
DriverLog_pfile2	26	1.91E13	9.56E12	2.39E12	5.97E11	1.99E10	5.31E09	3.54E09	16	218
DriverLog_pfile5	19	3.9E09	3.9E09	9.74E08	2.44E08	3979530	73728	24576	1	140
DriverLog_pfile10	28	1.63E10	5.44E09	1.21E09	1.91E08	2359296	589824	147456	1	156
DriverLog_pfile15	47	3.38E16	3.38E16	3.38E16	3.38E16	7.51E14	9.66E10	2.42E10	7	186
DriverLog_pfile22	18	2.5E12	5.06E11	6.3E10	8.78E08	2.91E08	1.49E08	53747712	2	203

Table 2. Constraints on the start and end of the plan

Timing intervals	Start-End time	**[10,20]**	**[12,20]**	**[14,19]**	**[14,18:35]**
2	Makespan	245	264	284	No plan (274)
	Start time	10:00	12:00	14:00	(14:00)
	End time	14:05	16:24	18:44	(18:34)
	Time (secs.)	0.17	0.047	0.031	0.031 (0.0625)
Timing intervals	Start-End time	**[10,20]**	**[12,20]**	**[14,19]**	**[14,18:20]**
4	Makespan	301	296	268	No plan (253)
	Start time	10:00	12:00	14:00	(14:00)
	End time	15:01	16:56	18:28	(18:13)
	Time (secs.)	0.843	0.0625	0.0468	0.172 (0.906)
Timing intervals	Start-End time	**[10,20]**	**[12,20]**	**[14,19]**	**[14,18:30]**
6	Makespan	284	286	273	No plan (263)
	Start time	10:00	12:00	14:00	(14:00)
	End time	14:44	16:46	18:33	(18:23)
	Time (secs.)	0.25	0.0624	0.203	2.14 (3.46)

focuses on the search of a plan that fulfills all the temporal constraints in a TCSP. We run different experiments with a hand-made problem (`driverlog_pfile22`), from the `driverlog` domain. The temporal plan for this problem instance contains 18 actions and a makespan of 200 time units (before applying delays). Table 2 shows the results when the user imposes constraints on the start and end time of the plan for three different temporal settings (2, 4 and 6 timing intervals). The first row for each timing interval represents the plan makespan, the second row the start time, the third row the end time of the plan and the last row the CPU time in seconds. For example, the first column indicates that the start and end time of the plan must fall within the interval [10,20]. The values in brackets indicates the results from a second CSP resolution when the first one fails. As we can see when the plan execution interval is more restricted, the number of consistent timing intervals decreases and the search is faster. However, in this case, if it does not exist a valid combination of timing intervals, the necessary time to return *No plan* increases as long as the number of intervals. The CPU time of the experiments depends on when the intervals that provide the shortest duration to actions are selected in the CSP resolution, the set of valid intervals according to the start/end or duration restriction, etc. We want to highlight from the experiments that the CPU time for the CSP resolution is perfectly affordable even though no heuristic information at all has been used to select the most appropriate CSP or timing intervals for a particular solution. Moreover, modeling time constraints at execution time is much simpler in a CSP framework than in a planning system. The main conclusion we can draw from the

experiments is that the number of potential CSPs for a given plan can be vastly reduced when applying appropriate heuristics. Moreover, taking into account the short times to solve a CSP, the number of consistent CSPs is perfectly affordable by our approach. This allows us to infer that scheduling a plan in time through a CSP approach is a promising future direction for solving planning problems with delays.

6 Conclusions and Further Work

In this paper we have presented an extended model of durative actions which takes into account the fact that actions are delayed at the time of being executed in a real domain. This way, we introduce the concept of delay as the increase in the duration of an action due to very common causes in daily life but rarely considered in planning modelling. The introduction of delays create a complete different scenario in temporal planning. Now plans may have a different makespan according to its timing of execution and this aspect has to be taken into account in order to meet the user restrictions on the start/end time or duration of the temporal plan. Using a CSP approach to tackle time restrictions at execution time is a very promising working line. Experiments show that even using a completely uninformed CSP, this approach brings significant benefits at a very low cost, specially if we consider to obtain the same gains from a planning perspective. This makes us keep on considering a separate process for allocating a plan in time. This process could be used not only to handle delays or temporal user requirements but also all kind of restrictions that come out when a plan is to be instantiated in a particular temporal setting.

References

1. Fox, M., Long, D.: PDDL 2.1 : An extension to pddl for expressing temporal planning domains. JAIR **20** (2003) 61–124
2. Edelkamp, S., Hoffmann, J.: Pddl 2.2: the language for the classical part of ipc-04. In: ICAPS-2004 - IPC. (2004) 2–6
3. Biundo, S., Holzer, R., Schattenberg, B.: Project planning under temporal uncertainty. In: ECAI Workshop on Planning and Scheduling: Bridging Theory to Practice. (2004)
4. Bresina, J., Dearden, R., Meuleau, N., Ramakrishnan, S., Smith, D., Washington, R.: Planning under continuous time and resource uncertainty: A challenge for ai. In: Proc. AIPS-02 Workshop on Planning for Temporal Domains. (2002)
5. Do, M.B., Kambhampati, S.: Planning as constraint satisfaction: Solving the planning graph by compiling it into csp. Artificial Intelligence **132(2)** (2001) 151–182
6. Kautz, H.: Satplan04: Planning as satisfiability. In: IPC, 14th ICAPS, abstract booklet of the competing planners. (2004)
7. Ghallab, M., Nau, D., Traverso, P.: Automated planning: Theory and Practice. Morgan Kauffman (2004)
8. Penberthy, J., Weld, D.: UCPOP: A sound, complete, partial order planner for ADL. In: 3rd. Int. Conf. on Principles of Knowledge Representation and Reasoning. (1992) 103–114
9. Detcher, R.: Constraint processing. Elsevier Science (2003)
10. Gerevini, A., Saetti, A., Serina, I.: Planning in pddl2.2 domains with lpg-td. In: IPC, 14th ICAPS, abstract booklet of the competing planners. (2004)

Sliding Mode Control of a Wastewater Treatment Plant with Neural Networks

Miguel A. Jaramillo-Morán[1], Juan C. Peguero-Chamizo[2], Enrique Martínez de Salazar[1], and Montserrat García del Valle[1]

[1] E. de Ingenierías Industriales, University of Extremadura, Avda. de Elvas s/n. 06071 Badajoz, Spain
miguel@unex.es, dsalazar@unex.es, montse@nernet.unex.es
[2] Centro Universitario de Mérida, S. Joaquina de Jornet, s/n. 06800 Mérida, Spain
jcpeguero@unex.es

Abstract. In this work a sliding mode control carried out by neural networks and applied to a wastewater treatment plant is proposed. The controller has two modules: the first one performs the plant control when its dynamics lies inside an optimal working region and is carried out by a neural network trained to reproduce the behavior of the technician who controls an actual plant, while the second one drives the system dynamics towards that region when it works outside it and is carried out by another neural network trained to perform that task. Both controllers are combined with a two layers neural network where the synaptic weights of the only neuron in the second one is adjusted by those in the previous layer in order to balance the contribution of each controller to the total control action.

1 Introduction

Sliding Mode Control [1] is a control theory whose aim is to drive the dynamics of a nonlinear system towards a certain surface and then force that dynamics to remain inside it. To do so the control has two different laws: the first one, usually named corrective control, tries to drive the system evolution towards a surface defined in the state space of the system where it is to be kept, while the second one tries to control the system dynamics inside that surface and is usually named equivalent control.

Nevertheless, this technique has two important drawbacks. The first one deals with the mathematical definition of the surface where the system must converge, which, as it uses to be very complex, makes the corresponding control law that keeps the system dynamics inside it be also very complex. This definition may become impossible when a precise definition of the system or the surface is not provided. The second drawback arises from the fact that the corrective control will act whenever the system dynamics goes outside the optimal surface, a fact that will constantly happen because of the inability of the equivalent control to retain the system dynamics inside the optimal surface. So undesired high frequency

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 409 – 418, 2006.

oscillations will appear in the control signal. This effect is favored by the use of switching functions as corrective controllers.

To solve the first problem both neural networks [2] and fuzzy sets [3] have been used, because of their ability to identify the dynamics of complex systems, specially when their model is unknown.

To solve the second problem, saturating functions that allow a smooth transition between active and inactive states of the corrective control have been used. Neural networks [2] or fuzzy sets [3] also provides an easy solution to this problem.

Generally, the development of a sliding mode control needs a very precise definition of the sliding surface where the systems is to be driven, nevertheless, there are systems whose dynamics may be driven to a region instead of a surface. In many cases this region may have an imprecise definition. This kind of system will be even more difficult to control with algorithmic techniques because of that lack of accuracy in the definition of the sliding surface. Nevertheless these are the kind of systems neural networks or fuzzy sets were designed to deal with. In this work such a system, a Wastewater Treatment Plant (WTP), will be controlled with neural networks. Two networks will be defined to carry out the equivalent and corrective controls: the first one will be trained to learn the behavior of the technician that controls an actual plant, while the second will be trained to force the system dynamics to work inside an optimal region. Then they will be combined by a very simple neural structure defined to decide which controller will act at every moment depending on the distance from the system to the optimal region.

The rest of the work is organized as follows. A description of the plant that will be used is presented in Section 2. The neural networks which will carry out the sliding mode control are developed in Section 3. Finally, in Section 4, the results obtained in simulation are presented.

2 Plant Description

WTP are systems where urban and industrial wastewaters are processed to eliminate as much pollution as possible. This process is carried out in two stages. The first one is performed in an aeration tank where the incoming polluted water is mixed with a sludge made up of bacteria. After they have “eaten” most of the organic matter, water and sludge are driven to a settling tank where the second process is performed: the separation of these two components. The sludge flows downwards while the water stays at the top of the tank and flows over an overflow weir to be released to nearby rivers. The sludge is withdrawn from the tank and then divided into two streams: one is driven to the aeration tank to keep the sludge concentration at an appropriate level while the other is eliminated.

The model used in this work is restricted to the aeration tank, as the main processes of the treatment are carried out in it. A diagram of this element is shown in Fig. 1. Parameters defining a real plant, that of the city of Mérida [4], also appear.

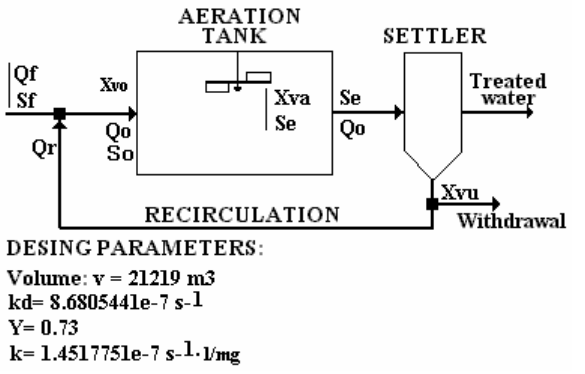


Fig. 1. Diagram of a Wastewater Treatment Plant and values of the parameters of the actual plant used for simulation

In this model the input variables are the "Influent Flowrate" *Qf* and the "Influent Pollution Concentration" *Sf*. The output variables are the "Sludge Concentration" in the aeration tank *Xva* and the "Output Pollution Concentration" *Se*. The "Sludge Concentration" in the settler, *Xvu*, is assumed to be a process parameter that will be provided at every time step. It defines the sludge concentration in the settler, a portion of which will be recirculated to the aeration tank to keep the sludge concentration at an appropriate level. This recirculation is defined by a flowrate, *Qr*, which is adjusted by the technician supervising the plan dynamics. It is represented by a parameter, *r*, which provides the ratio between this "Recirculation Flowrate" and the "Influent Flowrate". The control of this parameter is the aim of this work.

The system dynamics may be defined by a set of differential equations of the variables *Se* and *Xva*. No dependence on the temperature or the aeration will be considered. Their values will be assumed to be optimal, a supposition that is quite close to reality because the great volume of the tanks in the plant ensure a quite constant value for temperature, while the aeration may be set to an adequate level by air pumping. So the plant model may be described by the following equations [5], [6]:

$$\frac{dSe}{dt} = -\frac{Qf}{V} Se - k\ Xva\ Se + Sf \frac{Qf}{V} \ , \tag{1}$$

$$\frac{dXva}{dt} = -\left[\frac{Qf}{V}(1+r) + kd\right] Xva - \frac{Qf}{V}\ Y\ Se + \frac{Qf}{V} r\ Xvu + \frac{Qf}{V}\ Y\ Sf \ , \tag{2}$$

where the recirculation is defined by:

$$r = \frac{Qo - Qf}{Qf} = \frac{Qr}{Qf} \cong \frac{Xva}{Xvu - Xva} \ . \tag{3}$$

In order to ensure an optimal behavior of the plant operation, a new parameter must be defined, the relation between "food" and "microorganisms":

$$F/M \cong \frac{Qf \;\; Sf}{Xva \;\; V} \quad . \tag{4}$$

In the model considered here the system will be working inside their optimal region when this parameter has a value near to 0.133. It is usually assumed that a value between 0.098 and 0.168 may be considered as optimum [5], [6].

3 Plant Control

The plant control will be carried out by the combined action of an equivalent control which works when the system has a value of *F/M* inside the optimal interval [0.098,0.168] and a corrective control which drives the system dynamics towards that region when *F/M* has a value outside it. The first one will be carried out with a neural network that will be previously trained to reproduce the actions of the technician controlling the actual plant whose model has been used in this work [4]. The second controller will be another neural network trained to drive the system to the region where *F/M* is inside the interval [0.098,0.168] if a value outside it is obtained.

As the working regions of both controllers are different it will be necessary to define a new element to decide which one is to be active for a certain value of *F/M*. It will be designed so that when a controller is active the other is inactive. Moreover, in order to avoid the system suffering from high frequency oscillation in the control signal, the transition between the two states of the controllers must be smooth. Then when one of them is switching on the other is switching off, so that there is a time interval where both controllers are working together. To carry out their combined action each one will be affected by an activation function that will "filter" its output depending on the value the parameter *F/M* has in a certain moment. So the whole controller that adjust the plant recirculation will have the following form:

$$r(t) = ro + F1(F/M)\;\delta rF/M(t) + F2(F/M)\;\delta r(t) \; . \tag{5}$$

In this expression a constant term *ro* has been included in order to define the control action as a fluctuation around a constant value of the recirculation, as this is the way technicians of actual plants use to work. This constant has been assumed to be the mean value of *r(t)* in the actual plant used in this work throughout the whole period of available data: *ro*=1.74.

3.1 Equivalent Control

The neural model used to learn the technician's behavior is a Multilayer Perceptron [7]. The network inputs are the product of *Qf* and *Sf* (that is to say, the total pollution entering the aeration tank), measured in the present time step, and the system variables *Xva* and *Se*, the inner variable *Xvu* and the control signal δr, measured a time step before. The network output is the correction δr.

The network was trained with the daily data of a whole year (1998) of the actual plant [4]. The best results were provided by a network with two hidden layers with 15

and 10 neurons. Their activation functions were the logistic ones, while that of the neuron in the output layer was the arctangent, whose output ranges between –1 and +1 to provide the desired fluctuation around *ro*.

3.2 Corrective Control

This control will be carried out by a neural network trained to drive the system to a region where *F/M* has a value inside its optimal interval. So, the way this evolution is performed must be defined first. This process begins when the recirculation correction δr cannot force the system to generate a value of *F/M* inside its optimal interval. A correction is added to the recirculation and the system is simulated again. This procedure is repeated until a value of *F/M* inside its optimal interval is obtained. The whole process may be defined as:

$$r = ro + \delta r + \sum_i \gamma_i(x) \ , \tag{6}$$

where the subscript *i* represents the *i*th iteration of the process, and $\gamma_i(x)$ the correction added to the recirculation in the *i*th iteration, whose value is:

$$\gamma(x) = \frac{1}{1+e^{-50(x-0.0182)}} + \frac{1}{1+e^{-50(x-0.184)}} - 1 \ . \tag{7}$$

In this equation *x* represents the value of *F/M* in the *i-1*th iteration. Their parameters have been fixed by a trial and error procedure to obtain a proper adjustment of *r* in an as low as possible number of iterations.

The daily corrections to *ro* needed to obtain an appropriate value of *F/M* throughout a whole year (1998) were obtained in this way. A Multilayer Perceptron was trained with all these data. The best results were obtained with two hidden layers with 6 and 10 neurons. Their activation functions are again the logistic ones, and that of the output layer neuron the arctangent, although their saturation values are –2 and +2 in order to appropriately reproduce the corrections obtained with (6). The network inputs are the product of *Qf* and *Sf* in the present time step along with *Xvu* and *F/M* in the previous time step. The network output is the correction $\delta r F/M$.

3.3 Combination of the Two Controllers

As it has been previously stated the combination of the control signals must be done so that when one is switched on the other is not, although while the transition between their two states both controllers may be active with different intensities. This behavior may be implemented with two functions with the form described in Fig. 2.

To obtain those functions the sum of two logistic functions may be used. So, *F1(x)* may be obtained from:

$$y1(x) + y2(x) = \frac{1}{1+e^{P(x-T1)}} + \frac{1}{1+e^{-P(x-T2)}} \ . \tag{8}$$

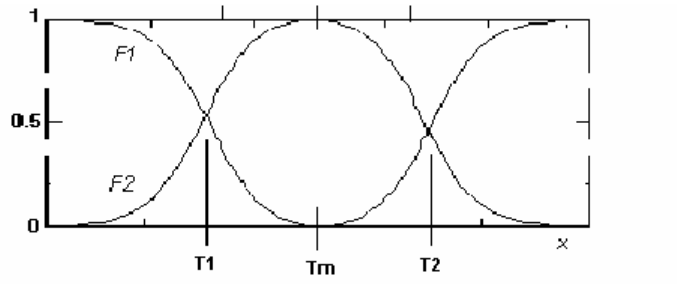


Fig. 2. Plots of functions *F1(x)* and *F2(x)* used to combine the signals of the two controllers that adjust the value of the recirculation

This sum does not represent *F1(x)* because its minimum is not zero. To obtain an expression that fits with the form of *F1(x)* represented in Fig. 2., and assuming that it is symmetrical with respect to its minimum (there is no reason to suppose that control signals over the optimal value of *F/M* must be different to those below it), *T*1 and *T*2 may be defined as $T1 = Tm - \alpha$ and $T2 = Tm + \alpha$, where *Tm* is the point where (8) takes its minimum. Now the values of *y1(x)* and *y2(x)* in *Tm* will be:

$$\sigma = y1(T_m) = y2(T_m) = \frac{1}{1+e^{P\alpha}}. \tag{9}$$

F1(x) will be defined, after some simple manipulations, as:

$$F1(x) = \left[y1(x) + y2(x) - 2\sigma\right]\frac{1}{1-2\sigma}. \tag{10}$$

F2(x) may be easily obtained from *F1(x)* as *F2(x)=1-F1(x)*, then:

$$F2(x) = -\frac{1}{1-2\sigma}\left[y1(x) + y2(x) - 1\right]. \tag{11}$$

Now these two functions may adjust the contribution of the equivalent and corrective controllers to the recirculation *r* as expressed by equation (5) following the aforementioned behavior.

As *F1(F/M)* and *F2(F/M)* are obtained from logistic functions, a two layers neural structure may be used to carry out the control law expressed by (5) if *y1(F/M)* and *y2(F/M)* are assumed to be the activation functions of the neurons in the first layer, and an only neuron with linear output in the second layer provides equation (5), as it is shown in Fig. 3.

In this structure it has been assumed that the outputs of the first layer neurons (*y1(x)* and *y2(x)*) are the inputs to the only neuron in the second, while the corresponding synaptic weights are the corrective and equivalent control signals multiplied by a constant. Nevertheless this assumption means that the signals to be processed, δr and $\delta rF/M$, behave as synaptic weights, what may be considered as a distortion of the meaning of these parameters, as they are defined as constants that modulate the input

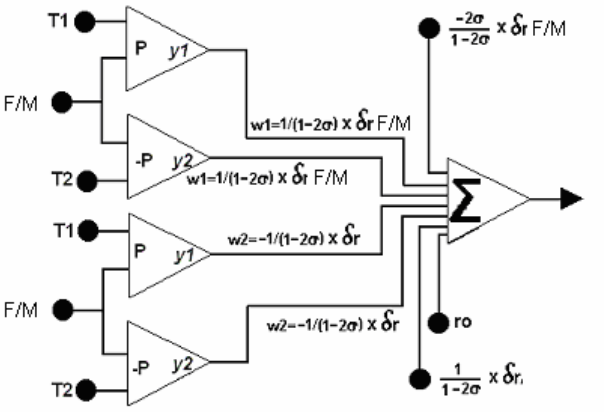


Fig. 3. Neural structure of the recirculation control carried out by (5)

signals of a neuron. To avoid this fact these junctions may be redefined so that the signals δr and $\delta rF/M$ are really the inputs to the last neuron, while the terms $y1(x)$ and $y2(x)$ multiplied by the constant $(1-2\sigma)^{-1}$ represent the synaptic weights. (An even more simplified representation may be obtained if it is assumed that this constant represents the upper saturation limit of the logistic functions, so that $y1(x)$ and $y2(x)$ so defined are the synaptic weights).

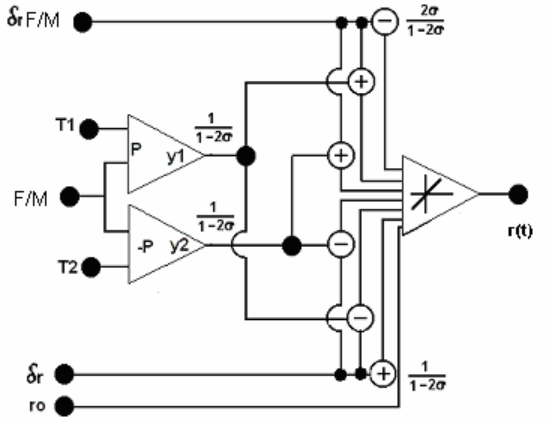


Fig. 4. Simplified neural structure of the recirculation control carried out by (5)

With this new definition it may be assumed that the synapses of the output neuron are controlled by the neurons in the previous layer. But, although this synapses modulate the neural inputs, their values are not fixed in a training process, the usual way to define the synaptic weights. Nevertheless there are neural models which consider that their values may be adjusted in a different way. The well-known CNN (Cellular Neural Network) [8] is a neural network whose synaptic weights are fixed and the same

for every neuron, whose values are selected by the user depending on the task the network will be devoted to, and only spread to the neighboring neurons. So, the assumption that synaptic weights are variable parameters whose values are provided by the neurons in a previous layer may be considered as a generalization of their definition that takes into account the fact that in some neural models [9] thresholds and slopes ($T1$, $T2$ and P in the neurons used here) of the neurons in a layer are adjusted by those in a previous one. So the structure in Fig. 3 may be simplified following these ideas to obtain that in Fig. 4.

4 Simulation Results

The plant model along with the controller defined by (5) have been simulated and the results so obtained have been compared with those provided by the operation reports of the actual plant used as reference [4]. As it has been previously mentioned, the neural network training was carried out with the daily data of a whole year (1998). Those corresponding to the following year (1999) have been used to test the control performance. The values obtained for the system variables *Xva* and *Se* in December 1999 are shown in Fig. 5. Two simulations have been carried out: in the first one the corrective controller was the iterative model described by (6) (Model B in the figure), while in the second the neural network that learned that behavior was used (Model C in the figure). As it is shown in the figure the values obtained with both simulations are almost identical and very similar to the actual data (Reference in the figure). This fact proves the great capability of the neural structure proposed in this work to carry out the same tasks the technicians controlling WTP do. It is worth noting that the plant model used here is a simplified version of a more complex one [5], [6]. The results obtained in this works proves that this simplified model represents a good approximation to an actual system.

In Fig 6. (upper graphics) the values of *F/M* obtained form the actual plant along with the initial values (i. e. the value of *F/M* obtained before the corrective control has actuated) of the simulations with the iterative algorithm (6) and the neural network are

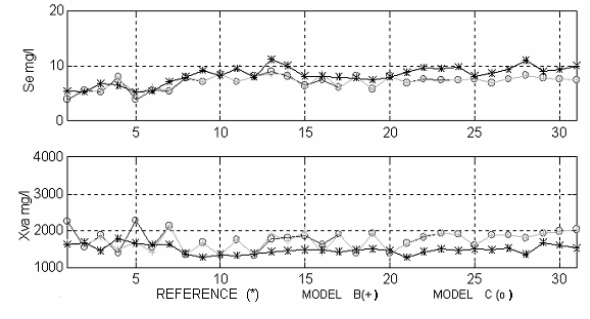


Fig. 5. Simulation results. Values of *Se* and *Xva.* December 1999.

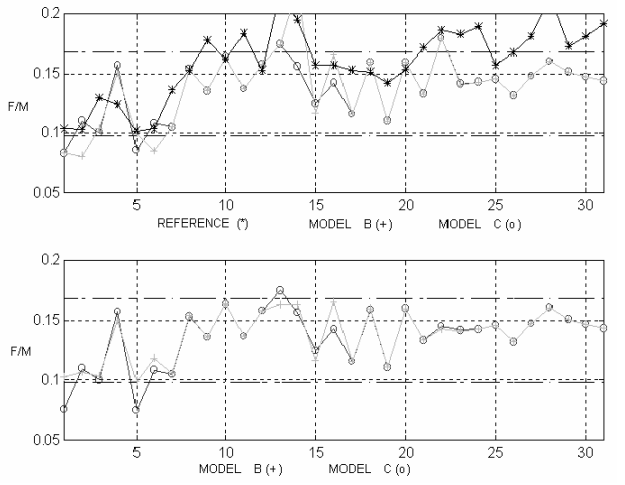


Fig. 6. Simulation results. Values of *F/M*. December 1999.

shown. Several data of the actual *F/M* are outside the optimal region, a fact that is hardly surprising as the technician cannot measure this parameter online, because a long time is needed to obtain the value of *Xva* that allows the calculation of *F/M* (Equation 4). So this lack of information prevents him from carrying out any action in order to drive the system towards their optimal region.

The initial values of *F/M* obtained with the neural network and the iterative algorithm may differ in some cases as they are obtained from the resolution of equations (1) and (2) with different initial conditions: the values of the system variables provided by the two simulation of the whole system the day before.

In the lower graphics of Fig.6. the final values of *F/M* obtained with the neural network and the iterative algorithm (6) are presented. It may be seen how both controllers force the system to have values of this parameter inside the optimal region in most of the cases (always when the iterative process described by (6) is used, a logical result as the algorithm does not stop until the value of *F/M* is inside that interval).

The fact that some values of *F/M* obtained with the neural network falls outside the optimal region must not be assumed as a lack of performance of the controller with the neural network, as the definition of the optimal region is quite subjective (the thresholds $T1$ and $T2$ may be adjusted to modify the interval width). Driving the system dynamics as close as possible to the optimal value F/M=0.133 may be assumed as a realistic aim for the evaluation of the controller performance. On the other hand it is to be noted that the controller with the neural network provides a correction to the recirculation in an only time step, whereas the model with the iterative algorithm (6) needs several time steps to provide a system dynamics with an optimal value of *F/M*.

Finally it is to be pointed out that the neural network which combines the equivalent and the corrective controls may deal with more than two control signals

and in a more flexible form (definition of thresholds and slopes with no restriction) than that presented here, what will enable it to be used in the control of more complex systems. The capabilities of this neural structure will be explored in future works.

References

1. DeCarlo, R. A., Zak, S. H., Drakunov, S. V.: Variable Structure, Sliding-Mode Controller design. In: Levine, S. W. (ed.): The Control Handbook. CRC Press-IEEE Press. (1996) 941-951
2. Tsai, C. H., Chung, H. Y., Yu, F. M.: Neuro-Sliding Mode Control with its Applications to Seesaw Systems. Trans on Neural Networks. Vol. 15, Num. 1 (2004) 124-134
3. Wang, J., Rang, A. B., Chang, P. T.: Indirect Adaptive Fuzzy Sliding Mode Control: Part I: Fuzzy Switching. Fuzzy Sets and Systems. 122 (2001) 21-30
4. E.D.A.R. Aguas de Mérida S. L.: Exploitation and maintenance report (1998 and 1999)
5. Ramalho R. S.: Introduction to Wastewater Treatment Processes. Academic Press (1983)
6. Olsson, G., Newell, B.: Wastewater Treatment System. IWA (1999)
7. Norgaard, M.: Neural Networks for Modeling and Control of Dynamic Systems. Springer-Verlag, Berlin Heidelberg New York (2000)
8. Chua, L. O., Yang. L.: Cellular Neural Networks: Theory. IEEE Trans. on Circuits and Systems. Vol. 35, No. 10 (1988) 1257-1272
9. Jaramillo-Morán, M. A., Fernández-Muñoz, J. A.: Adaptive Adjustment of the CNN Output Function to Obtain Contrast Enhancement. Lectures Notes in Computer Science, Vol. 1607. Springer-Verlag, Berlin Heidelberg New York (1999) 412-421

Techniques for Recognizing Textual Entailment and Semantic Equivalence

Jesús Herrera, Anselmo Peñas, and Felisa Verdejo

Departamento de Lenguajes y Sistemas Informáticos
Universidad Nacional de Educación a Distancia
Madrid, Spain
{jesus.herrera, anselmo, felisa}@lsi.uned.es

Abstract. After defining what is understood by textual entailment and semantic equivalence, the present state and the desirable future of the systems aimed at recognizing them is shown. A compilation of the currently implemented techniques in the main Recognizing Textual Entailment and Semantic Equivalence systems is given.

1 Introduction

The concept "textual entailment" is used to indicate the state in which the semantics of a natural language written text can be inferred from the semantics of another one. More specifically, if the truth of an enunciation entails the truth of another enunciation. For example, given the texts:

1. The three-day G8 meeting will take place in Scotland.
2. The Group of Eight summit will last three days.

It is clear that the semantics of the second one can be inferred from the semantics of the first one; then, it is said that textual entailment exists between both texts. Textual entailment is a directional relationship: in the example above, the first statement entails the second one, but this entailment is not given in the opposite direction. The recognition of textual entailment requires a processing at the lexical level (for example, synonymy between *meeting* and *summit* or between *G8* and *Group of Eight*), as well as at the syntactic level and the sentence semantic level. Entailment between natural language texts has been studied in the last years, either as a part of more complex systems or as an independent application. The long-term interest for Recognizing Textual Entailment (RTE) systems is to give service to a wide range of applications which need to determine entailments between pieces of text written in natural language.

When the entailment relation is verified in both directions, then there is a semantic equivalence between the pair of statements, sometimes named paraphrase. Lin and Pantel [17] show a classification of the fields in which the recognition of semantic equivalence is useful, identifying the following:

– Language generation: where efforts in the detection of semantic equivalence have been focused mainly on rule-based text transformations, in order to satisfy external restrictions such as length and readability.

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 419–428, 2006.

- Automatic summarization: in which the detection of paraphrasing is relevant for avoiding redundancy between statements in a summary.
- Information Retrieval: in which is common to identify phrasal terms from queries and to generate their variants for query expansion.
- Text mining: in which a goal is to find semantic association rules between terms.

Other applications, such as answer validation in question answering tasks or translation comparison in machine translation, can be added.

Classically, the detection of entailment between texts has been tackled by means of some kind of calculus applied to abstract representations of texts. The formalism used to represent texts depends on the kind of treatment given to them. When applying a surface treatment to texts, they can be represented by means of formalisms such as syntactic trees. But when a deep treatment is accomplished, more complex formalisms are needed in which different normalization levels can be given; for example, a normalization level between active voice and passive voice, or a deeper normalization level, as the one proposed by Schwank, using primitives. Thus, the richness of every inference level varies with the kind of normalization level. The kind of calculus necessary to determine when a pair of texts hold entailment depends on the representation formalism selected. Therefore, when using representations corresponding to a surface treatment of the texts, usually similarity metrics between representations are computed; but when using a deep treatment, logic calculus, theorem provers, etcetera are the more suitable techniques in order to detect entailment. Apart from these classic techniques of Natural Language Processing, the advent of mass access to textual information in digital format has meant the success for empirical methods, such as statistical analysis and machine learning. These methods are usually applied in a quite superficial level of knowledge representation.

Despite the number of systems aimed at determining the existence of equivalence and entailment relations between pieces of text written in natural language, there is not a systematization of techniques and tools for the development of such kind of systems. In the following sections, a compilation of the currently implemented techniques in the main Recognizing Textual Entailment and Semantic Equivalence systems is given.

2 Linguistic Techniques

One way or another, all the techniques for linguistic processing are liable of being included in a RTE or a Semantic Equivalence based system. Following, the used ones for developing this kinds of systems are shown:

2.1 Preprocessing

Apart from the necessary token identification, there are systems that develop a preprocessing of the texts before applying them a morphosyntactic analysis, which is stated at the bottom of linguistic processing levels. This processing

consist, in most cases, in the segmentation of sentences and phrases, which has been used as a preparation for the morphological analysis or for the creation of structures for representing texts.

The MITRE[1] is an example for it. They apply to the texts and the hypothesises a sentence segmenter, previously to the morphological analysis.

The system of the Concordia University [2] does not accomplish a morphological analysis but a noun phrase chunking as a basis for creating predicate structures with arguments for every text and hypothesis. A similarity metric between the structures of every pair of text snippets is established in order to determine if there is an entailment between them.

2.2 Morphological and Lexical Analysis

From this kind of analysis, they can be distinguished the following cases: lemma or stem extraction, part-of-speech tagging, use of morphological analyzers and extraction of relations forced by derivational morphology.

The morphological analysis has been used as a first text processing in order to obtain information for subsequent stages which permit to assess the entailment between texts.

The **lemma extraction** is a fairly profusely used technique and, in some cases, it supposes a great part of the total processing accomplished by RTE systems. Lemmatization is necessary not only for accessing lexical resources as dictionaries, lexicons or *wordnets* but it has been used with three different goals: to assess the coincidence between lemmas in similarity measures when treating texts as bags of words, as attributes of graph representations of the texts, and to fit parameters in assessing similarity algorithms. Therefore, as an example, the universities of Edinburgh and Leeds' system [5] uses lemmatization as the most sophisticated language processing; after it, only an overlap measure between lemmas from the hypothesis and the text is applied to determine the existence of an entailment between them. The University of Illinois at Urbana-Champaign' system [8] uses lemmas as a part of the attributes associated to the nodes of concept trees which represent both the texts and the hypothesises. The University of Rome "Tor Vergata" and the University of Milano-Bicocca [19] developed a system in which a morphological analysis is applied for lemma extraction; these lemmas are used in combination with tokens and other items for fitting – by means of a SVM[2] learning algorithm – the parameters of an overall similarity measure between the two graphs representing the text and the hypothesis.

The **stem extraction** has been a technique basically used to obtain data as an input for other system's modules. The use of stems in monolingual English is justified because the good performance shown, motivated by the simplicity of the English morphology; in the future, when RTE systems will be developed for other languages, it will be necessary to assess the possibility of working only with stems or, on the contrary, it will be compulsory to use lemmas. As an example,

[1] The MITRE Corporation, United States.
[2] Support Vector Machine.

the system of the universities "Tor Vergata" and Milano-Bicocca [19] compares stems in order to measure the subsumption of nodes of the graphs they use to represent textual information. This measure, in conjunction with other measure for the subsumption of edges, determine the overall subsumption between graphs representing the text and the hypothesis; the overall subsumption measure is useful to detect the entailment between the text and the hypothesis.

The **part of speech tagging** has been used in two different ways: the system of the MITRE [4] and the one of the University Ca' Foscari and the ITC-irst[3] [9] include it as a linguistic analysis module in a typical cascaded system; but the University of Illinois at Urbana-Champaign [8] uses parts of speech as a subset of the attributes associated to the nodes of conceptual trees representing both the text and the hypothesis.

The **use of morphological analyzers** as such was accomplished only by the MITRE [4], applying a morphological analyzer (Minnon et al., 2001) which action was added to the part of speech tagging, and the results were used as an input for the following stages (a syntactic analyzer of syntactic constituents, a dependency analyzer and a logic proposition generator).

The **extraction of relations given by derivational morphology** is a not frequently used technique; an example can be found in the system of the Language Computer Corporation [10], which extracts relations between words from *WordNet* derivational morphology.

2.3 Multiword Recognition

Is a not widely used technique. For example, the system of the UNED[4] [13] uses it to detect entailment between lexical units; for this, a fuzzy search of the multiwords of the texts in *WordNet* is accomplished, by means of the Levenshtein distance. It permits to establish semantic relations (synonymy, hyponymy, etcetera) not only between words but between multiwords and words.

2.4 Numerical Expressions, Temporal Expressions and Entity Recognition

There are not very used techniques yet. In the case of entity recognition, two examples can be found only: the Stanford University [20] and the University of Illinois at Urbana-Champaign [8]. Stanford's system detects named entities and resolves correferences, aiming at finding dependencies between nodes of the graphs representing the texts. The one of the University of Illinois, uses named entities as attributes of the nodes of the graphs representing the texts. As for the detection of numeric and temporal expressions, two other examples can be found: the Stanford University [20] accomplishes a treatment of numeric expressions, being able to determine inferences like *"2113 is more than 2000"*. The

[3] ITC-irst, Centro per la Ricerca Scientifica e Tecnologica, Scientific and Technological Research Center, Italy.

[4] Universidad Nacional de Educación a Distancia, Spanish Distance Learning University.

University Ca' Foscari and the ITC-irst [9] detect temporal expressions, in order to accomplish coherence checks.

2.5 Syntactic Analysis

The **dependency analysis** is one of the most used techniques; probably, this situation has been favored by the public availability of dependency analyzers for the English language showing a high efficiency and a high recall, such as the one developed by Dekang Lin [16] (Minipar).

Using Minipar, Dekang Lin and Patrick Pantel [17] proposed a non-supervised method for the extraction of inference rules from text (DIRT algorithm); some examples of these rules are the following: *"X is author of Y" = "X wrote Y"* , *"X solved Y" = "X found a solution to Y"*, *"X caused Y" = "Y is triggered by X"*. Their algorithm is based on an extended version of the Harris' Distributional Hypothesis [12], which states that words that occurred in the same contexts tend to be similar; instead of it, they applied the hypothesis not to words but paths from dependency trees obtained from a corpus of texts. Lin and Pantel's work aimed at simplifying the creation of knowledge bases for this kind of rules, which usually is done manually and it is very laborious.

In most cases, a parsing tree representing the analyzed text is obtained; but it is used as an auxiliary to obtain a logic representation, too. Examples for the former kind of use are UNED's system [13] and the team of the University of Trento and the ITC-irst's system [15]. The first one assesses the existence of entailment between text and hypothesis by means of the overlap between the dependency trees of both text snippets. The second one assesses the existence of entailment between text and hypothesis by means of the editing distance between the dependency trees of both text snippets; it is based on the previous work of Hristo Tanev, Milen Kouylekov and Bernardo Magnini [21], who developed a textual entailment recognizing system in order to use it as a subsystem of a question answering system. As an example for the other kind of use, the MITRE [4] implements a set of cascaded linguistic analysis subsystems, which includes a stage for dependency analysis; before the dependency analyzer there is a constituent syntactic analyzer, and a logic predicate generator after it.

The **constituent analysis**, on the other hand, is a not very used technique. The University "Tor Vergata" and the University of Milano-Bicocca [19] use constituents in order to extend dependency graphs. The University Ca' Foscari, with the ITC-irst [9], accomplish a constituent analysis as a part of a hybrid syntactic analysis.

2.6 Semantic Analysis

The **semantic roles** tagging was used by the universities of Illinois at Urbana-Champaign [8], Stanford [20] and Ca' Foscari in association with the ITC-irst [9]. The system of Illinois at Urbana-Champaign searches for coincidences between sets of attributes and the structure of the arguments, either at the semantic role level either at the syntactic analysis level. For the case of Stanford, this tagging

permits to add relations between words not previously identified by means of the syntactic analysis; in addition, it permits to classify temporal and locative sentences. In all these cases, the tags were applied to the nodes of the graphs representing text snippets. The University Ca' Foscari and the ITC-irst used semantic roles in a similarity measure between the text and the hypothesis, by means of the count of similar tags between the ones of the text and the ones of the hypothesis.

Some systems represent texts in a logic form after a linguistic analysis, such as the one of the University Macquarie [1]. This one uses an **automated deduction** system that compares the atomic propositions obtained from the text and the hypothesis in order to determine the existence of entailment.

3 Other Techniques

Apart from the techniques showed before, a significant part of the systems implement one or more of the following:

3.1 Using Thesauri, Big Corpora and *WordNet*

An important part of the systems obtains knowledge from thesauri, big corpora and *WordNet*. The queries to *WordNet* have been launched either searching for the acquisition of relations between lexical units from the relations of *WordNet* – such as UNED's system, which searches for synonymy, hyperonymy and *WordNet* entailment relations in order to detect entailments between lexical units from the text and the hypothesis –, either for the obtention of relations from lexical chains, such as University of Concordia's system [2]. Thesauri have been used in order to extract knowledge of concrete fields such as geographical knowledge, obtained by the universities of Edinburgh and Leeds [5] from the "CIA factbook". Big corpora such as the web or the Gigaword newswire corpus have been used in order to acquire lexical properties [4] or co-occurrence statistics [11].

3.2 Paraphrase Detection

The use of paraphrases aims at the obtention of rewriting rules, in order to improve performance when determining if two expressions are equivalent or not. As an example, the University of Illinois at Urbana-Champaign, from a paraphrasing rules corpus developed by Lin y Pantel (2001), obtained a set of rewriting rules, which were used by their system in order to generate variants of the texts [8].

3.3 Machine Learning

Some systems used this kind of algorithms such as, for example, the one of the universities 'Tor Vergata" and Milano-Bicocca [19], which applied a SVM algorithm in order to assess the parameters of an evaluation measure.

3.4 Definition of a Probabilistic Frame

The only existing example is the University Bar Ilan's one, which defines a probabilistic frame in order to modelize the notion of textual entailment [11]; in addition, it uses a bag of words representation in order to describe a lexical entailment model from co-occurrence statistics obtained from the web. It is said that a text probabilistically entails a hypothesis if the text increases the likelihood of the hypothesis being true. In order to treat lexical entailment, a probabilistic model is established for which a word of the hypothesis must be entailed by other word of the text, in a similar way as done in statistical machine translation [6]. Therefore, the probabilistic entailment between text and hypothesis is computed according to the referred lexical entailment. The probabilities of lexical entailment are empirically estimated by means of a non-supervised process based on web co-occurrences.

3.5 Machine Translation

The MITRE developed a system inspired in statistical machine translation models [4] that: trains a machine translation system by means of leads and headlines from a newswire corpus; estimates manually the reliability of the previous training; trains a text classifier for refining the previously obtained corpus; inducts aligning models from the selected subset of the newswire corpus; combines all the features using a k-nearest-neighbour classifier that chose, for every pair <text, hypothesis>, the dominant truth value among the five nearest neighbours in the development set.

4 Evaluation and Corpora

The First PASCAL[5] RTE Challenge [7], aimed at providing an opportunity to present and compare diverse approaches for modeling and for recognizing textual entailment. The task that systems had to tackle was the automatic detection of semantic entailment between pairs of texts written in natural language (monolingual English). For this purpose, the organizers provided to the participants two corpora, one for training and one for testing. The corpora were conformed by pairs of short texts in natural language pertaining to the press news domain. The components of a pair were named as "text" and "hypothesis", respectively. The systems had to detect if the meaning of the hypothesis could be inferred from the meaning of the text. The pairs <text, hypothesis> conforming the corpora provided to the participants of the PASCAL RTE Challenge were chosen so that typical features of diverse text processing applications were present; therefore, the following classification was obtained: Information Retrieval, Comparable Documents, Reading Comprehension, Question Answering, Information Extraction, Machine Translation and Paraphrase Acquisition. The Second PASCAL RTE Challenge [3] toke place while this paper was been revised. It was

[5] Pattern Analysis, Statistical Modeling and Computational Learning. `http://www.pascal-network.org/`.

very similar to the First Challenge, but the tasks considered this time for the classification of the pairs were: Information Extraction, Information Retrieval, Multi-Document Summarization and Question Answering.

Between the two PASCAL RTE Challenges, some other actions related to RTE and semantic equivalence have been performed. In the ACL[6] Workshop on Empirical Modeling of Semantic Equivalence and Entailment, several items about how to analyse and to develop the kinds of systems of interest,s and how to build corpora for training and testing them were treated, following the ideas given in the First PASCAL RTE Challenge. Related to RTE in Spanish, two initiatives were accomplished by the UNED NLP Group[7]: *a)* the development of the SPARTE test suite for Spanish[18], which is based on the answers given by several systems in Question Answering (QA) exercises from the CLEF[8], and *b)* the organization of an Answer Validation Exercise[9] in order to apply RTE systems for emulating human assessment of QA responses and decide whether an answer is correct or not according to a given text snippet.

5 Conclusions

Broadly speaking, just after the First PASCAL RTE Challenge some tendencies in the development of RTE systems can be distinguished: *a)* Those treating texts as bags of words, being lemma extraction the deeper linguistic analysis accomplished. *b)* Those based on a syntactic representation of texts, including some morphological and lexical processings in order to increase system's performance; in this case, overlap between dependency trees is the preferred technique. *c)* Those accomplishing a deep linguistic treatment, by means of a classical cascaded analysis, covering a wide range of levels: morphological, lexical, syntactic and semantic.

There are little examples of systems implementing only statistical treatments or systems accomplishing a deep linguistic analysis.

The results obtained in the First PASCAL RTE Challenge are not significant about the suitability of the used techniques, because all the participants achieved very similar values of accuracy, ranging between 49.5 % and 58.6 % [7]. But the results of the Second Challenge permit to glimpse what are the more suitable techniques to tackle the Recognizing Textual Entailment problem: while most of the systems ranged between 50.9 % and 62.6 % accuracy – showing a remarkable overcome with respect to the previous Challenge's results – two teams from the Language Computer Corporation reached 73.8 % and 75.4 % accuracy, respectively [3]. One of these latter systems (73.8 % accuracy) exploits the logical entailment between deep semantics and syntax of texts and hypothesises as well as shallow lexical alignment of the two texts [22]; the other one (75.4 % accuracy)

[6] The Association for Computational Linguistics (USA). `http://www.aclweb.org/`.
[7] Natural Language Processing and Information Retrieval Group at the Spanish Distance Learning University. `http://nlp.uned.es/`.
[8] Cross Language Evaluation Forum. `http://www.clef-campaign.org/`.
[9] `http://nlp.uned.es/QA/AVE/`.

utilizes a classification-based approach to combine lexico-semantic information derived from text processing applications with a large collection of paraphrases acquired automatically from the web [14].

Some tasks using RTE are arising and, hopefully, more new tasks will be launched in the near future. These tasks could determine the way in which RTE systems will be developed.

Acknowledgments

This work has been partially supported by the Spanish Ministry of Science and Technology. Project TIC-2003-07158-C04-02: R2D2-SyEMBRA.

References

1. E. Akhmatova. Textual Entailment Resolution via Atomic Propositions. In *Proceedings of the First PASCAL Challenges Workshop on Recognising Textual Entailment, Southampton, UK*, pages 61–64, April 2005.
2. A. Andreevskaia, Z. Li, and S. Bergler. Can Shallow Predicate Argument Structures Determine Entailment? In *Proceedings of the First PASCAL Challenges Workshop on Recognising Textual Entailment, Southampton, UK*, pages 45–48, April 2005.
3. R. Bar-Haim, I. Dagan, B. Dollan, L. Ferro, D. Giampiccolo, B. Magnini, and I. Szpektor. The Second PASCAL Recognising Textual Entailment Challenge. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment, Venezia, Italy*, pages 1–9, April 2006.
4. S. Bayer, J. Burger, L. Ferro, J. Henderson, and A. Yeh. MITRE's Submissions to EU PASCAL RTE Challenge. In *Proceedings of the First PASCAL Challenges Workshop on Recognising Textual Entailment, Southampton, UK*, pages 41–44, April 2005.
5. J. Bos and K. Markert. Combining Shallow and Deep NLP Methods for Recognizing Textual Entailment. In *Proceedings of the First PASCAL Challenges Workshop on Recognising Textual Entailment, Southampton, UK*, pages 65–68, April 2005.
6. P. F. Brown, S. A. Della Pietra, V. J. Della Pietra, and R. L. Mercer. The Mathematics of Statistical Machine Translation. In *Computational Linguistics 19(2)*, 1993.
7. I. Dagan, O. Glickman, and B. Magnini. The PASCAL Recognising Textual Entailment Challenge. In *Proceedings of the First PASCAL Challenges Workshop on Recognising Textual Entailment, Southampton, UK*, pages 1–8, April 2005.
8. R. de Salvo Braz, R. Girju, V. Punyakanok, D. Roth, and M. Sammons. Textual Entailment Recognition Based on Dependency Analysis and WordNet. In *Proceedings of the First PASCAL Challenges Workshop on Recognising Textual Entailment, Southampton, UK*, pages 29–32, April 2005.
9. R. Delmonte, S. Tonelli, M. A. Picollino Boniforti, A. Brsitot, and E. Pianta. VENSES – a Linguistically-Based System for Semantic Evaluation. In *Proceedings of the First PASCAL Challenges Workshop on Recognising Textual Entailment, Southampton, UK*, pages 49–52, April 2005.
10. A. Fowler, B. Hauser, D. Hodges, I. Niles, A. Novischi, and J. Stephan. Applying COGEX to Recognize Textual Entailment. In *Proceedings of the First PASCAL Challenges Workshop on Recognising Textual Entailment, Southampton, UK*, pages 69–72, April 2005.

11. O. Glickman, I. Dagan, and M. Koppel. Web Based Textual Entailment. In *Proceedings of the First PASCAL Challenges Workshop on Recognising Textual Entailment, Southampton, UK*, pages 33–36, April 2005.
12. Z. Harris. Distributional Structure. In J. J. Katz, editor, *The Philosophy of Linguistics*, pages 26–37, 1985.
13. J. Herrera, A. Peñas, and F. Verdejo. Textual Entailment Recognition Based on Dependency Analysis and WordNet. In *Proceedings of the First PASCAL Challenges Workshop on Recognising Textual Entailment, Southampton, UK*, pages 21–24, April 2005.
14. A. Hickl, J. Williams, J. Bensley, K.Roberts, B. Rink, and Y. Shi. Recognizing Textual Entailment with LCC's GROUNDHOG System. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment, Venezia, Italy*, pages 80–85, April 2006.
15. M. Kouylekov and B. Magnini. Recognizing Textual Entailment with Tree Edit Distance Algorithms. In *Proceedings of the First PASCAL Challenges Workshop on Recognising Textual Entailment, Southampton, UK*, pages 17–20, April 2005.
16. D. Lin. Dependency-based Evaluation of MINIPAR. In *Workshop on the Evaluation of Parsing Systems, Granada, Spain*, May 1998.
17. D. Lin and P. Pantel. DIRT - Discovery of Inference Rules from Text. In *Proceedings of ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 323–328, 2001.
18. A. Peñas, Á. Rodrigo, and F. Verdejo. Sparte, a test suite for recognising textual entailment in spanish. In *CICLing*, pages 275–286. Springer, 2006.
19. M. T. Pazienza, M. Pennacchiotti, and F. M. Zanzotto. Textual Entailment as Syntactic Graph Distance: a Rule Based and a SVM Based Approach. In *Proceedings of the First PASCAL Challenges Workshop on Recognising Textual Entailment, Southampton, UK*, pages 25–28, April 2005.
20. R. Raina, A. Haghighi, C. Cox, J. Finkel, J. Michels, K. Toutanova, B. MacCartney, M. C. de Marneffe, C. D. Manning, and A. Y. Ng. Robust Textual Inference using Diverse Knowledge Sources. In *Proceedings of the First PASCAL Challenges Workshop on Recognising Textual Entailment, Southampton, UK*, pages 57–60, April 2005.
21. H. Tanev, M. Kouylekov, and B. Magnini. Combining Linguistic Processing and Web Mining for Question Answering. In *Proceedings of the 2004 Edition of the Text Retrieval Conference*, 2004.
22. M. Tatu, B. Iles, J. Slavik, A. Novischi, and D. Moldovan. COGEX at the Second Recognizing Textual Entailment Challenge. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment, Venezia, Italy*, pages 104–109, April 2006.

Temporal Enhancements of an HTN Planner

Luis Castillo, Juan Fdez-Olivares, Óscar García-Pérez, and Francisco Palao*

Dpto. Ciencias de la Computación e I.A.
University of Granada
siadexwww@decsai.ugr.es,
http://siadex.ugr.es

Abstract. This paper presents some enhancements in the temporal reasoning of a Hierarchical Task Network (HTN) planner, named SIADEX, that, up to authors knowledge, no other HTN planner has. These new features include a sound partial order metric structure, deadlines, temporal landmarking or synchronization capabilities built on top of a Simple Temporal Network [3].

1 Introduction

The achievement of an efficient and expressive handling of time is still a pending task for most HTN planners. This issue becomes even harder if the planner follows a state-based forward paradigm like SHOP [8] or SIADEX [2] since, despite being a very fast HTN planning paradigm, it does not easily allow to obtain temporal plans with concurrent branches. From a practical point of view, real world applications need the plans to have the possibility of executing several activities at the same time. Furthermore, real applications usually require complex synchronization mechanisms between the activities of the plan for a successful execution. This paper explains how the HTN planner SIADEX has been extended to cope with these requirements.

2 Description of SIADEX

SIADEX is a state-based forward HTN planner with the same foundations than SHOP [8]. Before going into the details, some introductory notions are explained first.

2.1 HTN Planning Foundations

HTN planning domains are designed in terms of a hierarchy of compositional activities. Lowest level activities, named actions or primitive operators, are non-decomposable activities which basically encode changes in the environment of

* This work has been completely funded under research contract NET033957/1 with the Andalusian Regional Ministry of Environment.

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 429–438, 2006.

the problem. In SIADEX, these primitive operators are represented as PDDL 2.2 level 3 durative actions [4] (Figure 1.b). PDDL is the standard planning domain description language and it is the basis of most well known planners. On the other hand, high level activities, named tasks, are compound actions that may be decomposed into lower level activities. Depending on the problem at hand, every task may be decomposed following different schemes, or methods, into different sets of sub-activities. These sub-activities may be either tasks, which could be further decomposed, or just actions (Figure 1.a). Tasks and their, possibly multiple, decompositions encode domain dependent rules for obtaining a plan, that can only be composed of primitive actions. Other HTN features are the following ones:

- The initial state is a set of literals that describe the facts that are true at the beginning of the problem.
- Unlike non HTN planners, goals are not specified as a well formed formula that must be made true by the plan from the initial state. Instead, goals are described as a partially ordered set of tasks that need to be carried out.
- The main planning algorithm (Figure 2) takes the set of tasks to be achieved, explores the space of possible decompositions replacing a given task by its component activities, until the set of tasks is transformed into a set of only primitive actions that make up the plan.

```
(:task travel-to
 :parameters (?destination)
 (:method Fly
  :precondition (flight ?destination)
  :tasks (
          (go-to-an-airport)
          (take-a-flight-to ?destination)))
 (:method Drive
  :precondition (not (flight ?destination))
  :tasks (
          (take-my-car)
          (drive-to ?destination))))
```

(a)

```
(:durative-action drive-to
 :parameters(?destination)
 :duration (= ?duration
           (/ (distance ?current ?destination)
              (average-speed my-car)))
 :condition(and (current-position ?current)
                (available my-car))
 :effect(and (current-position ?destination)
             (not (current-position ?current))))
```

(b)

Fig. 1. The basics of HTN planning domains in SIADEX' domain language: (a) A compound *task* with two different *methods* of decomposition. (b) A primitive *action*.

2.2 Including Inference Capabilities in SIADEX

HTN planning approaches have a very rich knowledge representation that may arise in a variety of forms, from methods preconditioning and control of the search [8], ontology based representation of planning objects [6,2] or knowledge-intensive planning procedures [10]. In our case we use two augmented inference capabilities that allows the planner to infer new knowledge, either by abduction or deduction, over the current state of the problem at every planning step.

Regarding the use of deductive inference, SIADEX provides inference tasks that may be fired when needed in a task decomposition scheme. The effect of

- Set $\mathcal{A}$, the agenda of remaining tasks to be done, to the set of high level tasks specified in the goal.
- Set $\Pi = \emptyset$, the plan.
- Set $\mathcal{S}$, the current state of the problem, to be the set of literals in the initial state.
  1. Repeat while $\mathcal{A} \neq \emptyset$
     (a) **Extract** a task t from $\mathcal{A}$
     (b) if t is a primitive action, then
        i. If $\mathcal{S}$ satisfies t preconditions then
           Apply t to the state, $\mathcal{S} = \mathcal{S} + additions(t) - deletions(t)$
           Insert t in the plan, $\Pi = \Pi + \{t\}$
        ii. Else **FAIL**
     (c) if t is a compound action, then
        i. If there is no more decomposition methods for t then **FAIL**
        ii. **Choose** one of its decomposition methods of t whose preconditions are true in $\mathcal{S}$ and map t into its set of subtasks $\{t_1, t_2, \ldots\}$
        iii. Insert $\{t_1, t_2, \ldots\}$ in $\mathcal{A}$.
- **SUCCESS**: the plan is stored in Π.

Fig. 2. A rough outline of an HTN planning algorithm

these inference tasks is that they may produce binding of variables and assert/retract new literals into the current state. The form of these deductive inference rules is

```
(:inline <precondition_list> <consequents_list>)
```

Where both precondition and consequent are logical expressions. Let us consider the task description shown in Figure 3. We may see that the deductive inference rule allows the planner to include a new literal in the state (enabled-wifi) whenever the departure airport has a wi-fi hot-spot.

```
(:task travel-to
 :parameters (?destination)
 (:method Fly
  :precondition (flight ?destination)
  :tasks ((go-to-an-airport)
          (:inline (and (current-position ?airport) (has-wifi-hotspot ?airport))
                   (enabled-wifi))
          (take-a-flight-to ?destination))) ... )
```

Fig. 3. Including deductive rules in the expansion of a high level task

This might be seen as a conditional effect of some previous action but there are some differences that have to be clarified. Firstly, the activities in the decomposition which precede the inference rule may not be primitive actions, therefore, it is not always possible to encode them as conditional effects since high level tasks are not allowed to have effects. And secondly, the problem of the context of firing has to be considered, that is, if we encode every possible deducible consequence of a primitive action as sequence of exhaustive conditional effects, then we might be leading to a very well known problem in planning, the ramification problem [7], and to an overload of the deductive process and unifications with the current state. Therefore, the inclusion of an inference rule into the decomposition of a high level task provides the context in which this inference is necessary and therefore, it will only fire at that moment.

Although the use of deductive inference has also appeared in the literature [9], the use of abductive inference is becoming more widely used in the form of axioms [8] or derived literals, in terms of PDDL 2.2 [4]. These are abductive rules which appear in the form of a Horn clause and that allow to satisfy a given condition when this condition is not present in the current state, but it might be inferred by a set of inference rules of the form

```
(:derived <literal> <logical_expression>)
```

Both deductive and abductive inference rules are extensively used to incorporate additional knowledge to the planning domain while maintaining actions and tasks representations as simple as possible. This is particularly true in real-world planning problems like forest fire fighting plans design, the main application of SIADEX so far. In these cases, planning knowledge usually comes from very different sources, not always related to causality and therefore not very appropriate to encode in the causal structure of actions preconditions and effects. Another of the main purposes of these rules is to allow a process of temporal landmarking over the temporal representation of SIADEX that enrich the set of temporal constraints that may be encoded in a planning problem.

3 Temporal Enhancements of SIADEX

One of the main drawbacks of state-based forward planners (HTN and non-HTN) is that they usually return plans as a total order sequence of activities, that is, a chain of actions.

$$\Pi = \{a_1, a_2, \ldots, a_n\}$$

If the planner is based on states, then, this sequence of actions also induces a sequence of states.

$$INITIAL + s_1 + s_2 + s_3 + \ldots + s_n$$

where s_i is the state that results from the execution of action a_i over the state s_{i-1}. However, in many real world applications, several activities may be carried out in parallel and a total order of activities is not very appropriate for practical reasons. In order to obtain plans with parallel branches, SIADEX uses several techniques: the definition of qualitative partial order relationships in the domain, the inference of new metric temporal constraints from the causal structure of the plan, the definition of deadlined goals and complex synchronization schemas. These techniques are based on the inference capabilities explained before and the handling of metric time, and they are explained in the forthcoming sections.

3.1 Qualitative Orderings

One of the main sources of temporal constraints between the actions of the plan comes from the order in which the subtasks of a given task appear in its decomposition. In this way, SIADEX may impose three different types of qualitative ordering constraints in every decomposition.

Sequences. They appear between parentheses `(T1,T2)`, and they are a set of subtasks that must execute in the same order than the decomposition, that is, first `T1` and then `T2`. Please note that any possible subtask of `T1` and `T2` inherit these relations too, so that the imposed ordering is maintained even through their decompositions.

Unordered. They appear between braces `[T1,T2]`, and they are a set of subtasks that are not ordered in their decomposition, that is, either `T1` or `T2` could execute first.

Permutations. They appear between angles `<T1,T2>`, and they are set of subtasks that must execute in any of the total orders given by any of their permutations, that is, first `T1` and then `T2` or vice versa. In these cases, the choice of the best permutation takes part in the search process of the planner. Please note that any possible subtask of `T1` and `T2` also inherits these relations.

For example, a method that decomposes the task `t` into `(T1 [T2 (T3 T4)] T5)` represents a partially ordered decomposition depicted in Figure 4.a) and the plan obtained by SIADEX is shown in Figure 4.b).

(a) (b)

Fig. 4. A graphical representation of the partial order decomposition given by `(T1 [T2 (T3 T4)] T5)` (left hand side) and the plan representation found by SIADEX (right hand side)

3.2 Inferring New Quantitative Orderings From Causal Links

The qualitative orderings seen before allow to encode relatively simple order relations that are the main sources of partial ordering in the final plan. But there is an additional source of ordering information that is very useful for encoding additional metric temporal constraints: the causal structure of actions. This section explains how to enhance the representation of states with temporally annotated literals, and how to exploit the existence of causal links between primitive activities to encode sharply defined metric temporal constraints between them, providing a more precise, but complementary, source of temporal constraints than these qualitative orderings.

All the literals l_j^k in the effects of every action a_j have a delay Δt_j^k by which they are achieved after the execution of their corresponding action

$$\forall a_j \in \Pi = \{a_1, a_2, \ldots, a_n\}, l_j^k \in \ effects(a_j), 0 \leq \Delta t_j^k \leq duration(a_j)$$

This is a generalization of PDDL 2.2 `at-start` and `at-end` effects

$$l_j^k \in effects(a_j), l_j^k \; is \; at-start \iff \Delta t_j^k = 0$$

$$l_j^k \in effects(a_j), l_j^k \; is \; at-end \iff \Delta t_j^k = duration(a_j)$$

SIADEX represents states as a temporally annotated extension of classical states, where every literal is timestamped with the time by which it is achieved with respect to the action that produced it. Therefore, given a total order sequence of actions and its induced sequence of states

$$\Pi = a_1, a_2, \ldots, a_n \rightarrow INITIAL + s_1 + s_2 + s_3 + \ldots + s_n$$

every state s_i is given by $s_i = \{< l_j^k, a_{j,j \leq i}, \Delta t_j^k >\}$, that is, a set of literals l_j^k coming either from the effects of some previous action a_j ($l_j^k \in effects(a_j)$) or from the initial state (in this case the literal would have the form $< l_0^k, 0, 0 >$). This means that literal l_j^k was introduced in the state s_i Δt_j^k time units after the execution of a_j. Then, a temporally annotated state allows to know which action, if any, produced that literal amongst the preceding actions and at which time they were achieved. Therefore, once an action has satisfied its preconditions (either `at-start` or `over-all` in PDDL 2.2) one may define its execution time as the time needed to achieve all its preconditions given the temporal information annotated in the state. Let us consider $base(a_j)$ as the set of literals in the state s_{j-1} that do not come from the initial state and that were used to satisfy a_j preconditions:

$$base(a_j) \subseteq s_{j-1}$$

$base(a_j) = \{< l_i^k, a_i, \Delta t_i^k >_{i<j, a_i \neq 0}, \exists c_j^i \in at-start(a_j) \cup over-all(a_j), c_j^i \text{ unifies } l_i^k\}$

then the time of execution of action a_j, say $t(a_j)$, is given by this recursive equation

$$t(a_j) = \begin{cases} 0 & if \; base(a_j) = \emptyset \\ \max\limits_{< l_i^k, a_i, \Delta t_i^k > \in base(a_j)} \{t(a_i) + \Delta t_i^k\} & otherwise \end{cases}$$

In summary, this process explores the causal dependencies between actions and adds specific temporal constraints to the qualitative temporal constraints included formerly, in a two-step refinement process: once an abstract task is decomposed into subtasks, low detail temporal constraints are introduced in the plan (section 3.1). This could be seen as the plan's temporal skeleton. Once its sub-actions are being included definitely in the plan, more precise temporal constraints are added to encode the causal relationships between them (section 3.2). All these temporal constraints are stored in a Simple Temporal Network [3] so, at the end of the planning process, the plan does not have to have a total order structure, but the minimum required temporal constraints to ensure a correct causal structure.

4 Deadlines and Temporal Landmarks

The process explained so far is devoted to unfold the total order relation in which actions are obtained (see Figure 2) into a partial order plan that only records the causal structure of the plan, as a least commitment unfolding strategy (Figure 4.b). However, some real problems require the posting of more complex temporal constraints between the actions of a plan that not always have a cause-effect relationship. In this section, two additional enhancements are presented devoted, on one hand side, to deadline goals, that is, goals that must be achieved at a certain time, and in the other hand, complex synchronization schemes to allow actions to interact along the time.

4.1 Deadline Goals

A deadline activity (either a task or an action) is an activity that may have defined one or more metric temporal constraints over its start or its end or both. Furthermore, in the case of tasks, all its component subtasks will also inherit these constraints. SIADEX also allows to post deadline constraints on the start or the end of an activity (or both). In order to do that, the activity is preceded by a logical expression that defines the desired deadline as it is shown in Figure 5. There may be seen that any sub-activity (either task or action) has two special variables associated to it: `?start` and `?end` and that some constraints (basically `<=, =, >=`) may be posted to them. Deadline goals may also appear in the top level goal and they are very useful for defining time windows of activity, that is, sets of activities

```
(:task A
 :parameters ()
 (:method A
  :precondition ()
  :tasks ((A1)
          ((and (<= ?start 3)(>= ?end 5)) (A2)))))
```

Fig. 5. Definition of deadlines in the decomposition of a task `A` stating that subtask `A2` must appear after `A1` but it must start at a time unit less than or equal to 3 and it must end at a time unit greater than or equal to 5

```
(:task A2
 :parameters ()
 (:method A2
  :precondition ()
  :tasks ((:inline ()
           (and (assign (start A2) ?start)
           (assign (end A2) ?end)))
         (a21)
         (a22))))
```

(a)

```
(:durative-action b
 :parameters()
 :duration (= ?duration 1)
 :condition()
 :effect(and (assign (start b) ?start)
             (assign (end b) ?end)))
```

(b)

Fig. 6. Generating temporal landmarks both for a task (a) and for an action (b). These landmarks are treated as fluents but they really represent time points of the underlying Simple Temporal Network underlying the plan, enabling the posting of additional constraints over them.

that must be executed within a given temporal interval like observation activities of a satellite or civil/military interventions in the case of crisis.

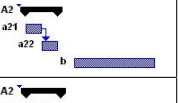
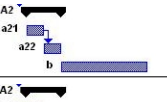
4.2 Temporal Landmarking and Complex Synchronizations

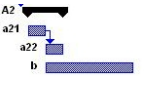
SIADEX is also able to record the start and end of any activity and to recover these records in order to define complex synchronizations schemes between either tasks or actions as relative deadlines with respect to other activities. The first step is the definition, by assertion, of the temporal landmarks that signal the start and the end of either a task (Figure 6.a) or an action (Figure 6.b). These landmarks are treated as PDDL fluents, and they are associated to the time points of the temporal constraints network and, therefore, fully operational for posting constraints between them.

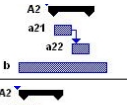
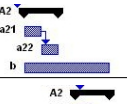
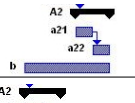
```
(:task A3
 :parameters ()
 (:method A3
  :precondition (...)
  :tasks (((= ?start (start A2)) (b)))))
```

Fig. 7. Recovering a temporal landmark in order to define a synchronization scheme in which action `b` starts exactly at the same time that task `A2`

Table 1. Encoding all Allen's relations between task `A2` and action `b`. In all the cases action `b` has a duration of 5 time units except in the `equal` relation, whose duration is 2 time units.

Allen's relation	SIADEX encoding	SIADEX output
A2 b **A2 BEFORE b**	`((> ?start (end A2)) b)`	
A2 b **A2 MEETS b**	`((= ?start (end A2)) b)`	

A2
b
A2 OVERLAPS b
	`(and (> ?start (start A2))`
	`(< ?start (end A2))`
	`(> ?end (end A2)) b)`
		

A2 b **A2 DURING b**	`(and (< ?start (start A2))` `(> ?end (end A2)) b)`	
A2 b **A2 STARTS b**	`((= ?start (start A2)) b)`	
A2 b **A2 FINISHES b**	`((= ?end (end A2)) b)`	
A2 b **A2 EQUAL b**	`(and (= ?start (start A2))` `(= ?end (end A2)) b)`	

These landmarks are asserted in the current state, and later on, they may be recovered and posted as deadlines of other tasks in order to synchronize two or more activities. For example, Figure 7 shows the constraints needed to specify that action `b` must start exactly at the same time point than task `A2`.

In particular, thanks to the expressive power of temporal constraints networks and to the mechanism explained so far, a planning domain designer may explicitly encode in a problem's domain all of the different orderings included in Allen's algebra [1] between two or more tasks, between two or more actions or between tasks and actions. Table 1 shows how these relations may be encoded between task `A2`, composed of actions `a21` and action `a22` with a duration of 1 time unit each, and action `b` with a duration of 5 time units.

5 A Short Note on Efficiency

SIADEX is performing very well in its main domain of use, i.e. generating forest fire fighting plans form the Andalusian Regional Government of the Environment [2], but it is also very competitive compared to other state of the art HTN planners like SHOP2 [8]. The handling of Simple Temporal Networks [3] implies both a high expressiveness in the handling of time for a planner, but it also implies an additional overload of computation time since consistency checking in large scale temporal networks (in the order of thousands of time points) is a difficult problem. Despite this computational overload, SIADEX has outperformed SHOP2 in a small test that is shown in Figure 8. This graphic shows the performance of both planners in the same domain known as zenotravel [5] and in the harder time problems (those with a greater difficulty in the handling of time).

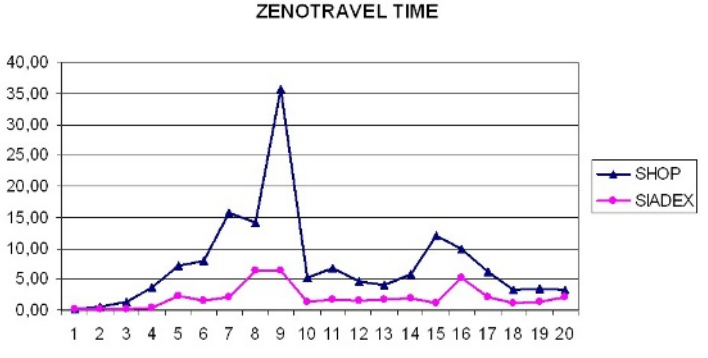


Fig. 8. Performance of SHOP2 and SIADEX in zenotravel time problems. Y-Axis represents computation time in seconds and X-Axis represents the set of test problems.

6 Conclusions

In summary, this paper has presented several valuable temporal extensions of an HTN planner that allow to cope with a very rich temporal knowledge

representation like temporal causal dependencies, deadlines, temporal landmarks or synchronization schemes. These capabilities have been found to be of extreme necessity during the application of SIADEX to the research contract NET033957 with the Andalusian Regional Ministry of Environment for the assisted design of forest fighting plans and, up to authors's knowledge, no other HTN planner has these capabilities for handling temporal constraints.

References

1. J.F. Allen. Maintaining knowledge about temporal intervals. *Comm. ACM*, 26(1):832–843, 1983.
2. M. de la Asunción, L. Castillo, J. Fdez-Olivares, O. García-Pérez, A. González, and F. Palao. Siadex: an interactive artificial intelligence planner for decision support and training in forest fire fighting. *Artificial Intelligence Communications*, 18(4), 2005.
3. R. Dechter. *Constraint processing.* Morgan Kaufmann, 2003.
4. S. Edelkamp and J. Hoffmann. The language for the 2004 international planning competition. http://ls5-www.cs.uni-dortmund.de/ edelkamp/ipc-4/pddl.html, 2004.
5. M. Fox and D. Long. Domains of the 3rd. international planning competition. In *Artificial Intelligence Planning Systems (AIPS 02)*, 2002.
6. Y. Gil and J. Blythe. PLANET: A shareable and reusable ontology for representing plans. In *AAAI 2000 workshop on representational issues for real-world planning systems*, 2000.
7. S. A. McIlraith. Integrating actions and state constraints: A closed-form solution to the ramification problem (sometimes). *Artificial Intelligence Journal*, pages 87–121, 2000.
8. D. Nau, T.C. Au, O. Ilghami, U. Kuter, J. W. Murdock, D. Wu, and F. Yaman. SHOP2: An HTN Planning System. *Journal of Artificial Intelligence Research*, 20:379–404, 2003.
9. D. E. Wilkins. *Practical planning: Extending the classical AI planning paradigm.* Morgan Kaufmann, 1988.
10. D. E. Wilkins and M. desJardins. A call for knowledge-based planning. *AI Magazine*, 22(1):99–115, 2001.

The Multi-Team Formation Defense of Teamwork

Paulo Trigo[1] and Helder Coelho[2]

[1] Instituto Superior de Engenharia de Lisboa, Departamento da Engenharia da Electrónica e Telecom. e de Computadores, R. Conselheiro Emídio Navarro, 1, 1949-014 Lisboa, Portugal
[2] Faculdade de Ciências da Universidade de Lisboa, Departamento de Informática, Bloco C5, Piso 1, Campo Grande, 1749-016 Lisboa, Portugal
[1]ptrigo@isel.ipl.pt, [2]hcoelho@di.fc.ul.pt

Abstract. We formulate the multi-team formation (M-TF) domain-independent problem and describe a generic solution for the problem. We illustrate the M-TF derogation component in the domain of an urban fire disaster.. The M-TF problem is the precursor of teamwork that explicitly addresses the achievement of several short time period goals, where the work to achieve the complete set of goals overwhelms the working capacity of the team formation space (all teams formed from the finite set of available agents). Decisions regarding team formation are made considering that team reformation is the means to counteract possible deviations from a desirable teamwork behavioral performance. The RoboCupRescue large-scale disaster environment is used to illustrate the design of the derogation domain-specific M-TF component.

1 Introduction

Teamwork has emerged as the dominating coordination paradigm for multiple agents aiming to achieve goals within dynamic, uncertain and hostile environments [1]. Despite the progress in multi-agent teamwork the approach to operational contexts is usually founded on the prevailing teamwork theories [2], [3], which in turn do not explicitly assume resources (agents and goals) to be bounded, neither teamwork to be repeatedly applied as the number (and difficulty) of goals increase.

The operational perspective of teamwork is that team formation repeatedly occurs over time within an environment where agents and goals are resource-bounded elements. The *agent availability* is the first boundary; the agent set is finite and each agent only belongs to a single team. The *time to goal* achievement is the second boundary; goals are to be achieved within pressing deadlines.

The operational perspective of teamwork implies the continuous and online team formation and team reformation as the way to better shape the overall teamwork to the overwhelming work to be done. Team formation is the process of how best to organize the agents into a collaborating team in order to achieve a specific goal [4]. Team reformation is the process that is triggered once the current team activity (teamwork) is no longer defensible; a goal is no longer believed to be achievable with the existing teams' organization. Thus, team reformation is a reorganization process that aims to better equip the whole teamwork (all set of teams) to achieve the overall existing goals. Team reformation possible reorganizations include the reinforcement

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 439 – 448, 2006.

of a team's power (e.g. augment the number of team members) and the team disbandment (hence members reinforce other, more defensible, teams).

The operational perspective of teamwork is best understood in the large-scale disaster mitigation domain. Within a large-scale disaster environment (e.g. earthquake or terrorist incident) several simultaneous goals (e.g. extinguish fires in buildings) are to be achieved as soon as possible given the work of a finite number of specialized teams (e.g. the available fire brigade teams). The RoboCupRescue [5] is a simulation environment of large-scale disasters. It is a resource-bounded simulated context that brings forth a semi-optimal behavior planning problem with extremely complex constraints and having time-varying multiple goals [6].

The operational perspective of teamwork, materialized during the participation at the RoboCupRescue 2004 International Championship (with the 5Rings team [7]), shaped our formulation of the multi-team formation (M-TF) problem.

We conceive the M-TF problem as the most general precursor of teamwork. The M-TF is defined as a domain-independent problem that explicitly accounts for both a global and a local rationality for teamwork (respectively, the preference relation, $\Psi\gamma$, component and the derogation function, X_G, component) and also explicitly considers the impact of simple futuristic causal relations (the expected achievement goal set, Λ_G, component) on the teamwork performance. In this paper we illustrate the design of the derogation function, X_G, component in the RoboCupRescue domain.

The solution of the M-TF problem is a team formation decision. We formulate two basic decision strategies. One strategy searches for the minimal global cost of commitments. The other strategy considers that higher importance goals "choose agents first"; it searches for several locally minimal cost commitments.

The Section 0 of this paper defines the M-TF problem and the Section 0 describes the general solution of the M-TF problem. The Section 0 illustrates the design of the M-TF domain specific derogation component, in the context of RoboCupRescue environment. The Section 0 summarizes our proposals and outlines the future work.

2 The Multi-Team Formation (M-TF) Problem

Intuitively we see the multi-team formation as the problem to decide, at each time instant, the most important goals to achieve and the most effective agents to establish a joint commitment to achieving each goal, taking into account expected (future) goals and the team reformation (reorganization) required to counteract possible deviations from a desirable behavioral performance.

Given, at a time t, a finite set of agents, α, who intend to perform some joint task, a finite set of achievement goals, γ, and a finite set of teams, τ, we pretend to build a new set of teams, τ', committed to achieve the goals at γ that, at time t are unattended.

The M-TF components are represented as a 3-tuple $< \Psi\gamma, \{\Lambda_G\}, \{X_G\} >$, for all $G \in \gamma$, where $\Psi\gamma$ is a preference relation over γ, Λ_G is the set of the expected achievement goals given that G is an achievement goal and X_G is the derogation function concerning G.

The preference relation, $\Psi\gamma$, defines a total order relation over the set of all achievement goals. This relation materializes the global rationality for teamwork which gives higher priority to achieving those goals that, given the current knowledge, are

expected to maximize a global performance measure. We refer to [8] for the formal and comprehensive presentation of the $\Psi\gamma$ component.

The expected achievement goal set, Λ_G, is a concept that captures the necessity of a teamwork rationality that accounts for expectations regarding futuristic causal relations in the domain. We adopt a relaxed approach to futuristic causality, such as to separately consider each achievement goal as the main cause for future effects. The reduced problem is stated as "given that G is an achievement goal, what are the propositions that I believe will turn false in a near future world?".

The derogation function, X_G, represents the local rationality for the team committed to achieve G, throughout the time period of such commitment. We consider that, regarding goal G, a *locally rational teamwork*, selects the actions expected to minimize the effects of not working at all to achieve G.

Intuitively the derogation function represents the justification (defense) for a team to keep committed working to achieve a specific goal; or, the other way around, it expresses the penalization for total agents' inaction regarding that goal.

The design of a derogation function, given an achievement goal G, starts from the identification of the set of world features, $F_{X,G}$, that eventually get devaluated for not achieving G. Therefore, the derogation function, X_G: t, $\prod_i \Xi_i \rightarrow \Re$, represents the expected progress along time, t, for the domain, Ξ_i, of each feature $f_i \in F_{X,G}$.

The design of each $\Psi\gamma$, Λ_G and X_G component is a domain-specific task to be attained by the agent's designer with the hopeful assistance of a domain specialist.

3 The Multi-Team Formation (M-TF) Solution

We consider that the M-TF search for a solution is a two step process. The first step's input is the $\Psi\gamma$, Λ_G and X_G domain-dependent specifications while the output is the instance of $\Psi\gamma$, Λ_G and X_G, given the available perceptions (evidences) at each time instant. The second step takes the previous step's output and builds a new set of teams, τ'.

The new set of teams, τ', depends on $\Psi\gamma$, Λ_G and X_G instances; the Fig. 1 "Prolog like" syntax (with boldface output parameters), shows the directives of the predicate used to build the τ' instance.

```
m-tf( Time, τ, γ, α, Ψγ, {X_G}, τ' ):-
    achievedGoals( Time, τ, γ, AG ),
    impossibleToAchieveGoals( Time, γ, {X_G}, IAG ),
    agentsNotCommitted( Time, τ, α, AG, IAG, β ),
    bestMultiTeamFormation( Time, Ψγ, β, τ' ).
```

Fig. 1. The Multi Team Formation Predicate

Each output parameter (boldface at Fig. 1) is defined as follows:

- AG ≡ the difference between the set of goals for which there is a team (in τ) and γ; any goal that is not already in γ is assumed to have been achieved,

- IAG ≡ the set of G goals such X_G(Time) > ε, where ε is the maximum penalization that any team is willing to accept, while committed to achieving G,
- β ≡ the difference between α and the set of committed agents, which are those that belong to a team (in τ) whose goal does not belong to AG neither to IAG,
- τ' ≡ set of teams, where each $T_G \in \tau'$ is a team with the lowest team formation cost to achieve G, subject to the following guidelines i) for any two goals, $g_1 \in \gamma$ and $g_2 \in \gamma$, such that $\Psi(g_1) > \Psi(g_2)$, then the team formation of Tg_1 is to be achieved before the team formation of Tg_2 (informally we say that the priority of Tg_1 is higher then that of Tg_2), and ii) the only teams under consideration are those with sufficient number of agents to achieve the goal. T_G.

To formally express the above τ' guidelines, we introduce the α_i and $|s|$ symbols. We write α_i ($i \geq 1$) to represent the set of agents committed to achieve the G_i goal, which is the i^{th} goal (as defined by $\Psi\gamma$); we recall that β symbolizes the set of agents not committed to any goal (at a certain time instant). We write $|s|$ to represent the cardinality of the *s* set. We have $|\alpha_i| = 0$ ($i \geq 1$) whenever the set of available agents is not sufficient to achieve the G_i goal. Also, during team formation, the construction order of each α_i follows the $\Psi\gamma$ prescribed (highest lo lowest) order.

Our team formation guidelines are formulated as follows:

$$\begin{gathered} \alpha_1 \subseteq \beta, \text{ and } \alpha_{i+1} \subseteq \beta - \cup_{j=1..i}\, \alpha_j \\ \text{subject to, } \Psi(G_i) > \Psi(G_{i+1}), \text{ for } i=1..|\gamma| - 1 \\ \text{such that, } |\alpha_i| > 0 \Leftrightarrow |\alpha_i| = \text{“sufficient team cardinality to achieve } G_i\text{”} \end{gathered} \tag{1}$$

The “sufficient team cardinality to achieve a goal” is a function to be designed in the context of each specific domain; it depends on the domain and also on the available perception of the environment.

In order to define the lowest team formation cost to achieve a goal, G_i, we specify the cost function as follows:

$$\begin{gathered} \text{cost}(\alpha_i, G_i) = \text{cost}_{\text{formation}}(\alpha_i, G_i) + \kappa \times \Sigma_{r=1..m}\, \text{cost}_{\text{reformation}}(T_r), \\ \text{where, } T_r \text{ is a team in } \tau \text{ with agents in } \alpha_i \text{ (a team to suffer reformation)} \\ \kappa \geq 0, \text{ and } m = \text{number of team reformations to build } \alpha_i. \end{gathered} \tag{2}$$

The $\text{cost}_{\text{formation}}(\alpha_i, G_i)$ is a domain dependent metric that represents the effort of α_i agents to begin working to achieve G_i; e.g. the total (or maximum) amount of time spent, to physically move all α_i agents to the geographical location of G_i.

The $\text{cost}_{\text{reformation}}(T_r)$ is a domain dependent metric that represents the reformation effort of T_r team; e.g. the derogation value of the team at that time instant.

We now use equation (2) to specify two basic strategies to build each $T_G \in \tau'$. Both strategies comply with the team formation guidelines taken from equation (1).

The first strategy, described at equation (3), is to build the globally best agent to goal commitment.

$$\text{argmin}_{<\alpha_1, \alpha_2, \ldots, \alpha_n>} \Sigma_{i=1..|\gamma|}\, \text{cost}(\alpha_i, G_i) \tag{3}$$

The second strategy, described at equation (4), is to build each goal's locally best agent to goal commitment. With this strategy we give a higher priority goal the privilege to first choose the best agents to achieve that goal.

$$\text{argmin}_{\alpha_i} \text{cost}(\alpha_i, G_i), \text{ for } i=1..|\gamma| \quad (4)$$

A third strategy is to apply equation (4) for some first goals and equation (3) for the remaining goals. Such strategy is to be applied when a big gap (over a certain threshold) exists between some first $\Psi\gamma$ values and the remaining values.

4 The M-TF Problem Illustration

For concreteness and to illustrate the design of the M-TF domain-dependent components we materialize the X_G component in the context of an illustrative domain.

The illustrative domain is that of an urban fire disaster. The RoboCupRescue simulation environment is used to simulate a fire scenario that evolves at the Nagata ward in Kobe, Japan (an official city map used at RoboCupRescue competition).

The fire ignites in a building, B, and to simplify our illustrative scenario we assume that: i) the fire is not affected by other neighborhood fires, and ii) the environmental circumstances (e.g. wind or air temperature) do not influence the fire evolution.

The design of the Derogation Function. The design of X_G starts from the identification of the set of world features, $F_{X,G}$, that eventually get devaluated for not achieving G. Within the RoboCupRescue environment this is an easy task, for the reason that there is a global evaluation of the simulation performance. The global evaluation measures both the human and the building damage (lower values are better for both criteria); therefore, given that our illustrative example only considers the building damage, $F_{X,G} = \{\text{building\_damage}\}$.

Intuitively the $X_{\text{extinguish(B)}}$ function represents a baseline by which, in the course of a simulation, the B building's observed damage is compared. Such baseline depicts the B building's damage evolution during the time period, Δ, of fire; from the fire catch time instant to the total destruction time instant.

The baseline for the damage evolution destruction assumes a linear destruction along time, t, during a Δ time period of fire, such that:

- $A_B(0) = A_{B,start}$,
- $A_B(t) = A_B(t-1) - A_{B,start} / \Delta$, for $t = 1, 2, 3, \ldots \Delta$,
  where, $A_{B,start}$ is the total area of B building; $A_B(t)$ is the total, not burnt area of B building, at t time; and Δ is the time period from fire catch until total destruction.

The above formulation is normalized so that it can be used to compare different Δ extinguish periods and different buildings' initial conditions ($A_{B,start}$, different values):

- $A_B(n(t)) / A_{B,start} = 1 - n(t)$, where $n(t) = t / \Delta$.

We now closely follow the RoboCupRescue damage evaluation function to define the derogation function:

$$X_{extinguish(B)}(n(t)) = [\ 1 - n(t)\ ]^{½} \tag{5}$$

The equation (5) depends on Δ and is independent of B (our baseline is a linear destruction along Δ). In the course of a simulation, the baseline destruction is compared with the observed evolution of the B building's destruction; hence, we measure the difference, at t time instant, between equation (5) and equation (6).

$$[\ A^{O}_{B}(n(t)) / A_{B,start}\ ]^{½}, \tag{6}$$

where $A^{O}_{B}(n(t))$ is the observed not burnt area of B building at t time.

We are more interested on the difference tendentiousness along time, and less interested about each t time step's observed difference. Also such difference is to be measured by a monotonous index that follows the increase/decrease of the goals achievement performance; we call it the "performance index".

Hence, we define the performance index as a parameter of the derogation function derivative, which can be used to depict several performance behaviors, as follows:

$$\partial/\partial n(t)\ X_{extinguish(B)}(n(t)) = p(t, 2) \tag{7}$$

$$p(t, K) = - ½ \times [\ 1 - n(t)\ ]^{-1/K},\ K \in \Re \tag{8}$$

The K value is the performance index, and we say that K=2 represents the *theoretical baseline value* (tBv); the higher the K value the better the performance. The tBv traduces the theoretical evolution of a building's destruction by fire, given that no effort is made to extinguish such fire.

To use this framework (derogation function and performance index) we register the K value associated with the observed destruction, $A^{O}_{B}(n(t))$, value at each t time step. The K value is given, from the manipulation of equation (6), as follows:

$$K = - 1 / \log_{(1 - n(t))} [\ - 2 \times A^{O}_{B}(n(t))\ ] \tag{9}$$

The set of all registered K values is then used to calculate the K that "suitably" describes the performance behavior; to simplify we define "suitably" as the Ks' set average value.

The main purpose of the K value is to assist a decision maker (e.g. the fire brigade's commander) in concluding on the efficacy of a team's activity. Yet, in order to conclude, the decision maker needs to follow a confrontation pattern; the observed K value must be compared with a reference value. One possibility is to use the tBv (K=2) as the reference value. But, the tBv is a theoretical reference and likely needs to be calibrated in order to traduce the observed (practical) evolution of a building's destruction by fire. It is admissible that the building properties (e.g. ground area, total area and construction materials) affect the course of destruction by fire.

Therefore, the simple observation of K values, in the absence of any effort to extinguish the fire, enables to calibrate of the tBv indicator and to achieve a *calibrated baseline value* (cBv). Throughout simulation it is possible to calculate the cBv of each building or class of buildings (e.g. those with similar areas and similar construction materials). Although the cBv is operationally relevant, we get on with our simple illustrative purpose and adopt the tBv as our reference value; as if in the absence of simulation (or historical) information.

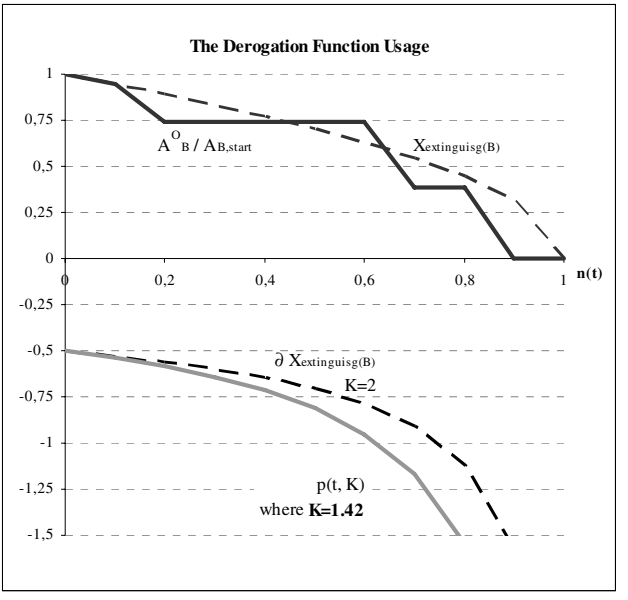


Fig. 2. A fire evolution analysis of B building using the derogation function

The Fig. 2 illustrates the evolution of a fire during a Δ=10 total destruction fire period of the B building (where $A_{B,start}$=194.02). In the upper half of Fig. 2, the dashed line is the derogation function, cf. equation (5), and the dark solid line is the observed destruction, cf. equation (6), along time. In the lower half, the dashed line is the derogation function's derivative, cf. equation (7), and the gray solid line is the performance behavior, cf. equation (8), where K=1.42. The K value is the average of all Ks, cf. equation (9), registered at each time step; at t=1 $A^O_B(n(1))$=174.62; from t=2 to t=6, $A^O_B(n(t))$=106.71; from t=7 to t=8, $A^O_B(n(t))$=29.10 and at t=9 the building was observed to be fully destroyed.

The analysis of a team of fire brigade agents, with the performance depicted at Fig. 2, yields the immediate conclusion that, regarding the building's destruction by fire, the team performed poorly while trying to extinguish the fire (K value is below the tBv, i.e. K<2).

The first defense on a poor team performance is that the team is sub-dimensioned. This indicates that the number of fire brigade agents, used at Fig. 2, was insufficient to extinguish the B building's fire. The deficient number of fire brigade agents in the team reveals either an inadequate situation evaluation during the team formation stage or individual agent's failure (e.g. fire brigade caught by fire) without proper substitution during the team activity, or both deficiencies.

The second reason for a poor team performance is that the team activity started later then expected when compared to the goal occurrence. This indicates either a late (or delayed) team formation or an unpredictable delay of fire brigades (e.g. road blockades, traffic jams), or both deficiencies.

The third element to consider in the analysis of a poor performance is whether the performance index is below the tBv (or cBv, when available). Under the assumption that the tBv is a valid reference, then when K gets below tBv it indicates the effect of elements that were not considered in the estimation of the Δ time period (total destruction period). For example, a strong wind and neighbors in fire are likely to affect (reduce) the Δ time period.. This is an opportunity to create a new derogation function that is specialized on that particular situation. On the other hand, if we come to conclude that the situation is not a special one, then our initial assumption (that the tBv is a valid reference) is incorrect. In this case it is an opportunity to create (or adjust) the cBv indicator.

We note that the simplest way to increase the performance (K value) is to lengthen the time period of the highest not burnt area. It represents a situation where team activity started on time, even though sub-dimensioned. For example, if from t=1 to t=3, $A^{O}_{B}(n(t))=174.62$ or, in other words, having two additional time steps (one third of the total burning time) at the highest not burnt area (and after that, the same evolution as above), we would get K=2.3 which is slightly better then the tBv. Even though K>tBv the fire did not extinguish before the total destruction of the building, which indicates a sub-dimensioned team to extinguish the fire.

Our derogation framework also supports the detailed analysis of the fire evolution. As an example the Table 1 displays the analysis of each consecutive three time step periods (from t=1 to t=3, then from t=2 to t=5, then from t=3 to t=6, etc). The Table 1 "t" labeled column represents each time interval and the "S1" and "S2" labeled columns show the K values' average at the corresponding time interval. The "S1" and "S2" are two different fire evolution scenarios. The "S1" scenario is the same as the one described at Fig. 2. The "S2" scenario, just as the above example suggests, considers that the three initial time steps hold the same (highest) not burnt area and after those three steps, the evolution is the same as in "S1".

The Table 1 facilitates an in-depth analysis of "S1" and "S2" scenarios. We notice that the problem at "S1" is located before the fourth time interval and after the sixth time interval (K<2, which indicates that performance is below the baseline).

One reading of the "S1" scenario is that the initial destruction was not be sustained and although the building was kept in a moderate destruction level for a while (the three lines where K>2), it suddenly collapsed.

t	S1	S2
[1..3]	1.31	***4.33***
[2..4]	1.22	***4.24***
[3..5]	1.74	***3.60***
[4..6]	***2.36***	***2.36***
[5..7]	***2.22***	***2.22***
[6..8]	***2.01***	***2.01***
[7..9]	0.99	0.99
[8..10]	0.64	0.64

Table 1. Detailed analysis of the fire evolution

The analysis of teamwork at S1 indicates the possibility of a late team activity. The rapid initial destruction (K<2 at the first three lines) may have been caused by a fragmented team activity, followed by the effective fire contention (K>2 from fourth to sixth line) of a fully operating team. Given that teamwork was tardy, the team reinforcement (additional fire brigades) if available, might have prevented the final sudden collapse of the building.

On the other hand, at "S2", we see that the performance is highly prized (K>2 at the first six lines). There is an highly positive influence of the initially sustained destruction (K>3 at the first three lines). After that initial positive behavior, the building was kept in a moderate destruction level for a short period and suddenly collapsed (just as in "S1").

The analysis of teamwork at S2 indicates that although the team seems to have the capability to sustain the fire, the team felt short of resources (e.g. fire brigades or water) in the course of its activity. Three possible justifications for this behavior are: i) inadequate reformation of the team (e.g. overmuch reduction of members), ii) occurrence of neighborhood fires (e.g. fire brigade dispersal between fires), and ii) deficient management of the extinguishment resources (e.g. long water refill periods).

The composition of different perspectives (e.g. global as in Fig. 2 and in-depth as in Table 1) yields a mature understanding of the team activity and provides the arguments for the team reformation.

5 Summary and Future Work

This work addresses a major shortcoming of the current work in team formation for dynamic real-time domains: the formation of multiple teams in response to the occurrence of several simultaneous achievement goals.

The shortcoming is most relevant within RoboCupRescue-like complex environments; hostile, uncertain and dynamically changing environments where each achievement goal overwhelms the personal capability and the complete set of achievement goals overwhelms the total teamwork capability. The shortcoming was addressed by the formulation of the M-TF problem. The M-TF problem formulation assumes a set of achievement goals and a set of existing teams and constructs the team formations and reformations required by those achievement goals.

The illustrative design of the X_G component uses the RoboCupRescue environment and defines a baseline function that depicts the evolution of a fire in a building given the elapsed time from the fire catch time instant to the building's total destruction time instant. We picked a parameter of the baseline function such that the increase/decrease of the parameter represents a better/worse fire evolution, given the goal to extinguish the fire. We called such parameter the "performance index" and used it to measure the team activity throughout time. The derogation function and the corresponding performance index assist the decision maker (human or agent) in the defense of a team's activity (teamwork); it sustains the decision on whether to trigger the team reformation process.

We consider the multi-team formation an important teamwork issue within complex domains (e.g. the RoboCupRescue) and we pretend to further exploit the formation of teams equipped with the multitude of capabilities that are expected to be

required in the near future. We are currently exploring the usage of simulations to learn the value of team activity (perception to action transitions), given that agents acquire too much sensory raw data and that their perceptual limitations is likely to hide environmental crucial features.

Acknowledgements

This research has been partially supported by the program PRODEP III 5.3/13/03.

References

[1] N. Schurr, S. Okamoto, R. Maheswaran, P. Scerri, and M. Tambe, *Cognition and Multi-Agent Interaction: From Cognitive Modeling to Social Simulation*, chapter entitled *Evolution of a Teamwork Model*, Cambridge University Press, 2004.

[2] A. Rao and M. Georgeff, "Modeling rational agents within a BDI-architecture," *in Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning (KR&R-91)*, Morgan Kaufmann Publishers: San Mateo, CA, pp. 473–484, April 1991.

[3] B. Grosz and S. Kraus, "Collaborative plans for complex group action," *Artificial Intelligence*, vol. 86(2), pp. 269–357, 1996.

[4] R. Nair, M. Tambe, and S. Marsella, "Team formation for reformation in multi-agent domains like RoboCupRescue," in G. Kaminka, P. Lima and R. Rojas (eds.), *RoboCup 2002: Robot Soccer World Cup VI. Lecture Notes in Computer Science #2752*, Springer pp. 150-161, 2003.

[5] H. Kitano, S. Tadokoro, I. Noda, H. Matsubara, T. Takahashi, A. Shinjou and S. Shimada, "RoboCupRescue: Search and Rescue in Large-Scale Disasters as a Domain for Autonomous Agents Research," *In Proceedings of IEEE International Conference on Man, System and Cybernetics (MSC-99)*, vol. VI, pp. 739-743, Tokyo, Japan October 12-15, 1999.

[6] S. Tadokoro, H. Kitano, T. Takahashi, I. Noda, H. Matsubara, A. Shinjoh, T. Koto, I. Takeuchi, H.. Takahashi, F. Matsuno, M. Hatayama, J. Nobe and S. Shimada, "The RoboCup-Rescue Project: A Robotic Approach to the Disaster Mitigation Problem," *In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA00)*, pp. 4090-4095, April 23-28, San Francisco, USA, 2000.

[7] P. Trigo, P. Araújo, A. Remédios, C. Lopes, B. Basílio, T. Loureiro, L. Moniz and H. Coelho, "The 5Rings Team Description Paper," *In Proceedings of the RoboCup2004 Symposium, Team Description Papers*, July 4-5, Lisbon, Portugal, 2004.

[8] P. Trigo, H. Coelho. "The Multi-Team Formation Precursor of Teamwork," *In Proceedings of the EPIA05*, LNAI 3808, Springer-Verlag, 2005.

[9] P. Cohen and H. Levesque, "Teamwork," *Special. Issue on Cognitive Science and Artificial Intelligence, Noûs*, vol. 25(4), pp. 487–512, 1991.

Tokenising, Stemming and Stopword Removal on Anti-spam Filtering Domain

J. R. Méndez[1], E. L. Iglesias[1], F. Fdez-Riverola[1], F. Díaz[2], and J.M. Corchado[3]

[1] Dept. Informática, University of Vigo, Escuela Superior de Ingeniería Informática, Edificio Politécnico, Campus Universitario As Lagoas s/n, 32004, Ourense, Spain
{moncho.mendez, eva, riverola}@uvigo.es
[2] Dept. Informática, University of Valladolid, Escuela Universitaria de Informática, Plaza Santa Eulalia, 9-11, 40005, Segovia, Spain
fdiaz@infor.uva.es
[3] Dept. Informática y Automática, University of Salamanca, Plaza de la Merced s/n, 37008, Salamanca, Spain
corchado@usal.es

Abstract. Junk e-mail detection and filtering can be considered a cost-sensitive classification problem. Nevertheless, preprocessing methods and noise reduction strategies used to enhance the computational efficiency in text classification cannot be so efficient in e-mail filtering. This fact is demonstrated here where a comparative study of the use of stopword removal, stemming and different tokenising schemes is presented. The final goal is to preprocess the training e-mail corpora of several content-based techniques for spam filtering (machine approaches and case-based systems). Soundness conclusions are extracted from the experiments carried out where different scenarios are taken into consideration.

1 Introduction

Removal of non-informative words and *stemming* are document preprocessing techniques commonly used in text classification and text filtering [1, 4] to improve the accuracy of the results and to reduce the redundancy of the computation. Non-informative words are often defined by a “stopword list” which typically consists of about 400 terms including articles, prepositions, conjunctions and certain high-frequency words (some verbs, adverbs and adjectives). Stemming is a technique to reduce words to their grammatical roots so that they can be represented with a only term. A stemmer for English, for example, should identify the string "cats" (and possibly "catlike", "catty", etc.) as based on the root "cat", and "stemmer", "stemming", "stemmed" as based on "stem". Although the presence of sophisticated algorithms, the most popular stemmer is that by Porter [5].

Previously to apply stopword and stemming techniques it is necessary to carry out a lexical analysis process of the documents. Lexical analysis, also named *tokenising*, involves dividing the content of each text into strings of characters called *tokens*.

A suitable tokenising scheme can enhance meaningfully accuracy in text classification. In the most of the filtering systems, a token (also named *feature*) is considered as a string of characters delimited by punctuation symbols, blanks or tabulators.

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 449–458, 2006.

E-mail classification to spam filtering cannot be considered a traditional text classification problem. Junk e-mails have a "special" structure and content since spammers often introduce "noise" in their messages using phrases like "MONEY!!", "FREE!!!" or placing special characters into the words like "R-o-l-e?x".

The addition of seemingly nonsensical words is aimed at fooling the anti-spam filters that incorporate content-based techniques [6]. These filters examine incoming e-mail messages and calculate the probability of being spam according to intrinsic properties of them (e.g. message subject or body contents, structure, etc.) [7]. Unlike elementary content filters that simply look for specific words in the text, content-based spam filters evolve according to each user needs, analysing all e-mail to determine what words are common to appear in a user legitimate e-mail and which are not. This process is called training, and results in a highly personalized and efficient filtering system.

In order to train and test content-based filters, it is necessary to build a corpus (or use a publicly available corpus) with spam and legitimate e-mails. Anyway, corpora have to be preprocessed to extract their features. Just like text classification, the effectiveness of content-based anti-spam filters relies on the appropriate choice of the features. If the features are chosen so that they may exist both in a spam and legitimate messages, then no matter how good learning algorithm is, it will make mistakes. Therefore, the preprocessing steps of e-mail feature extraction (including the tokenising scheme) and the later selection of the most representative terms are crucial for the performance of the filter.

In this paper we show the results obtained by different well-known content-based techniques when the preprocessing of the training corpus changes. The selected models for the evaluation are three well-known machine learning approaches (Naïve Bayes [8], boosting trees [9] and Support Vector Machines [10]) and three case-based systems for spam filtering that can learn dynamically (a Cunningham *et al.* system which we call *Cunn Odds Rate* [11], its improved version named *ECUE* [12] and our SPAMHUNTING system [13]).

The rest of the paper is organized as follows: Section 2 outlines machine learning and case-based e-mail filters mentioned above, Section 3 describes the public corpus used for the empirical model evaluation and studies several issues related with message representation and feature selection. In Section 4 we present the experiments carried out and, finally, in Section 5 we expose the main conclusions reached.

2 Spam Filtering Techniques

The most popular classical filtering models are bayesian methods. Bayesian filtering is based on the principle that most of the events are conditioned. So, the probability that an event happens can be deduced from the previous appearances of that event. This technique can be used for spam filtering. If some feature is often in spam but not in legitimate e-mails, then it would be reasonable to assume that an e-mail including this feature will be probably spam. Although there are several approaches of the bayesian method, the most widely used to spam filtering is the Naïve Bayes algorithm [8].

Besides bayesian models, boosting techniques and Support Vector Machines (SVM) are also well-known machine learning (ML) techniques used in this field.

Boosting algorithms [9] are based on the use of weak learners. The main idea of boosting is to combine the hypotheses to one final hypothesis, in order to achieve higher

accuracy than the weak learner's hypothesis would have. Among different boosting algorithms developed for classification tasks we could highlight Adaboost [14].

SVMs [10] are based on representing e-mails as points in an *n*-dimensional space and finding an hyperplane that generates the largest margin between the data points in the positive class and those in the negative class. Some implementations of SVM can be found in ML environments such as Waikato Environment for Knowledge Analysis[1] (WEKA) or Yet Another Learning Environment[2] (YALE). Particularly, WEKA includes the *Sequential Minimal Optimization* (SMO) algorithm which has demonstrated a good trade-off between accuracy and speed (see [15] for details).

By other side, the most recent trends in anti-spam filtering are based on the use of Case-Based Reasoning (CBR) systems. Case-based approaches outperform classical machine learning techniques in anti-spam filtering [12] because they work well for disjoint concepts as spam whereas classical ML models try to learn a unified concept description. Another important advantage of case-based approach is the ease with which it can be updated to tackle the *concept drift* problem in the anti-spam domain [16].

Cunningham *et al.* present in [12] a successful case-based system for anti-spam filtering that can learn dynamically. The system uses a similarity retrieval algorithm based on Case Retrieval Nets (CRN) [17]. This classifier uses unanimous voting to determine whether a new e-mail is spam or not. All the returned neighbours need to be classified as spam in order to classify as spam the new message. Delany *et al.* present in [11] an evolution from this previous system called ECUE (*E-mail Classification Using Examples*) that uses a different feature selection method.

Included in the context of lazy learning, our hybrid system named SPAMHUNTING has been introduced in [13] to accurately solve the problem of spam labelling and filtering. The model follows an Instance-Based Reasoning (IBR) approach. According to this, SPAMHUNTING uses an instance memory structure as primary way of manage knowledge. The retrieval stage is carried out using a novel dynamic *k*-NN Enhanced Instance Retrieval Network (EIRN). The EIRN network facilitates the indexation of instances and the selection of those that are most similar to the new message. Similarity between two given e-mails is measured by the number of relevant features found in both messages. EIRN can quickly retrieve all stored e-mails having at least one shared feature with a target message. The reuse of similar messages is done by using a simple unanimous voting mechanism to determine whether the target case is spam or not. The revision stage is only carried out in the case of unclassified messages, where the system employs general knowledge in the form of meta-rules extracted from the e-mail headers to assign a final class.

3 Corpus Preprocessing

3.1 Analysing the Available Corpus

In order to carry out a benchmarking of the spam filter models, it is essential to have several collections of emails for training and testing purposes. Some researchers on the spam filtering domain had built their own corpus and shared it with the scientific community.

[1] WEKA is available from http://www.cs.waikato.ac.nz/ml/weka/
[2] YALE is available from http://yale.sourceforge.net

Despite privacy issues, a large number of corpus like SpamAssassin[3], Ling-Spam[4], DivMod[5], SpamBase[6] or JunkEmail[7] can be downloaded from Internet. Table 1 shows a short description of the pub.'licly available corpus focussing in the spam and legitimate ratio and the distribution form.

Table 1. Message distribution of messages belonging to the SpamAssassin corpora of emails

Corpus version	Legitime Messages		Spam Messages	
	Downloaded file	Message number	Downloaded file	Message number
2002	20021010_easy_ham.tar.bz2	2552	20021010_spam.tar.bz2	502
	20021010_hard_ham.tar.bz2	251		
2003	20030228_easy_ham.tar.bz2	2500	20030228_spam.tar.bz2	500
	20030228_easy_ham_2.tar.bz2	1400		
	20030228_hard_ham.tar.bz2	250	20030228_spam_2.tar.bz2	1397

In our work we use the SpamAssassin corpora, merging emails from 2002 and 2003 versions. It contains 9332 different messages from January 2002 up to and including December 2003. Since this corpus has not been preprocessed by the author, it can successfully be used to analyse the impact of applying different preprocessing techniques.

3.2 Message Representation

The first step in the corpora preprocessing is the e-mail representation. Each message is usually represented as a vector of weighted terms (features) much as in the vector space model in information retrieval domain [3, 4].

Feature identification can be performed by using a variety of generic lexical tools, primarily by tokenising the e-mail into words. At first glance, it seems to be a simple tokenising task guided by spaces and special symbols as word separators. However, at least the following particular cases have to be considered with care: hyphens, backslashes, punctuation marks and the case of the letters (lower and upper case).

Normally, punctuation marks are removed entirely in the process of lexical analysis. However, as mentioned above, punctuation marks and hyphenated words are among the best discriminating attributes in the spam domain because they are more common in spam messages than legitimate ones. Supported by the experiments showed in the following section and the studies carried out by other authors [18], we think that certain punctuations like exclamation, interrogation and hyphens must be part of a term. Also special terms such as *url*, *ip address* should not be splitted but treated as single terms because phrases like "*click http://xxx.xxx.com*" have a high frequency in spam messages. Finally, the case of letters is usually not important for the identification of index terms. As a result, the lexical analyser normally converts all the text to either lower or upper case.

[3] Available at http://www.spamassassin.org/publiccorpus/
[4] Available at http://www.iit.demokritos.gr/
[5] Available at http://www.divmod.org/cvs/corpus/spam/
[6] Available at http://www.ics.uci.edu/~mlearn/MLRepository.html
[7] Available at http://clg.wlv.ac.uk/projects/junk-e-mail/

When the tokenising step has been completed, stopword removal and/or stemming can be applied to identified tokens to reduce the number of representative terms of each e-mail [4].

Once carried out the lexical analysis over the whole corpus, the weight of each term in each message need to be calculated. The measure of the weight can be (*i*) binary (1 if the term occurs in the message, 0 otherwise), (*ii*) the *term frequency* (TF) representing the number of times the term occurs in the message about the total number of occurrences of terms in it, or (*iii*) TF.IDF where IDF means *Inverse Document Frequency* denoting those terms that are common across the messages of the collection [3].

Usually, the number of terms obtained by the preprocessing stage is very high and the filtering algorithms cannot work with them. Therefore, a last step in the preprocessing is carried out to reduce this number selecting the most representative features. Although some measures exist, *Information Gain* (IG) [19] has been successfully used for aggressive feature removal in several research works [7,11]. This method is based on computing the IG measure for each identified feature *t* by using the equation given in Expression (1), where $c \in \{l, s\}$ (legitimate and spam categories, respectively), and selecting those terms having the highest score.

$$IG(t) = \sum_{c \in \langle l,s \rangle} P(t \wedge c) \cdot \log \frac{P(t \wedge c)}{P(t) \cdot P(c)} \tag{1}$$

4 Evaluation

In this work we are trying to find out the impact of applying several basic preprocessing techniques over the corpus before the training and testing stages in the spam filtering domain. Moreover, we are also interested in a comparative study of the feature extraction processes. Therefore, we had compared two different tokenising schemas based on using different token separator characters: (*i*) [:blank:] and [:punt:] and (*ii*) only [:blank:]. The first one is based on using blank chars (space, tabs and carriage return marks) and punctuation marks as token separators. In the second approach, we had tried to preserve noise in spam e-mails by using only blank chars as token separators.

Besides of the above tokenising schemas, an implementation of the Porter stemming algorithm [5] and a stopword removal process have been combined and used with Naïve Bayes, SVM, Adaboost and the three previously commented CBR systems (ECUE, *Cunn Odds Rate* and SPAMHUNTING). The experiments entail measuring the performance of the analysed models combined with the proposed tokenising schemas over three different scenarios of stemming and stopword removal: (*i*) applying stopword removal and stemming analysis, (*ii*) applying stopword but without stemming and (*iii*) without applying neither stopword nor stemming.

Six well-known metrics [7] have been used in order to evaluate the performance of all the analysed models: *total cost ratio* (TCR) with three different cost values, spam *recall*, spam *precision*, percentage of correct classifications (%OK), percentage of False Positives (%FP) and percentage of False Negatives (%FN). All the experiments have been carried out using a 10-fold stratified cross-validation [20] in order to increase the confidence level of results obtained.

For our comparisons we have selected the best performance configuration of each technique varying between 100 and 2000 selected features. In order to test the *Cunn*

Odds Rate and the SPAMHUNTING models we have maintained their original feature selection methods. The first one uses an odds-ratio algorithm described in [12] in order to compute a feature selection having 30 words for representing spam class and 30 words representing legitimate category.

SPAMHUNTING feature selection is carried out in an independent form for each message. The relevant feature list of each message is computed as the minimum set containing the most frequent terms of the specified e-mail, which frequency amount is greater than a specified threshold in the range [0,1]. We are currently using 30% frequency amount as previous works have shown to be the most suitable configuration for achieving the best performance [13].

4.1 Experimental Results

Firstly, the performance of the analysed models is measured from a cost-sensitive point of view in the three predefined scenarios combined with the proposed tokenising schemas. For this purpose, we compute the TCR metric in the above mentioned different situations. TCR assumes than FP errors are λ times more costly than FN errors, where λ depends on the usage scenario (see [7] for more details). In the experiments carried out in this paper the values for λ parameter were 1, 9 and 999.

Table 2 shows the TCR scores comparative of the analysed models, tokenising schemes and preprocessing scenarios. The number of selected features used for each model is placed into square brackets. Results show that preserve noise can help to get better performance in almost all models. However, SVM model works better when noise has been removed. Case-based approaches get the highest correctly classified amount when noise is not removed and stopword removal is used (Scenario 2). Table 2 also shows that noise removal is sometimes useful to get high security (TCR λ=999).

Table 2. TCR scores over 10 stratified fold-cross validation

		Scenario 1		Scenario 2		Scenario 3	
Tokenising separator chars		[:blank:] [:punct:]	[:blank:]	[:blank:] [:punct:]	[:blank:]	[:blank:] [:punct:]	[:blank:]
Naïve Bayes [1000]	TCR λ=1	2.612	2.985	2.609	2.863	2.943	3.174
	TCR λ=9	0.501	0.568	0.501	0.508	0.566	0.560
	TCR λ=999	0.005	0.006	0.005	0.005	0.006	0.005
Adaboost [700]	TCR λ=1	4.550	4.659	4.403	5.003	5.153	4.854
	TCR λ=9	1.640	1.689	1.578	1.618	1.494	1.432
	TCR λ=999	0.021	0.022	0.020	0.019	0.017	0.017
SVM [2000]	TCR λ=1	27.771	19.011	20.677	17.990	27.423	23.120
	TCR λ=9	6.443	3.353	4.257	3.559	6.088	4.920
	TCR λ=999	0.068	0.033	0.043	0.037	0.064	0.050
Cunn Odds Rate [60]	TCR λ=1	1.308	1.174	1.296	1.364	1.305	1.247
	TCR λ=9	1.296	1.153	1.285	1.351	1.294	1.242
	TCR λ=999	1.082	0.779	1.074	1.130	1.083	1.140
ECUE [700]	TCR λ=1	4.441	4.562	4.699	6.020	4.226	5.622
	TCR λ=9	2.432	3.071	2.496	2.846	2.265	2.498
	TCR λ=999	0.059	0.099	0.052	0.046	0.042	0.036
SPAM HUNTING [-]	TCR λ=1	3.903	6.024	3.958	7.498	4.218	4.594
	TCR λ=9	3.493	4.823	3.384	5.331	2.904	2.982
	TCR λ=999	2.437	1.545	1.681	0.874	0.499	0.081

From another point of view, Table 3 shows the recall and precision scores obtained for each considered experimental situation. Analysing recall results, and keeping in mind maximizing the highest correctly classified amount, we can realize that classical ML models should not use stopword removal nor stemming (Scenario 3). However, if the same goal is required, CBR/IBR approaches should use stopword removal but not stemming (Scenario 2). Precision scores clearly shows that Naïve Bayes, SVM and *Cunn Odds Rate* get the highest security level (minimize FP errors) when stemming and stopword removal are not used. Nevertheless, these preprocessing techniques should be applied when Adaboost, ECUE or SPAMHUNTING are used.

Table 4 shows the amount of correct classifications, false positives and false negatives belonging to the experimental work with the six analysed models over the

Table 3. TCR scores over 10 fold-cross validation

		Scenario 1		Scenario 2		Scenario 3	
Tokenising separator chars		[:blank:] [:punct:]	[:blank:]	[:blank:] [:punct:]	[:blank:]	[:blank:] [:punct:]	[:blank:]
Naïve Bayes [1000]	Recall	0.817	0.842	0.837	0.853	0.818	0.869
	Precision	0.801	0.824	0.801	0.807	0.823	0.825
Adaboost [700]	Recall	0.832	0.836	0.827	0.854	0.865	0.858
	Precision	0.939	0.943	0.938	0.939	0.934	0.928
SVM [2000]	Recall	0.976	0.978	0.976	0.977	0.979	0.916
	Precision	0.979	0.968	0.975	0.964	0.983	0.978
Cunn Odds Rate [60]	Recall	0.235	0.150	0.233	0.266	0.228	0.198
	Precision	0.997	0.986	0.996	0.997	0.996	0.998
ECUE [700]	Recall	0.798	0.793	0.787	0.857	0.805	0.846
	Precision	0.969	0.981	0.969	0.971	0.966	0.967
SPAM HUNTING [-]	Recall	0.748	0.837	0.776	0.871	0.753	0.795
	Precision	0.993	0.992	0.991	0.991	0.979	0.978

Table 4. Mean value of correct classifications, FPs and FNs with 10 fold-cross validation

		Scenario 1		Scenario 2		Scenario 3	
Tokenising separator chars		[:blank:] [:punct:]	[:blank:]	[:blank:] [:punct:]	[:blank:]	[:blank:] [:punct:]	[:blank:]
Naïve Bayes [1000]	OK	841.2	852.5	841.4	849.6	851.5	857.9
	False Positives	48.5	43	48.5	48.6	42.8	44
	False Negatives	43.5	37.7	43.3	35	38.9	31.3
Adaboost [700]	OK	880.3	881.9	878.8	885.1	886.4	883.4
	False Positives	13	12.2	13.1	13.4	14.6	16
	False Negatives	39.9	39.1	41.3	34.7	32.2	33.8
SVM [2000]	OK	922.7	920.2	921.5	919.1	924.1	922.2
	False Positives	4.9	7.8	6	8.7	4.1	5.3
	False Negatives	5.6	5.2	5.7	5.4	5	5.7
Cunn Odds Rate [60]	OK	750.8	730.4	749.2	758.3	750.4	742.1
	False Positives	0.2	0.5	0.2	0.2	0.2	0.1
	False Negatives	182.2	202.3	183.8	174.7	182.6	191
ECUE [700]	OK	878.9	880.2	880.6	893	875.8	889.8
	False Positives	6.2	3.6	6.2	6.1	6.7	6.8
	False Negatives	48.1	49.4	46.4	31.4	50.7	36.6
SPAM HUNTING [-]	OK	871.9	892.9	872.7	900.8	875.8	880.3
	False Positives	1.2	1.5	1.6	1.8	4	4.2
	False Negatives	60.1	38.8	58.9	30.6	53.4	48.7

defined scenarios and tokenising schemas. Analysing Table 4 we can realize that removing noise on tokenising is advisable for achieving the highest amount of correctly classified messages with Naïve Bayes and Case-based models. Adaboost and SVM get the highest amount of correctly classified messages when noise is not removed.

Table 4 also shows that the minimum amount of FP errors using ECUE, *Cunn Odds Rate* or Adaboost models is achieved when noise is removed in the tokenising stage. In the other hand, Naïve Bayes, SVM and SPAMHUNTING models work with a highest level of security when noise is not removed.

5 Conclusions and Further Work

In this work we perform a deep analysis of tokenising, stopword removal and stemming techniques. We had chosen a representative set of spam filtering models (including Naïve Bayes, Adaboost, SVM and three case-based systems) in order to see the performance when the context of tokenising, stopword removal and stemming changes.

Table 5. Recommended tokenising, stopword removal and stemming configurations

		Scenario 1		**Scenario 2**		**Scenario 3**	
Tokenising separator chars		[:blank:] [:punct:]	[:blank:]	[:blank:] [:punct:]	[:blank:]	[:blank:] [:punct:]	[:blank:]
Naïve Bayes	Maximize hits						√
	Minimize FP					√	
Adaboost	Maximize hits					√	
	Minimize FP		√				
SVM	Maximize hits					√	
	Minimize FP					√	
Cunn Odds Rate	Maximize hits				√		
	Minimize FP						√
ECUE	Maximize hits				√		
	Minimize FP		√				
SPAM HUNTING	Maximize hits				√		
	Minimize FP	√					

Table 5 summarizes the recommended preprocessing context for achieving two different goals: (*i*) get the maximum amount of correctly classified messages and (*ii*) get highest security level. As we can see from results, stopword removal is useful for achieving the highest amount of correctly classified messages when case-based reasoning systems are used. This fact represents a weakness of the IG and Odds Ratio feature selection methods as they select some unhelpful stopwords. This happens because the presence of stopwords hamper the selection of features which are really able to differenciate between spam and legitimate messages.

However, although stopwords are semantically void, some of them (such as "however" or "therefore") are unusual in spam messages and should not be filtered. In this way we can highlight the unexpectedly performance of SVM when stemming and/or stopword removal is not used as preprocessing techniques. This happens because SVM transforms the input space into a new linearly separable one by removing unhelpful features but keeping useful stopwords. SVM model has its own implicit feature selection method.

The tokenising schema constitutes a relevant issue in the context of spam filtering because, selecting the right choice contributes to improve performance. Keeping in mind the closely relationship between tokenising and classifier, we think that new tokenising schemas (able to remove or preserve noise data) should be proposed, analysed and studied in order to improve the performance of the existing models.

Besides tokenising, stemming and stopword removal issues, we are also interested in analysing the impact of several well-known feature selection methods commonly used in the current domain. We think that modern and classical classifier models are good enough for achieving accuracy results. We could try to improve them or probe new spam filtering models. However, instead this, we are wondering what would happen if we feed current models with an accuracy representation of the training and test messages. Thinking in future work, new and more accuracy techniques of representing the input messages can boost the spam filtering world.

References

1. van Rijsbergen. C.J.: Information Retrieval, 2nd Ed. Butterworths, (1979)
2. Salton, G.: Automatic text processing: The transformation, analysis, and retrieval of information by computer. Addison-Wesley, (1989)
3. Salton, G., McGill, M.: Introduction to Modern Information Retrieval, McGraw-Hill, (1983)
4. Baeza-Yates, R., Ribeiro-Neto, B.: Modern Information Retrieval. Addison Wesley, (1999)
5. Porter, M.F.: An algorithm for suffix stripping. In Readings in Information Retrieval, (1997) 313-316
6. Oard, D.W.: The state of the art in text filtering. User Modeling and User-Adapted Interaction, Vol.7, (1997) 141–178
7. Androutsopoulos, I., Paliouras, G., Michelakis, E.: Learning to Filter Unsolicited Commercial E-Mail. Technical Report 2004/2, NCSR "Demokritos", (2004)
8. Sahami, M., Dumais, S., Heckerman, D., Horvitz, E.: A Bayesian approach to filtering junk e-mail. In Learning for Text Categorization – Papers from the AAAI Workshop, Technical Report WS-98-05, (1998) 55–62
9. Carreras, X., Màrquez, L.: Boosting trees for anti-spam e-mail filtering. Proc. of the 4th International Conference on Recent Advances in Natural Language Processing, (2001) 58–64
10. Vapnik, V.: The Nature of Statistical Learning Theory. 2nd Ed. Statistics for Engineering and Information Science, (1999)
11. Delany, S.J., Cunningham P., Coyle L.: An Assessment of Case-base Reasoning for Spam Filtering. Proc. of Fifteenth Irish Conference on Artificial Intelligence and Cognitive Science: AICS-04, (2004) 9–18
12. Cunningham, P., Nowlan, N., Delany, S.J., Haahr, M.: A Case-Based Approach to Spam Filtering that Can Track Concept Drift. Proc. of the ICCBR'03 Workshop on Long-Lived CBR Systems, (2003)
13. Fdez-Riverola, F., Méndez, J.R., Iglesias, E.L., Díaz, F. Representación Flexible de e-mails para la construcción de filtros antispam: un caso práctico. Proc. of the I Congreso Español de Informática CEDI'05 (2005) 109-116

14. Schapire, R.E., Singer, Y.: BoosTexter: a boosting-based system for text categorization. Machine Learning, Vol. 39 (2/3). (2000) 135–168
15. Platt, J.: Fast training of Support Vector Machines using Sequential Minimal Optimization. In Sholkopf, B., Burges, C., Smola, A. (eds.). Advances in Kernel Methods – Support Vector Learning, MIT Press, (1999) 185–208
16. Widmer, G., Kubat, M.: Learning in the presence of concept drift and hidden contexts. Machine Learning, Vol. 23 (1). (1996) 69–101
17. Lenz, M., Auriol, E., Manago, M.: Diagnosis and Decision Support. Case-Based Reasoning Technology. Lecture Notes in Artificial Intelligence, Vol. 1400, (1998) 51–90
18. Le, Z., Tian-shun, Y.: Filtering Junk Mail with A Maximum Entropy Model. Proc. of the ICCPOL2003, (2003) 446–453
19. Yang, Y., Pedersen, J.O.: A comparative study on feature selection in text categorization. Proc. of the Fourteenth International Conference on Machine Learning: ICML-97, (1997) 412–420
20. Kohavi, R.: A study of cross-validation and bootstrap for accuracy estimation and model selection. Proceedings of the 14th International Joint Conference on Artificial Intelligence: IJCAI-95, (1995) 1137–1143

Toward a Motivated BDI Agent Using Attributes Embedded in Mental States

José Cascalho[1], Luis Antunes[1], Milton Corrêa[2], and Helder Coelho[1]

[1] Departamento de Informática
Faculdade de Ciências da Universidade de Lisboa
Bloco C6, Piso 3, Campo Grande
1749-016 Lisboa, Portugal
{jmc@notes.uac.pt, xarax@di.fc.ul.pt, hcoelho@di.fc.ul.pt}
[2] Coordenação de Informática
Laboratório Nacional de Cálculo Científico
Av. Gertúlio Vargas, 333
Petrópolis, RJ-Brasil
mcorrea@lncc.br

Abstract. In this paper we discuss how to apply the Mental State Framework to model an agent that uses different motivations to make decisions. We employ several mental states attributes to characterize the mechanism of decision and we add a set of motivation-derived desires to the top of a BDI-like architecture. We show that an agent can change her behavior if motives behind the decisions change. Finally we explore reasons behind reasons, arguing that we can expect to identify different agent types. Endowed with such operational extensions, the Mental State Framework can provide a tool with which to program flexible agents in a principled way.

1 Introduction

Let us suppose we need to buy plane tickets and we do it with the help of software agents. We decide to buy a ticket to a location where we have a meeting in a specific date, to a maximum price of 350 Euro and we want to go on a direct flight. If we search for a ticket using an interface belonging to a specific software house, the agents, indirectly by the users, are ranked by its performance. It is in the interest of the software house to get the best performance as possible.

Suppose agents have a measure of importance that evaluates the cost/benefit ratio toward decisions related with agents' goals, i.e., the goal to return with an answer (positive or negative) and the goal to do the search, and also have a measure of urgency which evaluates the time left for agents to fulfill that goal. As the deadline approaches, the urgency for returning will surpass the importance of searching even if the goal is not yet reached.

If an agent returns too soon, he may risk to loose for another agent returning later with a better offer. On the other hand, if an agent returns too late, the user might already have decided to choose another agent's proposal. What should be the best strategy for a software house to be successful in search?

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 459–469, 2006.

Probably the best strategy for a software house is to have different *types* of agents, some persistent that would carry a goal until a deadline, and others that would be satisfied with a result close enough to the restrictions proposed. We are interested in exploring these adapting agents' characteristics, i.e. changing some of the attribute values, which implies a different way of reaching a goal and/or choosing a strategy to accomplish it.

In this paper we use the Mental State Framework (MSF) to setup an agent which uses different motivations to make decisions in a *case-study* example presented in the next chapters. In the MSF an agent's mind is a set of mental states of different types (a set of different mental states entities). Each mental state type is composed by a set of basic components that explain the relationships among other mental states and their dynamics behind agent's behaviors. The components are organized in three groups: the nucleus or core attributes along with a set of criteria for dissatisfaction, uncertainty, intensity, insistence and intensity; the laws, containing the causal relationships among mental states and the controls which impose a set of constraints for the mental state activity through the triggering and filtering mechanisms [1][2]. Under constraints imposed by the laws and controls, an agent's *mind* consists of a set of *rules*. These rules form the agent's deliberation process.

In the next chapter, we will describe succinctly the MSF. Then, we will explain how to create a BDI-like agent in a case-study example. This agent is then modified in order to add a motivational component. Finally, we discuss the example and present the conclusions.

2 The Mental State Framework

In the MSF we represent all the components and their properties in a table (see table 1). The table identifies the three mental state ($\mathcal{MS}$) types characterizing a BDI-like architecture: *beliefs* (B) corresponding to a state in the world; *desires* (D) which can be goal-driven or motivational-driven, the former corresponding to usual notion of goal, i.e., a goal is a state the agent wishes to bring about, and the latter corresponding to the notion of motivation as described in [3]; and finally *intentions* (I) as commitments to a future state of affairs (they have the role of *pro-attitudes* [4])[1].

The marks in the different rows identify the different attributes, laws and controls used by each mental state type. For example, in a rule created for a soccer agent:

$$I(SearchBall)_{C5} \leftarrow_T D(ToSeeBall)_{C1} + B(KnowBallPosition)_{\neg C2} \quad \text{(RX)}$$

we use the law $L7$. The tag T in the arrow stresses that the rule is a triggering rule. The controls $C1$ and $C2$ attached to the $\mathcal{MS}$ *sources*, the desire *ToSeeBall* and the belief *KnowBallPosition* respectively, establish the conditions for the law triggering; only when $C1$ and $\neg C2$ return a *TRUE* value, the $\mathcal{MS}$ *target*

[1] In this paper, we assume that each intention has attached a pre-planned sequence of actions.

Table 1. *Table* used in the experiment. The mental state types are the B,D,I.

	B	D	I
Dissatisfaction (d)		x	
Importance (i)			x
Uncertainty (u)	x		
Intensity (n)			x
Urgency (g)		x	x
Insistence (s)		x	x
L1 B $\leftarrow$	x		
L2 B $\leftarrow B+$	x	x	
L3 D $\leftarrow$		x	
L4 D $\leftarrow B+$		x	
L5 I $\leftarrow$			x
L6 I $\leftarrow I+$			x
L7 I $\leftarrow B+$		x	x
C1 Triggering		x	x
C2 Triggering	x		
C4 Propagate	x	x	x
C5 Propagate	x	x	x
C8 Filtering	x		x
C9 Filtering			x
C10 Filtering			x
C11 Filtering			x

$I(SearchBall)$ becomes active[2](see table 2 for a description about the triggering conditions).

Metaphorically we can say that laws define syntax for rules that can be built according to a set of constraints, and controls define *semantics* that explain the circumstances under which rules take effect.

A triggering rule activates a target(s) if and only if all the sources are active and controls associated to each one of the $\mathcal{MS}$ return the value $TRUE$. This means that we use the triggering rule type to characterize mechanisms behind all the BDI-like architecture, i.e., selecting which desires will be transformed into goals/intentions supported by beliefs.

Looking at controls in table 2, we notice that in the rule example above, values of urgency and insistence are propagated from desire *ToSeeBall* to intention *SearchBall* by control $C5$ (in this case, the $\mathcal{MS}_{parent}$ in the table 2 corresponds to mental state $D(ToSeeBall)$). We call this type of control, a *propagation control* which in the example is attached to a triggering rule.

Along with propagation and triggering controls, we also define filter/interrupt controls with two different states: *Normal State* and *Urgent State* (see figure 1). Again in table 2, we show that the value returned from filter evaluation depends on filter state, e.g. for control $C8$, the value returned is equal to $\mathcal{MS}_{parent}$ importance or to $\mathcal{MS}_{parent}$ urgency as the filter state is 'normal' or 'urgent'

[2] The control $\neg C2$ returning $TRUE$ means the control $C2$ returns $FALSE$.

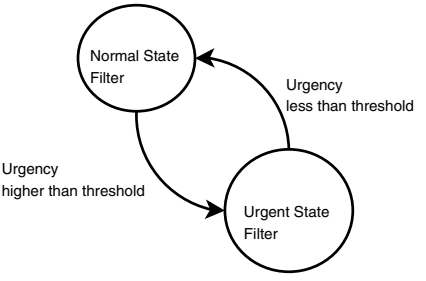


Fig. 1. The filter states

Table 2. The different table controls and the evaluation core attributes: d is the dissatisfaction value, u is the uncertainty value, i is the importance value, g corresponds to the urgency, s corresponds to intensity and finally n corresponds to the insistence

Control name	*Control type*	*Condition/Function*
C1	Triggering	$TRUE$ iff $d(\mathcal{MS}) > \epsilon_d$
C2	Triggering	$TRUE$ iff $u(\mathcal{MS}) < \epsilon_u$
C3	Propagate	$n(\mathcal{MS}) = d(\mathcal{MS}_{parent})$
C4	Propagate	$g(\mathcal{MS}) = 1 - d(\mathcal{MS}_{parent})$
C5	Propagate	$g(\mathcal{MS}) = g(\mathcal{MS}_{parent}) \wedge n(\mathcal{MS}) = n(\mathcal{MS}_{parent})$
C8	Filtering	$(value = i(\mathcal{MS}) \wedge state = normal) \vee$ $(value = g(\mathcal{MS}) \wedge state = urgent)$
C9	Filtering	$(value = i(\mathcal{MS}) * n(\mathcal{MS}) \wedge state = normal) \vee$ $(value = g(\mathcal{MS}) \wedge state = urgent)$
C10	Filtering	$(value = i(\mathcal{MS}) + s * i(\mathcal{MS}) \wedge state = normal) \vee$ $(value = g(\mathcal{MS}) \wedge state = urgent)$
C11	Filtering	$(value' = i(\mathcal{MS}) * n(\mathcal{MS}) \wedge$ $value = value' + s * value' \wedge state = normal) \vee$ $(value = g(\mathcal{MS}) \wedge state = urgent)$

respectively. In the next section, we will see an application for this kind of controls applied to filtering rules.

To sum up, an agent's mind is a set of triggering (with propagation controls) and filtering rules; target mental states become active or inactive as the rules to which they are connected to, trigger or not trigger, respectively; the state of activity of an agent's mind, i.e. the set of mental states in active state, will be dependent on values of core attributes (they are part of the internal state of each mental state); these attributes have a value in the interval $[0, 1]$, attributes uncertainty and dissatisfaction are applied to belief and desire $\mathcal{MS}$ types while attributes importance and urgency are related to intention $\mathcal{MS}$ type.

3 Managing Motivations in MSF

We now use the example we introduced in the first section to show how to tackle the motivation issue. We start by formalizing the problem by following

a BDI-like model. Then we introduce motives as new higher-level desires and restate the problem in this new setting. This prototypical example will illustrate general mechanisms for motivation and decision we propose to extend the BDI-like model.

As we have seen previously, we have an agent who has the goal to find tickets (make a reservation) to a plane travel under a certain set of conditions. We show the conditions for two different desirable travels in the tables 3 and 4.

Table 3. Conditions CND1 for a trip from Lisbon to Paris

Deadline	The value of deadline is t_4. After this value it is impossible to know if the result of the search will be accepted by the user.
Maximum price	230 Euro.
Date Fit arrival	The arrival date must correspond exactly to the proposed date.
Date Fit departure	Although the exact date is preferred, the departure can take place until 3 days after the date specified.

An agent can satisfy her goals in two different ways, a *fully-satisfied situation* (fit), where all constraints are satisfied, and a *partial-satisfied situation* (ps), where constraints are not satisfied in optimal conditions, e.g., having a date fit departure 2 days after ideal date in *CND1*. Finally we call *dissatisfied situation* (ds) those situations in which conditions doesn't fit at all, e.g. the price is higher than 750 Euro in *CND2*,

Moreover, we divide time instants $\{t1, t2, t3, \ldots\}$ into *before deadline* (bd) and *after deadline* (adl). For example, in conditions imposed in CND1, all time instants from t1 to t4 are *bd*.

Table 4. Conditions CND2 for a trip from Lisbon to New York

Deadline	The value of deadline is t_4. After this value it is impossible to know if the result of the search will be accepted by the user.
Maximum price	750 Euro.
Date Fit arrival	The arrival date must correspond exactly to the proposed date.
Date Fit departure	Although the exact date is preferred, the departure can take place until 3 days after the date specified.

Table 5 shows possible outcomes for a search executed by an agent in trips *Lx-Pr* and *Lx-NY* from time tag t_1 to t_6. In the table 5, the 'price' row denotes different travel prices found by agent, and in 'time' row we identify date constraints for each ticket, i.e. *fit* meaning that date constraints are satisfied, *ps* meaning that departure date is not the ideal, and finally *ds* meaning that dates are not acceptable by the agent. In the next section we will use these values to discuss the behavior of a BDI-like versus a Motivated BDI agent.

Table 5. The simulation of an agent's search outcome in the *Internet*. The tag 'fit' means that all the date conditions are satisfied, the tag 'ps' means that the departure date will not fit but still in the space of partial-satisfied situation and 'ds' means that none of the dates will fit (dissatisfied situation).

Travel		*t1*	*t2*	*t3*	*t4*	*t5*	*t6*
Lx-Pr	Price	250	230	150	200	199	50
	Time	fit	fit	ps	fit	ds	fit
Lx-NY	Price	1100	800	1500	750	1200	1000
	Time	fit	fit	ps	ps	ds	fit

3.1 The BDI-Like Agent

A BDI-like agent will have two goals, one for searching a ticket that fits user imposed constraints and another for returning with an answer (positive or negative) to user. We assume an agent will pursue one of the goals in each moment and when both are *desired* at the same time, she will have to choose one of them because they are mutually exclusive.

In this scenario, we *define* two triggering rules, i.e., rules responsible for activation of intention mental states (rules R1 and R2) and a filtering rule that will manage choice (rule R3).

$$I(Search)_{C5} \leftarrow_T D(GetTheTickets)_{C1} \quad \text{(R1)}$$

$$I(Return)_{C5} \leftarrow_T D(Return)_{C1} \quad \text{(R2)}$$

Both rules above use a law of type $L4$ which triggers an intention if and only if a desire is dissatisfied, i.e., if the evaluation of control $C1$ attached to desire returns $TRUE$.

In rule R1, desire *GetTheTickets* is dissatisfied while the price of all tickets found are above a maximum or the corresponding dates are *ds*.

In rule R2, desire *Return* will always be triggering until agent returns back to user.

In both rules, control C5 will propagate values of urgency and insistence from desire *Return* and *GetTheTickets* to intentions *Return* and *Search* respectively.

In some circumstances these two rules trigger simultaneously, e.g., an agent in time instant $t4$ without any ticket will activate rule R1 and rule R2. Hence, a filter is required to evaluate which intention should become active or maintain its activity in the presence of the other. This is done by comparing values returned from control $C8$, as expressed in filtering rule $R3$ below [3].

$$I(Search|Return) \leftarrow_F I(Search)_{C8} + I(Return)_{C8} \quad \text{(R3)}$$

When the filter is in 'normal state', rule $R3$ selects for activation the intention with the highest value of importance (see table 2). In the table 6 we show different

[3] The symbol '|' in left side of the law means that only one of the intentions will persist in active state.

values of *importance* for all different agent's time constraint situations. This is a simplification of a cost/benefits calculus and we assume these values are hardwired to the system.

Finally, for this rule it is also fundamental to characterize the value of urgency (see row for control $C8$ in table 2). For the urgency of *GetTheTickets* desire, we ascribe a constant value below a threshold maintaining the filter in 'normal state' (in the example, equal to 0.7). But the urgency for *Return* desire will grow as time approaches deadline in a way that it will have the maximum value, i.e., equal to 1.0, at deadline instant $t4$.

Table 6. The table of importance for the different situations (fit,ps and ds)

Situations	Goals' Importance	
	Search	Return
fit	0.4	0.6
ps	0.6	0.4
ds	0.9	0.1

With this rules set, agent will return from a search for a *Lx-Pr* ticket at time instant $t2$ and from a search for a ticket *Lx-NY* at time $t4$. In the former case, agent will stop the searching behavior at first fit situation instance. At this time step, importance of *Return* will be 0.6 and so rule R3 will 'decide' for this intention. In the latter case, agent will stop searching for a ticket because urgency of *Return*, at time instant $t4$, achieves a maximum value, changing the state of the filter to 'urgency state', in which case rule R3 will 'decide' for activation of *Return* intention, selecting the mental state with the highest urgency.

Now if we look into table 5, we notice that for the ticket *Lx-Pr*, agent could improve her result (getting a lower ticket price) if she insisted on searching until time instant $t4$.

We could have changed the importance table to extend the searching for ticket to instant $t4$ (changing the *fit* row to a 0.6 value for Search importance and 0.4 value for Return importance). But this change would not have improved control over global agent's behavior. Moreover, we argue that replacing table values is not a guarantee to improve agent's performance. As a matter of fact, an agent, by contrast, could have strong reasons to return with a result to user as quicker as possible [4]. What we need is to find ways to improve agents' control by exploring those reasons and we propose to do it with motivated agents.

3.2 The Motivation-Based BDI Agent

We now embed motivation in a desire mental state. To extend the model, we create a triggering rule using law $L3$ from the table 1 where a desire (as a goal) at the left side of the rule is 'generated' by a desire (as a motive) at the right side of the rule. As shown in table 2, this kind of rules will trigger if and only if

[4] At anytime another agent could have returned in time instant $t2$ with a ticket with the same price in a *ps* condition, and user could have accepted it.

motive's dissatisfaction is higher than a threshold ϵ_d. So, in this rule, motives will be responsible for the activation of goals. This is similar to what Sloman proposes when he discusses the role of motives in his proposed agent's architecture [5].

However, we also need to make a few changes in some of the previous rules. In rule $R1$, the desire now becomes active independently of tickets be found or not, i.e., the value for dissatisfaction will be permanently higher than the threshold and with the desire in active state, the rule will always trigger (this means that the desire will loose part of its 'autonomy' to motives as we will see in description below).

$$I(Search)_{C5} \leftarrow_T D(SearchTickets)_{C1} \quad \text{(R1')}$$

Table 7. The motives

NeedToSearchUntilFitSol	Agents will insist on searching for tickets until all constraints are satisfied.
PersistUntilFindIdenticalSituation	This motive uses data extracted from other 'similar' search cases. Evaluating that data will improve agents' confidence wrt. the finding of results at least as good as previous ones.
ReevaluateRiskInDeadLine	This motive will reduce pressure over agents to quickly return with an answer. Agents have access to data about previous cases where a response time equal to $t5$ or even $t6$ was accepted by the user.

We also replace a control from C8 to C10 in intention *Search* used by the rule $R3$ to contemplate the value of insistence as described in table 2.

$$I(Search|Return) \leftarrow_F I(Search)_{C10} + I(Return)_{C8} \quad \text{(R3')}$$

Finally, we add new rules related to motivators. We identify two motives for desire *SearchTickets* and another motive for desire *Return* (see table 7).

$$D(SearchTickets) \leftarrow_T D(NeedToSearchUntilFitSol)_{C1} \quad \text{(R4)}$$

$$D(SearchTickets)_{C3} \leftarrow_T D(PersistUntilFindIdenticalSituation)_{C1} \quad \text{(R5)}$$

A value of insistence is propagated to desire *SearchTickets* by receiving the value of the motive *PersistUntilFindIndenticalSituation* dissatisfaction, as described in the table 2 (the value of insistence equals the value of dissatisfaction of the motive).

If an agent has been successful in previous ticket-buying interactions and if she kept information about those experiences, e.g., the mean value of Lisbon to Paris tickets price bought in previous interactions, she could use this information to formulate an expectation about the price to destination.

Let us suppose an agent expects a value for travel Lx-Pr around 200 Euro. *PersistUntilFindIdenticalSituation* (rule R5) will trigger *SearchTickets* until time instance $t4$. This new *reason* implies a new value for insistence inside intention *Search*[5].

[5] Insistence of *Search* intention will be equal to insistence of *SearchTickets* desire by the influence of control C5 in the rule *R1'* which in turn will be equal to dissatisfaction of *PersisUntilFindIndenticalSituation* motive by control C3 in rule *R5*.

If at $t2$ the insistence value is 0.6 then the final value returned from the filter in the rule $R3'$ in I(Search) will be $0.4 + 0.4 * 0.6 = 0.64$ which is higher than the 0.6 value for the I(Return). So the agent will persist searching until $t4$. In this same instant, the motive *PersistUntilFindIdenticalSituation* is satisfied and then rule *R5* stops triggering, allowing the agent to select the intention *Return*.

Considerer now the case where the agent expects a value lower than 200 Euro for traveling to Lx-Pr. In this case, motive *PersistUntilFindIdenticalSituation* would activate rule R5 beyond time step $t4$. It is now the motive for desire *Return* which decides about the returning point at $t6$. But first we have to consider the fact that motive *ReevaluateRiskInDeadLine* dissatisfaction decreases as $t6$ approaches.

$$D(Return)_{C4} \leftarrow_T D(ReevaluateRiskInDeadline)_{C1} \quad \text{(R6)}$$

With rule R6, the value of urgency of desire *Return* increases as motive's dissatisfaction decreases (see table 2). This implies that the filtering rule *R3'* will change to 'urgent' state at $t6$. Finally all these imply that at $t6$, rule *R3'* activates intention *Return* because urgencies of the two intentions are now compared. By 'lucky' and a great deal of persistence, the agent returns with a ticket that cost 50 Euro!

4 Conclusions

We add to the top of a BDI paradigm a motivational level which improves the ability to tune agents in order to satisfy users' needs. Jennings et al. in [6] argue that motives "(...) could offer a substantially higher level of control than is currently available. This will become more important for agent systems that need to function in an autonomous yet persistent manner while responding to changing circumstances.". This is exactly what we discuss in this paper, by using a simple example of an agent trying to satisfy a goal on behalf of an user.

Adding new desires *on the top* of other desires came natural in the MSF framework, where mental design of an agent is built by the interconnection of different mental states entities. We are capable of playing with elements at our disposal and we can change agents response to the events. We argue that this is a starting point to identify patterns of response, which will conduct us to agent's types definition.

Several work has been made around motivation modeling. In [7] motivation is viewed as a *driving force*. Our motives also *explain* goals reasons and, as proposed by Norman and Long, has an intrinsic value that will *trigger* goals in which the motive's 'strength' is above a certain threshold. Differently from these proposals we incorporate the 'motivator' inside the framework structure.

In [8] motivations served as an encapsulation of all information needed to satisfy requirements, and compete for resources usage by a resource controller. We used filtering mechanism to deal with competing motivations. More recently, Coddington [3] shows how motivations can affect the way an agent builds a plan. In the final discussion, it is argued about the necessity to treat motivations

differently, i.e., assuming for them different values of importance. We tackle this issue when we propose that motives change the different dimensions of agents' goals, affecting the decision process that conducts agent's choice, e.g., we prioritize motives when agent changes goal selection mechanism depending on a filtering threshold.

In this work, our contribution relates to the way motives change different parameters attached to intention mental states. We could identify several different influences based on contextual information as studied in the presented example. The different attributes implied in evaluation, forced us to make choices about possible results based on a very coarse analysis of possible agent *types*. In future work, we intend to build more examples where we explore in depth implications of this first result. As a driving force, the motives affect the way the choices are made by an agent. In our approach, motives modify the way rules trigger by changing some of the attributes responsible for the triggering process. They act upon this rules, modifying the circumstances under which they trigger. A different proposal is presented in [3] where actions may support or undermine motives which conduct an agent to select the fittest strategy in respect to his motives.

The approach of using a motivator evaluation resembles worth-oriented models [9] [10]. In those models, worth is a benefit and costs are associated to the execution of (joint) intentions. We somehow extend this model, allowing to multi-dimensional evaluation of the worths. Although the evaluation becomes more complex, we argue that the flexibility in the agent's performance compensates this increase in complexity.

References

1. Corrêa, M., Coelho, H.: Collective mental states in an extended mental states framework. In: International Conference on Collective Intentionality IV, Certosa di Pontignano. (2004) 13–15
2. Corrêa, M., Coelho, H.: From mental states and architectures to agents' programming. In Coelho, H., ed.: Proceedings of the Sixth Iberoamerican Confrence in Artificial Intelligence. Volume 1484 of Lectures Notes in Artificial Intelligence., Springer-Verlag (1998) 64–75
3. Coddington, A., Luck, M.: A motivation-based planning and execution framework. International Journal on Artificial Intelligence Tools **13** (2004) 5–25
4. Wooldridge, M.: Reasoning about Rational Agents. The MIT Press (2000)
5. Sloman, A. In: Motive Mechanisms Emotions, M.A. Boden (ed), The Philososphy of Artificial Intelligence. Oxford University Press (1990) 231–247
6. Cohn, A.C., Jennings, N.R.: Interaction, planning and motivation. In: Cognitive systems: Information processing meets brain science. Springer (2005) 163–188
7. Norman, T.J., Long, D.: Goal creation in motivated agents. In Wooldridge, M., Jennings, N.R., eds.: Intelligent Agents: Theories, Architectures, and Languages (LNAI Volume 890). Springer-Verlag: Heidelberg, Germany (1995) 277–290
8. Lioncourt, S.W., Luck, M.: Motivated agent behaviour and requirements applied to virtual emergencies. In Marik, V., Stepankova, O., Krautwurmova, H., Luck, M., eds.: Multi-Agent Systems and Applications II, Lecture Notes in Artificial Intelligence. Volume 2322. Springer (2002) 43–60

9. Luis Antunes, J.a.F., Coelho, H.: Improving choice mechanisms with the bvg architecture. In L'esperance, Y., Castelfranchi, C., eds.: Intelligent Agents VII, LNAI. Springer (2000) 290–304
10. Munroe, S.J., Luck, M., d'Inverno, M.: Towards motivation-based decisions for worth goals. In V. Marik, J.M., Pechoucek, M., eds.: Multi-Agent Systems and Applications III, Lecture Notes in Artificial Intelligence. Volume 2691. Springer (2003) 17–28

Training and Analysis of Mobile Robot Behaviour Through System Identification

Roberto Iglesias[1], Ulrich Nehmzow[2], Theocharis Kyriacou[2], and Steve Billings[3]

[1]Electronics and Computer Science, University of Santiago de Compostela, Spain
[2]Dept. of Computer Science, University of Essex, UK
[3]Dept. of Automatic Control and Systems Engineering, University of Sheffield, UK

Abstract. In this paper we describe a new procedure to obtain the control code for a mobile robot, based on system identification: Initially, the robot is controlled by a human operator, who manually guides it through a desired sensor-motor task. The robot's motion is then "identified" using the NARMAX system identification technique. The resulting transparent model can subsequently be used to control the movement of the robot.

Using a transparent mathematical model for robot control furthermore has the advantage that the robot's motion can be analysed and characterised quantitatively, resulting in a better understanding of robot-environment interaction.

We demonstrate this approach to robot programming in experiments with a Magellan Pro mobile robot, using the task of door traversal as a testbed.

1 Introduction: The RobotMODIC Project

Research presented in this paper is part of the RobotMODIC project at the Universities of Essex and Sheffield, which investigates the interaction between a mobile robot and its environment, using mathematical modelling methods. Specifically, the methods developed under this project model and characterise all aspects relevant to the robot's operation: modelling of sensor perception ("environment identification" or simulation), sensor modelling, and task modelling.

1.1 Foundations

Fundamentally, the behaviour of a robot is influenced by three components: i) the robot's hardware, ii) the program it is executing, and iii) the environment the robot is operating in. This results in a highly complex, often nonlinear system whose fundamental properties are only partially understood. The aim of the RobotMODIC project is to quantify and model this robot-environment interaction. This would allow the investigation of, for instance, i) the effects of modifications of the robot, ii) the effect of modifications of the environment on the overall behaviour of the robot, and iii) the influence of the robot control program on robot behaviour.

R. Marín et al. (Eds.): CAEPIA 2005, LNAI 4177, pp. 470–479, 2006.

The RobotMODIC project aims to "identify" — in the sense of mathematical modelling — both a mobile robot's motor responses to perceptual stimuli (task identification), and the perceptual properties of the robot's environment (environment identification). Models are represented as either linear or nonlinear polynomials, and are obtained using ARMAX (Auto-Regressive Moving Average model with eXogenous inputs), and NARMAX (Nonlinear ARMAX) system identification [1].

The techniques developed under this project represent a step towards a science of mobile robotics, because they reveal fundamental properties of the sensor-motor couplings underlying the robot's behaviour. In fact, the transparent and analysable modelling methods (ARMAX and NARMAX), have been already applied to model and characterise a robot controller [2], to achieve platform independent programming [3,4], to analyse the relationship between a mobile robot's perception, motion and position (i.e. addressing the problem of sensor-based self-localisation) [5], and to "translate" one sensor modality into another. Furthermore, the RobotMODIC procedure has also been applied to model the sensor perceptions (publication under review), in such a way that generic simulation programs could be replaced by specific, precise computer models of robot-environment interaction, derived from real-world data obtained in robotics experiments.

The work described in this article, part of the RobotMODIC process, is mostly concerned with a novel procedure to program robots through behaviour identification (section 2). The application of this novel procedure to solve a particular task in mobile robotics is described in section 3. As we'll see in section 4, transparent modelling of a robot's behaviour allows the analysis of important factors involved in robot environment interaction and also the formulation of new and testable hypotheses which lead to new approaches in experimentation and design of robot controllers.

2 Task Identification and Robot Training

2.1 Motivation and Related Work

Our recent efforts have been oriented towards the development of a novel procedure to program a robot controller, based on system identification techniques. Instead of refining an initial approximation of the desired control code through a process of trial and error, we identify the motion of a manually, "perfectly" driven robot, and subsequently use the result of the identification process to achieve autonomous robot operation.

The process works in two stages: first the robot is driven under manual control (robot training), demonstrating the behaviour we want to achieve. While the robot is being moved we log enough sensor and motor information to model the relationship between the robot's sensor perceptions and motor responses. After this first stage, a NARMAX model is estimated. This model relates robot sensor values to actuator signals, and it can be analysed and subsequently used to control the movement of the robot.

There are alternatives to the approach we used here. For instance, artificial neural networks or genetic programming could be applied to model the robot's behaviour, but have the disadvantage that the models produced are opaque and therefore difficult to analyse formally. Using a system identification approach, the behaviour of the robot is modelled through a polynomial representation that can be analysed to understand the main aspects involved in robot behaviour. Furthermore, a polynomial model is easily and accurately transferable to any robot platform with a similar sensor configuration. As this polynomial can be used to control the robot's movement directly, the program code is very compact (which is useful for applications where memory and processing speed matter).

The robot we use for our experiments is a Magellan Pro mobile robot (figure 1), equipped with front-facing laser, sonar, infrared, tactile and vision sensors. In the experiments reported here we only used the laser, which covers the semi-circle in front of the robot, and the sixteen omni-directional sonar sensors of the robot.

2.2 NARMAX Modelling

The NARMAX modelling approach is a parameter estimation methodology for identifying both the important model terms and the parameters of unknown non-linear dynamic systems. For multiple input, single output noiseless systems this model takes the form:

$$
\begin{aligned}
y(n) = f(&u_1(n), u_1(n-1), u_1(n-2), \cdots, u_1(n-N_u), u_1(n)^2, u_1(n-1)^2, u_1(n-2)^2 \\
&, \cdots, u_1(n-N_u)^2, \cdots, u_1(n)^l, u_1(n-1)^l, u_1(n-2)^l, \cdots, u_1(n-N_u)^l, u_2(n) \\
&, u_2(n-1), u_2(n-2), \cdots, u_2(n-N_u), u_2(n)^2, u_2(n-1)^2, u_2(n-2)^2, \cdots, \\
&, u_2(n-N_u)^2 \cdots, u_2(n)^l, u_2(n-1)^l, u_2(n-2)^l, \cdots, u_2(n-N_u)^l, \cdots, \\
&u_d(n), u_d(n-1), u_d(n-2), \cdots, u_d(n-N_u), u_d(n)^2, u_d(n-1)^2, u_d(n-2)^2, \cdots, \\
&u_d(n-N_u)^2, \cdots, u_d(n)^l, u_d(n-1)^l, u_d(n-2)^l, \cdots, u_d(n-N_u)^l, y(n-1), \\
&y(n-2), \cdots, y(n-N_y), y(n-1)^2, y(n-2)^2, \cdots, y(n-N_y)^2, \cdots, y(n-1)^l, \\
&y(n-2)^l, \cdots, y(n-N_y)^l)
\end{aligned}
$$

were $y(n)$ and $\boldsymbol{u}(n)$ are the sampled output and input signals at time n respectively, N_y and N_u are the regression orders of the output and input respectively and d is the input dimension. In our case, $f()$ is a non-linear polynomial expansion of the arguments. The degree l of the polynomial is the highest sum of powers in any of its terms.

The NARMAX methodology breaks the modelling problem into the following steps: i) Structure detection, ii) parameter estimation, iii) model validation, iv) prediction, and v) analysis. A detailed procedure of how these steps are done is presented in [1,6].

Any data set that we intend to model is first split in two sets (usually of equal size). The first, referred to as the estimation data set, is used to calculate the model parameters. The remaining data set, referred to as the validation set, is used to test and evaluate the model.

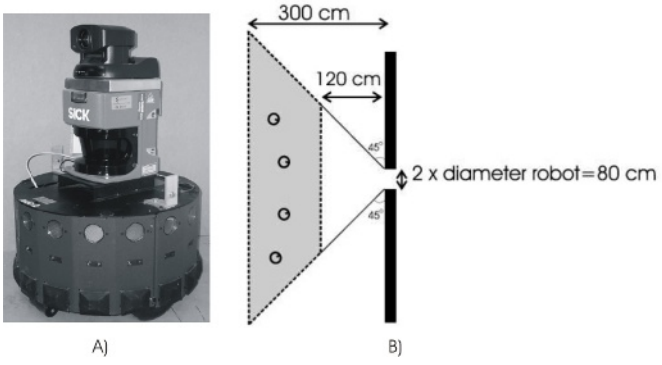


Fig. 1. Radix, the Magellan Pro robot used in our experiments. b) Experimental scenario for the door traversal behaviour, the initial positions of the robot were within the shaded area.

The number of terms of the NARMAX model polynomial can be very large depending on the number of inputs and the values of N_u, N_y ,and l. Nevertheless not all the terms are significant contributors to the computation of the output, in fact often most terms can be safely removed from the model equation without this introducing any significant errors. In order to do this, the so-called Error Reduction Ratio (ERR) [1] is computed for each term. The ERR of a term in the model is the percentage reduction in the total mean-squared error (i.e. the difference between model predicted and true system output) as a result of including (in the model equation) the term under consideration. The bigger the ERR is, the more significant the term. Model terms with ERR under a certain threshold are removed from the model polynomial.

3 Robot "Programming" Through Task Identification: Door Traversal

The example presented in this section demonstrates how robot programming through system identification works in practice. The task we want to solve with the robot is "door traversal", under the experimental conditions shown in figure 1. The trajectories of the robot after crossing 39 times the same door under manual control can be seen in figure 2 (a). The translational velocity was kept constant at 0.07 m/s, so that the human operator only controlled the steering direction and velocity of the robot at every instant.

We then identified the door traversal task, using a NARMAX process, and obtained the model given in table 1. In order to avoid making assumptions about the relevance of specific sensor signals, initially *all* ultrasound and laser measurements were taken into account. The values delivered by the laser scanner were averaged in twelve sectors of 15 degrees each (laser bins), to obtain a twelve dimensional vector of laser-distances. The inversion of laser and sonar signals in the model is a mere convenience, rather than a necessity: it means that large readings indicate close-by objects.

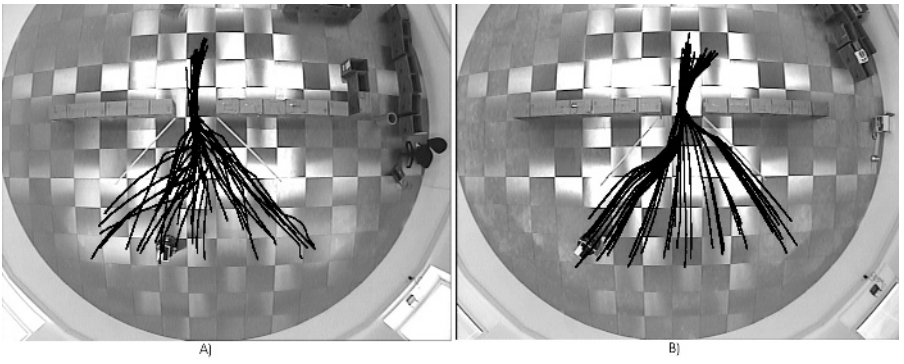


Fig. 2. a)Robot trajectories under manual control (39 runs, training data). b) Trajectories taken under model control (41 runs, test data). The white lines on the floor were used to aid the human operator in selecting start locations, they were invisible to the robot.

Table 1. NARMAX model of the steering direction and velocity $\dot{\theta}$ of the robot for the door traversal behaviour. The sonar readings are represented as $s_1, \cdots, s_{16}$, and the 12 laser bins are $d_1, \cdots, d_{12}$.

$\dot{\theta}(t) = 0.272 + 0.142 * (1/d_1(t)) - 0.470 * (1/d_3(t)) - 0.070 * (1/d_4(t)) - 0.347 * (1/d_6(t))$
$+0.157 * (1/d_8(t)) + 0.091 * (1/d_9(t)) - 1.070 * (1/s_9(t)) - 0.115 * (1/s_{12}(t))$
$+0.130 * (1/d_3(t))^2 - 0.166 * (1/d_8(t))^2 + 0.183 * (1/s_9(t))^2 + 0.081 * (1/(d_1(t) * d_3(t)))$
$-0.098 * (1/(d_1(t) * d_4(t))) - 0.382 * (1/(d_1(t) * d_5(t))) - 0.204 * (1/(d_1(t) * d_6(t)))$
$-0.049 * (1/(d_1(t) * d_8(t))) - 0.078 * (1/(d_1(t) * s_8(t))) + 0.060 * (1/(d_2(t) * s_7(t)))$
$+0.300 * (1/(d_3(t) * d_5(t))) + 0.037 * (1/(d_3(t) * s_5(t))) + 0.209 * (1/(d_3(t) * s_{12}))$
$+1.014 * (1/(d_4(t) * d_6(t))) + 0.061 * (1/(d_4(t) * s_4(t))) + 0.273 * (1/(d_4(t) * s_{12}(t)))$
$-0.536 * (1/(d_5(t) * d_6(t))) + 0.230 * (1/(d_5(t) * d_7(t))) - 0.503 * (1/(d_6(t) * d_9(t)))$
$+2.516 * (1/(d_6(t) * s_9(t))) - 0.067 * (1/(d_6(t) * s_{13}(t))) - 0.009 * (1/(d_7(t) * s_{15}(t)))$
$+0.086 * (1/(d_8(t) * s_3(t))) - 0.038 * (1/(d_8(t) * s_6(t))) - 0.060 * (1/(d_9(t) * s_4(t)))$
$-0.067 * (1/(d_{10}(t) * d_{12}(t))) - 0.040 * (1/(d_{10}(t) * s_{12}(t))) + 0.059 * (1/(d_{11}(t) * s_1(t)))$
$-0.045 * (1/(d_{12}(t) * s_7(t)))$

Having identified the model given in table 1, we used it to let the robot execute the door traversal task *autonomously*. Figure 2 (b), shows the trajectories of the robot under NARMAX model control. Door traversal was performed 41 times. The initial positions of the robot during testing were located in the same area as those used for training (see figure 1).

Figure 2 reveals some interesting phenomena: In the first door traversal under human control, the human operator moved the robot towards the centre of the door when the robot was still far from the opening. As the human operator gained experience, he was able to execute more efficient motions, nearer the door. Figure 2 (b) shows how the NARMAX model controlled the robot in a manner that was smooth in all trajectories.

To determine the degree of similarity between the trajectories achieved under human control and those observed under automatic control, we analysed the trajectories through the door, comparing the x positions the robot occupied

when it was at the centre of the opening. There is a statistically significant difference between these distributions (U-test, $p > 0.05$): the model-driven robot traverses the door more centrally than the human-driven robot.

4 Quantitative Task Characterisation

4.1 Sensitivity Analysis Through Modulation with Noise

This section shows several ways of how NARMAX models can be used to understand the main factors involved in the robot's operation in the environment.

Once a model is obtained, one would, for instance, want to estimate the influence of individual sensor readings upon the robot's global behaviour [7]. We have therefore carried out some experiments in order to have an idea of the sensitivity of our door traversal model.

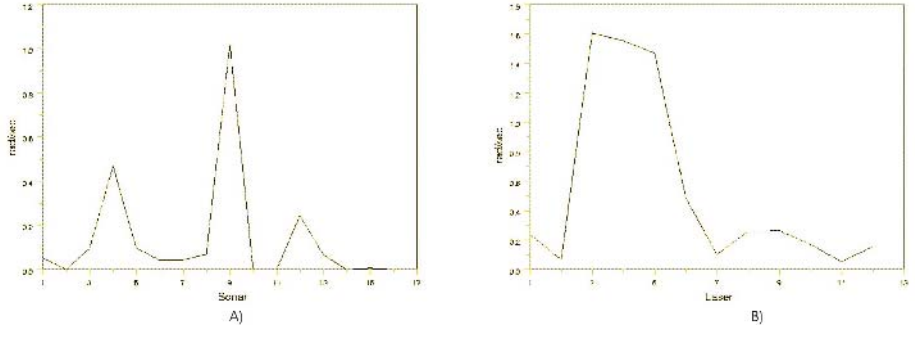


Fig. 3. Maximum change in the angular velocity when one individual sensor is modulated with noise along one of the robot's trajectories shown in figure 2(b), while all other sensors remain unchanged. In a) the sensors modulated are the ultrasound ones, while in b) we modified the laser bins.

To check the sensitivity of our model we selected randomly one of the trajectories of the robot across the door (see figure 2). To analyse the sensitivity of the steering velocity with respect to one particular sensor, we use the model to recalculate the angular velocity the robot would attain along the trajectory at every instant t, when the chosen sensor's reading $s(t)$ takes values from $0.4s(t)$ to $1.6s(t)$. It is important to notice that only one sensor is modulated, all other sensors remain unaltered. Figures 3 and 4 show one example of such an analysis.

Figures 4 and 5 show how the sensitivity varies along the trajectory of the robot for some of those sensors in which, according to figure 3, the model seems to be more sensitive. In the case of the laser information, figure 3 (b), the sensitivity seems to be higher for those laser bins which comprise the laser readings in the interval $[30°, 90°]$ (laser bins 3 to 6).

As we can see in figures 4 and 5, the perturbation of just one sensor doesn't affect the behaviour of the robot, the graphs are mostly flat in all the covered

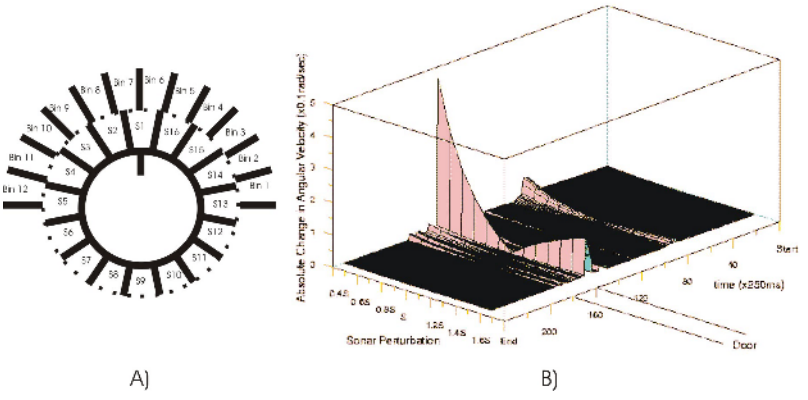


Fig. 4. a) Location of each sonar sensor and laser bin. b) Change in the angular velocity when the sonar 4 is modified.

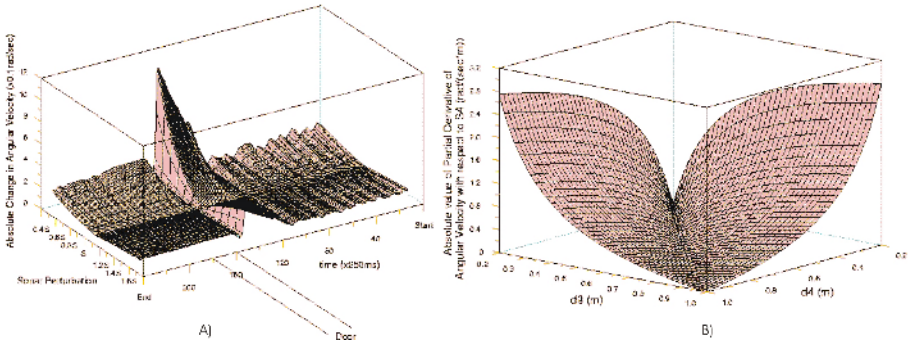


Fig. 5. (a) Change in angular velocity when sonar 9 is modulated with noise. (b) Representations of the absolute value of $\partial\dot{\theta}/\partial s_4$ when $s4 = 0.33m$.

area, indicating that our NARMAX model of door-traversal is stable and insensitive to noise. Only when the robot is close to the door a perturbation can alter the behaviour of the robot noticeably. This is reasonable, given that crucial part of the robot's operation is motion near the door, when the accuracy of *every* sensor matters.

We also applied the Monte Carlo mechanism proposed by I. M. Sobol [7] to estimate the sensitivity of the function given in table 1 with respect to individual sensor signals. This confirmed a high sensitivity of the model with respect to the rear sonar sensor 9, and also with respect to the laser bins and sonar sensors located on the right side of the robot (figure 6).

4.2 Sensitivity Analysis Through Partial Derivation

Partial derivatives can also be used for sensitivity analysis. In the case of sonars 4 and 9, for example, we get:

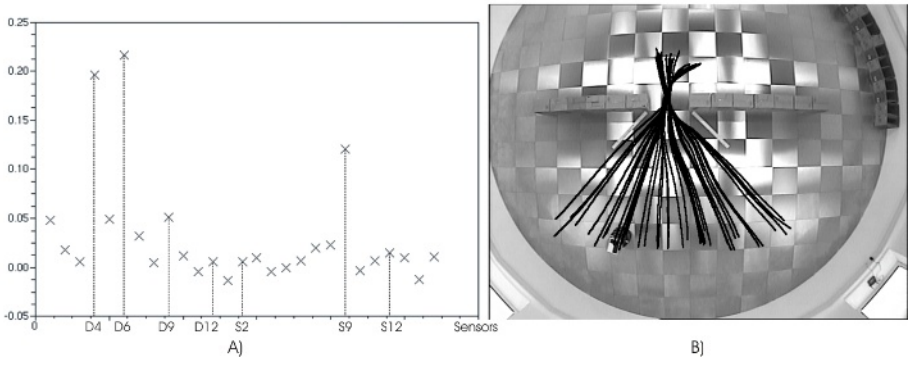


Fig. 6. a) Sensitivity of the Narmax model shown in table 1 with respect to the laser bins and sonar sensors, estimated by the Monte Carlo mechanism proposed by I.M. Sobol [7]. b) Trajectories taken under control of the revised model (table 2), 41 runs, test data.

$$\frac{\partial \dot{\theta}}{\partial s_4} = -0.076{d_4}^{-1}{s_4}^{-2} + 0.075{d_9}^{-1}{s_4}^{-2} \tag{1}$$

$$\frac{\partial \dot{\theta}}{\partial s_9} = 1.070{s_9}^{-2} - 0.366{s_9}^{-3} - 3.145{d_6}^{-1}{s_9}^{-2} \tag{2}$$

The presence of terms $1.070{s_9}^{-2}$ and $0.366{s_9}^{-3}$ in equation 2, explain why sonar 9 readings may alter the angular velocity even far away from the door.

To understand these terms, we went back to the training data and checked the values of this sensor especially far from the door. We then saw that in some of the trajectories the human operator moved the robot towards the centre of the door even when it was still far from the opening (figure 2). We therefore removed the first part of those episodes where this experimental artifact took place and re-obtained the NARMAX model. The result was that the sensitivity on sensor 9 far from the door almost disappeared. Because of this we make the hypothesis that it might be better not to consider sensor 9 in the modelling process.

Regarding sensor 4, we know that the possible change in the angular velocity due to *only* this sensor can be locally estimated as:

$$\Delta\dot{\theta} = \Delta S_4 \frac{\partial \dot{\theta}}{\partial s_4} \tag{3}$$

$| \frac{\partial \dot{\theta}}{\partial s_4} |$ increases very significantly if one of the opposite laser bins, 4 or 9, (figure 4 (a)), has a reading clearly bigger than the other (figure 5). According to equation 3 this should cause an important change in the angular velocity. Only when the two laser bins are similar (the robot is aligned with the centre of the door), is $| \frac{\partial \dot{\theta}}{\partial s_4} | = 0$.

4.3 Door Traversal Using Fewer Sensors

The results of the sensitivity analysis described in section 4.1 and 4.2, led us to a new stage in the design of our controller, where a new NARMAX model was obtained, using only the information coming from the sensors located on the right side of the robot. The new model, given in table 2, has proved to be able to solve the task equally well (figure 6).

This highlights a distinct advantage of using a transparent modelling mechanism such as a nonlinear polynomial: having identified a first model, one is often able to simplify and minimise this model through formal mathematical analysis (e.g. sensitivity analysis and partial derivatives), leading to highly efficient representations of sensor-motor couplings.

Table 2. Optimised NARMAX model of the angular velocity $\dot{\theta}$ for the door traversal behaviour. $s_{10}, \cdots, s_{16}$ are the inverted and normalised sonar readings ($s'_i = (1/s_i - 0.25)/19.75$), while $d_1, \cdots, d_6$ are the inverted and normalised laser bins $d'_i = (1/d_i - 0.12)/19.88$.

$$
\begin{aligned}
\dot{\theta}(t) = {} & 0.010 - 1.633 * d'_1(t) - 2.482 * d'_2(t) + 0.171 * d'_3(t) + 0.977 * d'_4(t) - 1.033 * d'_5(t) \\
& + 1.947 * d'_6(t) + 0.331 s'_{13}(t) - 1.257 * s'_{15}(t) + 12.639 * {d'_1}^2(t) + 16.474 * {d'_2}^2(t) \\
& + 28.175 * {s'_{15}}^2(t) + 80.032 * {s'_{16}}^2(t) + 14.403 * d'_1(t) * d'_3(t) - 209.752 * d'_1(t) * d'_5(t) \\
& - 5.583 * d'_1(t) * d'_6(t) + 178.641 * d'_1(t) * s'_{11}(t) - 126.311 * d'_1(t) * s'_{16}(t) \\
& + 1.662 * d'_2(t) * d'_3(t) + 225.522 * d'_2(t) * d'_5(t) - 173.078 * d'_2(t) * s'_{11}(t) \\
& + 25.348 * d'_3(t) * s'_{12}(t) - 24.699 * d'_3(t) * s'_{15}(t) + 100.242 * d'_4(t) * d'_6(t) \\
& - 17.954 * d'_4(t) * s'_{12}(t) - 3.886 * d'_4(t) * s'_{15}(t) - 173.255 * d'_5(t) * s'_{11}(t) \\
& + 40.926 * d'_5(t) * s'_{15}(t) - 73.090 * d'_5(t) * s'_{16}(t) - 144.247 * d'_6(t) * s'_{12}(t) \\
& - 57.092 * d'_6(t) * s'_{13}(t) + 36.413 * d'_6(t) * s'_{14}(t) - 55.085 * s'_{11}(t) * s'_{14}(t) \\
& + 28.286 * s'_{12}(t) * s'_{15}(t) - 11.211 * s'_{14}(t) * s'_{16}(t)
\end{aligned}
$$

5 Summary and Conclusion

In this article, we describe a novel mechanism to program robots through system identification, rather than an empirical trial-an-error process of iterative refinement. To achieve sensor-motor tasks, we first operate the robot under human supervision, making it follow a desired trajectory. Once data is acquired in this way, we use the NARMAX modelling approach to obtain a model which identifies the coupling between sensor perception and motor responses. This model is then used to control the robot when it moves autonomously.

We have also shown how a NARMAX model can be used to identify the main factors involved in the robot's execution of a particular task. In particular it allows not only the analysis of the stability and sensitivity of robot's operation, but also the formulation of new and testable hypotheses which lead to a new stage in the experimentation and design of a controller.

Acknowledgements

The RobotMODIC project is supported by the Engineering and Physical Sciences Research Council, grant GR/S30955/01. Roberto Iglesias is supported through grants PGIDIT04TIC206011PR, TIC2003-09400-C04-03 and TIN2005-03844.

References

1. M. Korenberg, S. Billings, Y. P. Liu, and P. J. McIlroy, "Orthogonal parameter estimation algorithm for non-linear stochastic systems," *International Journal of Control*, vol. 48, pp. 193–210, 1988.
2. U. Nehmzow R. Iglesias, T. Kyriacou and S. Billings, "Task identification and characterization in mobile robotics," in *Proceedings of TAROS 2004 - Towards Autonomous Robotic Systems*, 2004.
3. U. Nehmzow R. Iglesias, T. Kyriacou and S. Billings, "Robot programming through a combination of manual training and system identification," in *2nd European Conference on Mobile Robots*, 2005.
4. R. Iglesias T. Kyriacou, U. Nehmzow and S. Billings, "Cross-platform programming through system identication," in *Proceedings of TAROS 2004 - Towards Autonomous Robotic Systems*, 2004.
5. R. Iglesias U. Nehmzow, T. Kyriacou and S. Billings, "Self-localisation through system identification," in *2nd European Conference on Mobile Robots*, 2005.
6. S. Billings and S. Chen, "The determination of multivariable nonlinear models for dynamical systems," in *Neural Network Systems, Techniques and Applications*, C. Leondes, Ed., pp. 231–278. Academic press, 1998.
7. I.M. Sobol, "Sensitivity estimates for nonlinear mathematical models," *Mathematical Modelling and Computational Experiment*, vol. 1, pp. 407–414, 1993.

Author Index