Advances in Artificial Intelligence Applications

Research on Computing Science

Research on Computing Science es una publicación trimestral, de circulación internacional, editada por el Centro de Investigación en Computación del IPN, para dar a conocer los avances de investigación científica y desarrollo tecnológico de la comunidad científica internacional. **Volumen 17** Noviembre, 2005. Tiraje: 500 ejemplares. *Certificado de Reserva de Derechos al Uso Exclusivo del Título* No. 04-2004-062613250000-102, expedido por el Instituto Nacional de Derecho de Autor. *Certificado de Licitud de Título* No. 12897, *Certificado de licitud de Contenido* No. 10470, expedidos por la Comisión Calificadora de Publicaciones y Revistas Ilustradas. El contenido de los artículos es responsabilidad exclusiva de sus respectivos autores. Queda prohibida la reproducción total o parcial, por cualquier medio, sin el permiso expreso del editor, excepto para uso personal o de estudio haciendo cita explícita en la primera página de cada documento. Se usó la imagen obtenida de la siguiente dirección, para el diseño de la portada: www.absolutewallpapers.com/wallpapers/3dwallpapers/fractal/fractal_5.jpg. Impreso en la Ciudad de México, en los Talleres Gráficos del IPN – Dirección de Publicaciones, Tres Guerras 27, Centro Histórico, México, D.F. Distribuida por el Centro de Investigación en Computación, Av. Juan de Dios Bátiz S/N, Esq. Av. Miguel Othón de Mendizábal, Col. Nueva Industrial Vallejo, C.P. 07738, México, D.F. Tel. 57 29 60 00, ext. 56571.

Editor Responsable: *Juan Humberto Sossa Azuela*, *RFC SOAJ560723*

Volume 17

Volumen 17

Advances in Artificial Intelligence Applications

Volume Editors:
Editores del Volumen

Alexander Gelbukh
Raúl Monroy

Instituto Politécnico Nacional
Centro de Investigación en Computación
México 2005

ISSN: 1665-9899

Instituto Politécnico Nacional (IPN)
Centro de Investigación en Computación (CIC)
Av. Juan de Dios Bátiz s/n esq. M. Othón de Mendizábal
Unidad Profesional "Adolfo López Mateos", Zacatenco
07738, México D.F., México

http://www.ipn.mx
http://www.cic.ipn.mx

Printing: 500
Impresiones: 500

Printed in Mexico
Impreso en México

Preface

Artificial Intelligence is a branch of computer science aimed at providing the computer elements of human-like behavior such as ability to think, learn by example, doubt, act, see, and speak. Since its beginning artificial intelligence research has been influenced and inspired by nature—in the first place, by the way human being accomplishes such tasks. Recently, the repertoire of artificial intelligence methods was enriched by other naturally inspired optimization techniques, such as genetic algorithms, swarm intelligence, or ant colony optimization. In addition to creating human-like-behaving machines, modern artificial intelligence provides a very powerful platform for solving a wide range of super-complex optimization problems.

This volume presents original research papers on application of artificial intelligence techniques to practical real-life problems, from economy and education to creating of physical intelligent robots. It is structured into eight thematic fields representative of the main current areas of application of AI: Economy, Commerce, and Education; Information Systems and Data Mining; Information Security and Infrastructure; Anti-Spam filtering and Email Virus Detection; Image processing and Human-Computer Interaction; Data compression; Robotics; Route Planning. The previous volume of this journal has presented original papers devoted to the internal art and craft of artificial intelligence research: its theoretical foundations, specific techniques, and research methodologies.

Total of 61 full papers by 156 authors from 15 different countries were submitted for evaluation, see Tables 1 and 2. Each submission was reviewed by three independent members of the Editorial Board of the volume. This volume contains revised versions of 26 papers, by 74 authors, selected for publication after thorough evaluation. Thus the acceptance rate was 42.6%. In Table 1, the number of papers by country was calculated by the shares of all authors of the paper: e.g., if a paper has three authors, two from Mexico and one from USA, then we incremented the counter for Mexico by 0.66 (two authors of three) and the counter for USA by 0.33. Table 2 presents the statistics of papers by topics according to the topics indicated by the authors; note that a paper can be assigned more than one topic.

The academic and editorial effort resulting in this volume was carried out in collaboration with, and was supported by, the Mexican Society for Artificial Intelli-

Table 1. Statistics of authors and papers by country.

Country	Submitted		Accepted	
	Auth	Pap	Auth	Pap
Brazil	14	4	–	–
Canada	2	1	2	1
China	10	9	–	–
Czech Republic	1	1	–	–
France	1	0.33	1	0.33
Germany	1	1	–	–
Korea, South	4	2	–	–
Lithuania	2	1	–	–
Mexico	79	30.83	49	18.17
Norway	2	1	–	–
Poland	2	1	–	–
Spain	19	5	16	4.5
Taiwan	6	2	6	2
UK	2	1	–	–
USA	1	0.33	–	–
total:	156	61	74	26

Table 2. Statistics of submitted and accepted papers by topic.

Topic	Submitted	Accepted
Expert Systems / KBS	1	1
Multiagent systems and Distributed AI	13	5
Knowledge Management	3	2
Intelligent Interfaces: Multimedia, Virtual Reality	6	1
Natural Language Processing / Understanding	4	1
Computer Vision	4	2
Neural Networks	10	2
Genetic Algorithms	7	3
Fuzzy logic	5	2
Machine Learning	8	3
Intelligent Tutoring Systems	6	1
Data Mining	4	1
Knowledge Acquisition	3	–
Knowledge Representation	5	2
Knowledge Verification, Sharing and Reuse	1	–
Ontologies	6	1
Constraint Programming	1	–
Case-Based Reasoning	4	1
Nonmonotonic Reasoning	1	1
Spatial and Temporal Reasoning	1	1
Robotics	11	6
Planning and Scheduling	3	–
Navigation	1	1
Assembly	2	1
Hybrid Intelligent Systems	12	5
Logic Programming	1	1
Intelligent Organizations	3	3
Uncertainty / Probabilistic Reasoning	4	1
Bioinformatics	2	1
Philosophical and Methodological Issues of AI	1	–
Other	13	5

gence (SMIA). We cordially thank all people involved in its preparation. In the first place these are the authors of the papers constituting it: it is the excellence of their research work that gives sense to the work of all other people involved. We thank the members of the Editorial Board of the volume and additional referees. We express our gratitude to Álvaro de Albornoz, Ángel Kuri, Hugo Terashima-Marín, Francisco J. Cantú-Ortiz, Leticia Rodríguez, Fernando J. Jaimes, Rogelio Soto-Rodríguez, Hiram Calvo, Manuel Vilares, and Sulema Torres for their significant contribution at various stages of preparation of the volume. The submission, reviewing, and selection process was supported for free by the EasyChair system, www.EasyChair.org.

Alexander Gelbukh November 2005
Raúl Monroy

Table of Contents
Índice

Page/Pág.

Economy, Commerce and Education

Selection of Segments to be Sourced from Low Cost Countries for a Global Industrial Equipment Manufacturer based on a Multi-Attribute Decision Support System.. 3
Antonio Jiménez, Luis C. Rodríguez, Alfonso Mateos and Sixto Ríos-Insua

Decision Support Systems for Portfolio Optimization...................... 13
María A. Osorio-Lama and Abraham Sánchez-López

Location-based Support for Commerce using Multiagent Negotiation 23
Ramon Brena, Luis Marcelo Fernández, Jesús Héctor Domínguez and José Luis Aguirre

Ontologies for Student and Domain Models in Adaptive and Collaborative Learning System... 33
José M. Gascueña, Antonio Fernández-Caballero and Pascual González

Information Systems and Data Mining

Shallow Parsing and Information Extraction 45
Diego Uribe

Identifying Information from Heterogeneous and Distributed Information Sources for Recommender Systems.............................. 55
Silvana Vanesa Aciar , Josefina López Herrera and Josep Lluis de la Rosa

Model Selection in Data Mining: A Statistical Approach.................. 65
Jose A. Frydman P., Francisco J. Cantú Ortiz, Jorge H. Sierra Cavazos and Rubén Morales-Menéndez

Use of an ANFIS Network for Relative Humidity Behaviour Modelling on the South Region of Jalisco, México. 75
Marco Gamboa, Marco Castro and Francisco Herrera

Information Security and Infrastructure

Multi-Agent System Design Based on Security Policies 87
Zeus Andrade Zaldívar, Ma. De los Ángeles Junco Rey, Jorge Adolfo Ramírez Uresti, José Arturo Tejeda Gómez and Enrique David Espinosa Carrillo

An Artificial Manager for Security Policies in Organizations........... 97
Karen García, Raúl Monroy and Jesús Vázquez

A service-oriented approach for integrating multiagent systems with web-services in a collaboration application 107
Eduardo H. Ramirez and Ramon F. Brena

Anti-Spam Filtering and Email Virus Detection

A Collaborative Learning Method For Spam Filtering......................119
Hsiu-Sen Chiang, Jui-Chi Shen, Dong-Her Shih and Chia-Shyang Lin

Analyzing the Impact of Corpus Preprocessing on Anti-Spam Filtering Software 129
J. R. Méndez, E. L. Iglesias, F. Fdez-Riverola, F. Díaz and J.M. Corchado

Misuse Detectionof Email Viruses base on SOM with k-medoids....139
Dong-Her Shih, Sheng-Fei Hsu, Hsiu-Sen Chiang and Chun-Pin Chang

Image Processing and Human-Computer Interaction

A Statistical Learning-Based Method for Color Correction of Underwater Images 151
Luz Abril Torres-Méndez and Gregory Dudek

Recognition of Shorthand Writing using Neural Networks 161
Diana M. Vázquez, Karla L. Segovia and Roberto A. Vázquez

Data Compression

Data Compression Using a Dictionary of Patterns............................173
Angel Kuri and Jose Galaviz

MSIM: A Pattern Based Lossless Data Compressor..........................183
Angel Kuri-Morales and Oscar Herrera-Alcántara

Practical Estimation of Kolmogorov Complexity using
Highly Efficient Compression Algorithms..193
Angel Kuri-Morales, Oscar Herrera, José Galaviz and Martha Ortiz-Posadas.

Robotics

Steering Control of an Ackerman Mobile Robot
Using Fuzzy Logic 205
Enrique Alarcon-Avila, Stivalis Anahi Martinez-Cuevas and Alejandro Rangel-Huerta

Evolving Robot Behavior for Centralized Action Selection 213
Fernando Montes González and José Santos Reyes

Formations in Collective Robotics 223
Yazid León Fernández de Lara and Angélica Muñoz Meléndez

MINI-TRANS: A Multi-robot System with Self-assembling
Capabilities...233
María Guadalupe Jiménez Velasco and Angélica Muñoz Meléndez

Biofeedback Agents for Electromyocontroled teleoperated Robots ..243
Sandra Diaz, Carlos Nieto, Marissa Diaz and Ricardo Swain

Route Planning

Extending and applying PP language: An answer set planning
problem language 255
Claudia Zepeda, Mauricio Osorio, Christine Solnon and David Sol

Planning motion for animated characters..265
Abraham Sánchez López, Josué Sánchez Texis and René Zapata

Economy, Commerce, and Education

Selection of Segments to be Sourced from Low Cost Countries for a Global Industrial Equipment Manufacturer based on a Multi-Attribute Decision Support System

Antonio Jiménez, Luis C. Rodríguez, Alfonso Mateos and Sixto Ríos-Insua

Technical University of Madrid, Artificial Intelligence Department, Campus de Montegancedo S/N, Boadilla del Monte, 28660 Madrid, Spain
ajimenez@fi.upm.es, lc.rodriguez@accenture.es, {amateos,srios}@fi.upm.es
http://www.dia.fi.upm.es/grupos/dasg/index.htm

Abstract. We introduce a complex decision-making problem, the prioritization of potential high-profit category segments to be sourced from low cost countries, where several conflicting criteria must be taken into account simultaneously to help focus the attention on developing low cost countries sourcing strategies for candidate segments, where potential savings are higher and risk is minimum. The GMAA system will be used for this purpose. It is a decision support system based on the Decision Analysis cycle that accounts for incomplete information concerning the inputs, where so-called decision-making with partial information plays a key role.

1 Introduction

Competitive pressures are forcing companies to reduce their overall costs, while delivering faster and more diverse product portfolios to be more responsive to customers and competitors. In response to these pressures, companies are increasingly taking advantage of the opportunity to source from low cost countries (LCC) to achieve significant savings and give their organizations a competitive advantage.

For a global industrial equipment manufacturer with material costs accounting for about 50% of the value of its final products, sourcing performance is crucial to Original Equipment Manufactured (OEM) competitiveness. OEM management identified a number of potential purchasing categories for which OEM's different divisions will coordinate their sourcing activities to reduce total cost and optimize the supplier base, achieving significant savings on this addressable expenditure. OEM's overall strategy was to seek high quality and service levels, while minimizing total cost of creating a cost-efficient production process.

Looking to drive more cost-effective and global supply chains, the procurement organization was leveraging the procurement function to identify low cost and potential reliable overseas sources of supply and rapidly prioritizing the effort in terms of

© A. Gelbukh, R. Monroy. (Eds.)
Advances in Artificial Intelligence Theory
Research on Computing Science 16, 2005, pp. 3-12

high-profit category segments and LCC regions to gain a foothold in emerging markets. However, the sourcing function within the company faced specific constraints.

Even though multinational companies have been sourcing from LCC for many years, purchasing in these regions is often very risky, and a number of companies spend a lot of energy identifying and minimizing these risks (identifying reliable sources, political instability, currency risks, longer lead-times, more complex logistics, different/non-existent legal structures,…).

Typical incremental cost reductions of 15%-20% can be achieved by sourcing from LCC. Nonetheless, to move the supply source for some specific segment categories to these regions, these segments have to be proven to have a comprehensive risk assessment, balanced against potential for lower costs. Although benefits are compelling, they come with significant challenges.

While there is no single approach to entering the LCC market, the first critical step is to conduct a comprehensive category assessment and prioritization to determine opportunities for sourcing from LCC, allowing the company to assess LCC by priority segment and reduce the "time-to-benefit" realization of its LCC sourcing program.

For the purpose of determining the highest profit potential category segments to be sourced from LCC, a range of conflicting criteria were taken into account simultaneously to provide the most relevant information about other factors. Therefore, the promise of significant cost reductions is not the only consideration, and the country, industry and supplier risks will be key factors for application during the prioritization of the category segments. In this case, the responsible procurement organization has evolved into a formal decision process in which other strategic issues related to LCC sourcing activities were quantified and formally incorporated into the analysis, where the potential for lower costs was only one factor in the objectives of the purchaser.

We propose using the *Generic Multi-Attribute Analysis* (GMAA[1]) to deal with the above complex decision-making problem, [1,2]. The GMAA system is a PC-based decision support system based on the Decision Analysis (DA) cycle that accounts for incomplete information concerning the inputs, i.e., alternative performances, component utilities and objective weights. It uses an additive multiattribute utility model to evaluate the alternatives under consideration and includes different tools for performing co-called *decision-making with partial information* to take advantage of the imprecise inputs, see [3].

We have divided the paper, according to DA stages, into three sections. The first section deals with problem structuring, in which an objective hierarchy is built, attributes are established for the lowest-level objectives and the alternatives to be evaluated are identified, as are their performances in terms of the above attributes. Next, in the second section, stakeholder preferences are quantified, which implies assessing component utilities for the different attributes and the relative importance of objectives in the hierarchy by means of weights. The third section focuses on the evaluation of alternatives and sensitivity analysis. Finally, some conclusions are provided in the fourth section.

---

[1] http://www.dia.fi.upm.es/~ajimenez/GMAA

2 Problem Structuring

As mentioned above, the overall objective of this complex decision-making problem is to create a cost efficient production process by determining the most highest profit potential category segments to be sourced from LCC at the lowest risk. For this purpose, we have to take into account several conflicting objectives that were structured in an objective hierarchy as follows:

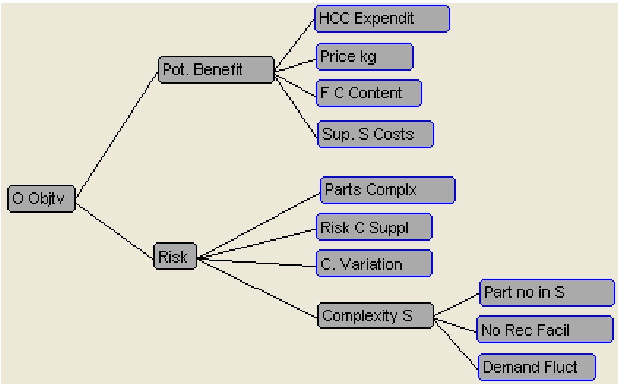


Fig. 1. Objectives hierarchy to create a cost efficient production process

The Overall Objective (O.Objtv) was split into two main sub-objectives: *Potential Benefits* (Pot. Benefit) and *Risks* (Risk). *Potential Benefits* were measured in terms of four sub-objectives. The *Total annual expenditure* (HCC Expendit) on all parts in the segment not sourced from LCC. The expenditure is an indicator of the potential volume with which we are dealing. The higher the expenditure is, the more room there is for savings. The *Price per kg* (Price kg) indicates the price regarding the value-added for the parts produced in high cost countries (HCC). The higher the HCC price/kg value-added represents high potential benefit. The *Factor cost content* (F C Content) is subject to comparison between HCC and LCC. Labor is the main factor cost to be taken into account. The higher the labor content is, the larger is the window for differences in cost between sourcing countries. High labor content represents potential high cost savings when sourcing from LCC. Finally, *Supplier switching costs* (Sup. S Costs) is the cost when switching from the current supplier set-up to a new supplier. The higher the switching cost, the lower the potential benefit. Tooling cost is the most important and most easily quantifiable switching cost to take into account. Other switching costs can be considered if known.

On the other hand, *Risks* is split into four sub-objectives. *Complexity of parts* (Complx Parts) represents part of the risk selecting a new supplier. Technical issues related to quality and material specification could be added to the assessment of the total complexity of parts in each segment. The higher the complexity, the higher the risk is. *Risk with current suppliers* (Risk C Suppl) quantifies the number of segments the supplier is supplying at that moment. Moving business in one segment from the current supplier to LCC will influence the supply of the other segments (price increasing, production stop, low performance, etc.). Therefore, the more segments supplied by one supplier, the higher the risk when moving to LCC.

The *Coefficient of variation* (C. Variation) tells us how homogeneous the price per kg of the parts in the segment is. The higher the coefficient of variation, the greater the risk, because there is more variation in the way the different parts of the segment are handled. Finally, *Complexity of segments* (Complexity S) represents supply chain issues in relation to the purchase of parts from a larger perspective. The *Number of parts within a segment* (Part no in S), the *Number of receiving facilities* for the parts in the segment (No Rec Facil) and *Demand fluctuation* (Demand Fluct) are the main quantifiable criteria to be taken into consideration. Table 1 shows the attribute names, units and ranges for the lowest-level objectives in the hierarchy.

Table 1. Attribute names, units and ranges

Attribute name	Units	Range
A1: HCC Expendit	Million euros	[0, 6]
A2: Price kg	Euros per kg	[0, 30]
A3: F C Content	% Labor Costs	[0, 100]
A4. Sup. S Costs	Discrete values	Low, Medium or High
A5: Parts Complx	Subjective Scale	[0,1]
A6: Risk C Suppl	Discrete values	1, 2, 3 or more segments
A7: C. Variation	% Variation	[0, 100]
A8: Part no in S	No. of parts	[0, 650]
A9: No Rec Facil	Discrete values	1, 2, 3-6, 7-8, 9 or more rec. facilities
A10: Demand Fluct	% Fluctuation	[0-100]

The following non-metallic product segments were identified: *SG1* (Polyurethane floor mats), *SG2* (Insulation parts), *SG3* (Fiberglass insulation & liner), *SG4* (Hydraulic hoses), *SG5* (Rubber mounts), *SG6* (Silicone hoses), *SG7* (Air hoses), *SG8* (Plastic injected ABS parts), *SG9* (Plastic injected ASA parts), *SG10* (Nylon Hydraulic tanks), *SG11* (Rotomoulded polyolefin plastic parts), *SG11* (Thermoformed ABS plastic parts), *SG12* (Thermoformed polyolefin plastic parts), *SG13* (InterWet ABS + polyurethane parts), *SG14* (Low compression molding composite parts), *SG15* (Reaction injection molding dicyclopentadiene parts), *SG16* (Reaction injection molding dicyclopentadiene hoods with metal inserts), *SG18* (Resin transfer molding injection molding composite parts) and *SG19* (Hand lay-up composite parts S).

Table 2 shows performances in terms of the attribute for the twenty segments under consideration.

Table 2. Non-metallic product segments and their performances

	A1	*A2*	*A3*	*A4*	*A5*	*A6*	*A7*	*A8*	*A9*	*A10*
SG1	1.73	3.8	26%	Low	0.33	2	23%	40	2	8%
SG2	1.07	8.4	21%	Low	0.92	2	76%	32	6	18%
SG3	0.72	14.7	29%	Low	0.85	6	57%	245	7	6%
SG4	1.20	13.7	21%	Low	0.54	4	50%	623	7	22%
SG5	1.53	18.7	22%	Medium	0.15	3	65%	80	8	13%

SG6	0.91	40	34%	Low	0.11	5	80%	43	5	11%
SG7	0.63	23	31%	Low	0.48	7	75%	54	9	9%
SG8	0.75	18	22%	High	0.47	1	15%	6	1	5%
SG9	0.65	7.4	17%	High	0.55	1	10%	45	8	13%
SG10	4.75	11.8	16%	High	0.11	1	16%	16	6	1%
SG11	5.10	12	47%	Medium	0.91	1	40%	48	9	14%
SG12	1.50	22.6	30%	Medium	0.89	4	63%	200	7	10%
SG13	0.95	8.7	33%	Medium	0.88	2	72%	25	7	1%
SG14	0.64	18.9	33%	Medium	0.12	3	39%	25	1	15%
SG15	1.49	8.2	28%	High	0.15	1	33%	36	7	3%
SG16	1.58	9.6	23%	High	0.47	1	20%	30	3	10%
SG17	2.49	22.1	28%	High	0.17	2	25%	30	6	10%
SG18	3.94	24.6	31%	High	0.19	1	47%	38	4	3%
SG19	2.30	14.1	35%	Medium	0.48	1	31%	12	2	1%

Note that although the above table includes precise values, uncertainty about some of them was taken into account by means of percentage deviations. Specifically, 5% and 3% deviations were introduced in *A2: Price kg* and *A3: F C Content*, respectively, for all the segments under consideration, except *SG2*, *SG8* and *SG9* with 10% and 7% deviations, respectively.

3 Preferences Quantification

Quantifying stakeholder preferences implies, on the one hand, assessing component utilities for the attributes under consideration that represent stakeholder preferences for the possible attribute values, and, on the other, eliciting objective weights that represent their relative importance throughout the hierarchy.

The GMAA system provides methods for quantifying preferences; see [1,2]. In both cases (component utilities and weight assessment) the stakeholders are allowed to provide imprecise information, leading to imprecise utilities and weights. Note that this makes the system suitable for group decision-making because individual conflicting views can be captured through value intervals.

The GMAA system was used to assess components utilities. Imprecise utilities for discrete values were provided for some attributes, while imprecise linear piecewise utility functions were assessed for others. Figure 2 shows the assessed imprecise linear piecewise utility function for *A1: HCC Expendit*, while Figure 3 shows the imprecise utilities provided for the three possible attribute values (1, 2 and 3 or more segments) in *A6: Risk C Suppl*.

A direct assignment and a method based on trade-offs were used to elicit objective weights representing their relative importance throughout the hierarchy, [4].

Remember that attribute weights for the decision, used in the additive multi-attribute model to evaluate alternatives, are assessed by multiplying the objective weights in the path from the *Overall Objective* to the respective attribute. Figure 4 shows the resulting attribute weights for the decision for the problem under consideration.

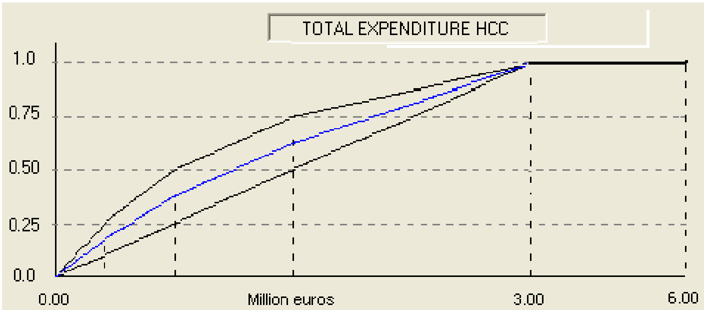


Fig. 2. Component utilities for *A1: HCC Expendit*

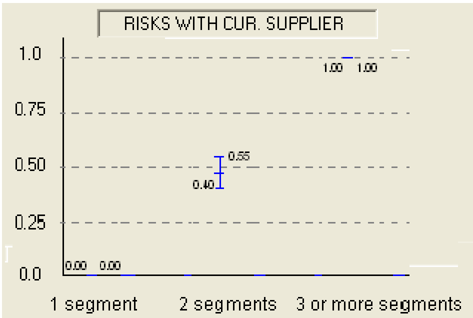


Fig. 3. Utilities for discrete attribute values in *A6: Risk C Suppl*

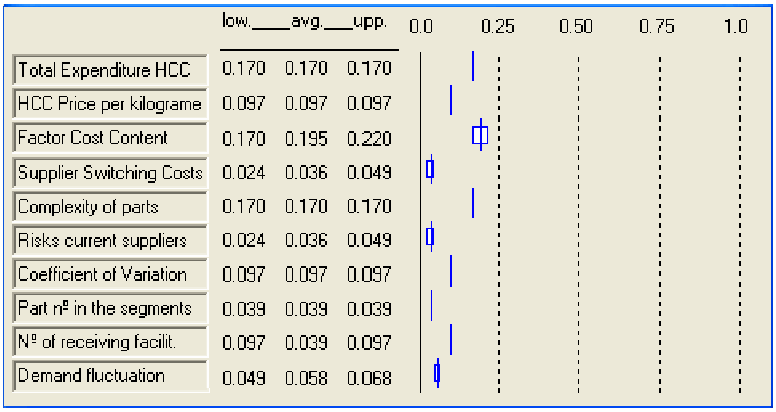


Fig. 4. Attribute weights for the decision

It is important to note that the two main sub-objectives, *Potential Benefits* and *Risks*, were initially equally important, i.e., their respective weights were 0.5, and the summation of the average decision-making weights for attributes stemming from either is 0.5.

4 Evaluation of Alternatives and Sensitivity Analysis

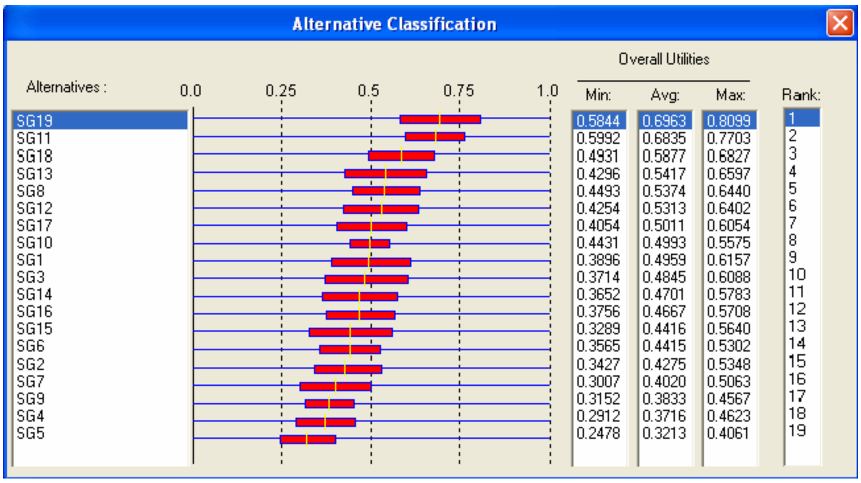
As mentioned earlier, an additive multi-attribute utility function was used to evaluate the segments under consideration. It takes the form

$$u(S^i) = \sum_{j=1}^{10} w_j u_j(x_j^i) \qquad \textbf{(1)}$$

where w_j is the *j-th* attribute decision-making weight, x_j^i is the performance of segment S^i for the *j-th* attribute and $u_j(x_j^i)$ is the component utility associated with the above segment performance. For the reasons described in [5,6], we consider (1) to be a valid approach.

As the system admits imprecision concerning component utilities and weights and uncertainty about segment performances, the above additive model was suitable for assessing the average overall utilities on which the ranking of segments is based, and minimum and maximum overall utilities that give further insight into the robustness of this ranking, see Figure 5.

Looking at Figure 5, *SG19*, *SG11*, *SG18* and SG*13* are the best ranked segments, with average overall utilities of 0.6963, 0.6835, 0.5877 and 0.5417, respectively; while *SG9*, *SG4* and *SG5* are the worst ranked segments, with average overall utilities of 0.3833, 0.3716 and 0.3213. Although *SG19* appears to be the most highly recommended segment, the overlapped utility intervals (ranking robustness) should be examined in a more detail through the sensitivity analysis (SA).



Alternatives	Min	Avg	Max	Rank
SG19	0.5844	0.6963	0.8099	1
SG11	0.5992	0.6835	0.7703	2
SG18	0.4931	0.5877	0.6827	3
SG13	0.4296	0.5417	0.6597	4
SG8	0.4493	0.5374	0.6440	5
SG12	0.4254	0.5313	0.6402	6
SG17	0.4054	0.5011	0.6054	7
SG10	0.4431	0.4993	0.5575	8
SG1	0.3896	0.4959	0.6157	9
SG3	0.3714	0.4845	0.6088	10
SG14	0.3652	0.4701	0.5783	11
SG16	0.3756	0.4667	0.5708	12
SG15	0.3289	0.4416	0.5640	13
SG6	0.3565	0.4415	0.5302	14
SG2	0.3427	0.4275	0.5348	15
SG7	0.3007	0.4020	0.5063	16
SG9	0.3152	0.3833	0.4567	17
SG4	0.2912	0.3716	0.4623	18
SG5	0.2478	0.3213	0.4061	19

Fig. 5. Ranking of segments and overall utilities

The GMAA system allows users to select another objective to rank by. In our problem it could be very interesting to view the ranking of alternatives for the main sub-objectives, *Potential Benefits* and *Risks*, see Figure 6. Note that both objectives were equally important.

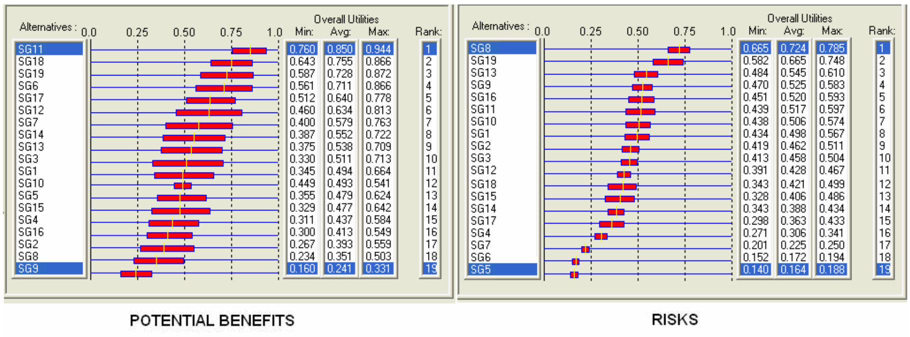


Fig. 6. Ranking of segments for *Potential Benefits* and *Risks*

Note that the best ranked segments for *Potential Benefits* are *SG11* and *SG18*, but they are ranked sixth and twelfth for *Risks*. On the other hand, *SG19* is ranked third and second for both objectives, respectively. Taking into account that *Potential Benefits* and *Risks* are equally important, this matches the ranking for the *Overall Objective*, in which *SG19* is the best ranked segment.

SA should be considered as a source of stimulation to make stakeholders think about the problem in more depth and can give further insight into the robustness of the recommendations. [3,7] introduce a framework for SA in multi-objective decision making.

The GMAA system includes several types of SA. First, non-dominated and potentially optimal alternatives (segments) can be assessed, [8]. In our problem, only three segments, *SG11*, *SG18* and *SG19*, are non-dominated and potentially optimal. Consequently, we should focus the analysis on these segments and discard the remainder because dominated segments can never be the optimal. Note that these were the best ranked segments.

We can also perform Monte Carlo simulation techniques for SA, [9], which allows simultaneous changes to attribute weights and generates results that can be easily analyzed statistically through box diagrams to provide more insights into the multi-attribute model recommendations.

The system selects the attribute weights at random within the respective normalized weight intervals in Figure 4 using a computer simulation program. Each combination of attribute weights is then used to assess a segment's ranking and, finally, the system computes several statistics about these rankings for each segment, like minimum, maximum, mean..., which are output by means of a multiple box plot, see Figure 7.

Looking at the box plots for *SG11*, *SG18* and *SG19*, we realize that they are always ranked second, third and first, respectively. Therefore, we can conclude that the segment category with the best tradeoff between potential benefit and risks to be sourced from LCC is *SG19: Hand lay-up composite parts*. However, we were not just interested in the best segment to be sourced from LCC, our aim was to identify a segment set with a good enough tradeoff between potential benefit and risk.

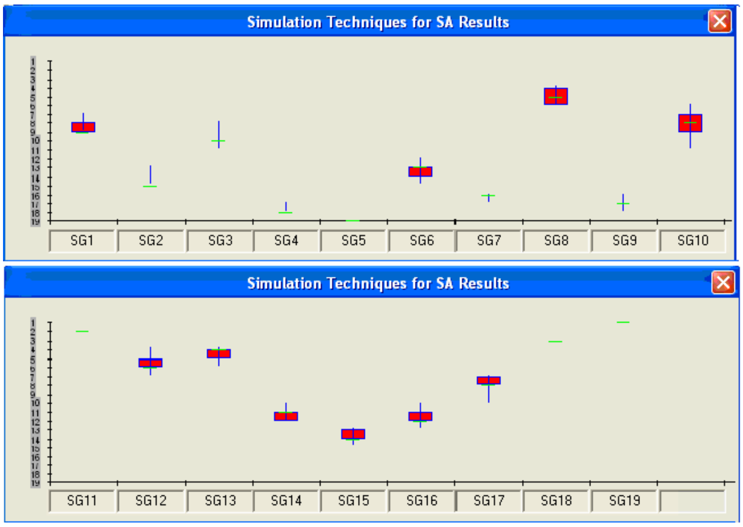


Fig. 7. Results of Monte Carlo simulation techniques

Taking into account the above segment's rankings and the results of SA, OEM management finally recommended the best ranked segments accounting for the 60% of the total expenditure of the non-metallic category segments to be sourced from LCC, see Figure 8.

Final recommended segments

Rank	Alternatives	Overall Utilities Min:	Overall Utilities Avg:	Overall Utilities Max:	Total Spend w/HCC	% spend	Cumm %
1	SG19	0.584421	0.696259	0.809945	2,30	6,78%	6,78%
2	SG11	0.599205	0.683484	0.770322	5,10	15,03%	21,81%
3	SG18	0.493066	0.587714	0.682714	3,94	11,61%	33,42%
4	SG13	0.429618	0.541691	0.659655	0,95	2,80%	36,22%
5	SG08	0.449297	0.537422	0.643996	0,75	2,21%	38,43%
6	SG12	0.425394	0.531276	0.640246	1,50	4,42%	42,85%
7	SG17	0.405360	0.501112	0.605404	2,49	7,34%	50,19%
8	SG10	0.443123	0.499324	0.557503	4,75	14,00%	64,19%
9	SG01	0.389639	0.495854	0.615723	1,73	5,10%	69,29%
10	SG03	0.371448	0.484540	0.608774	0,72	2,12%	71,41%
11	SG14	0.365216	0.470133	0.578293	0,64	1,89%	73,30%
12	SG16	0.375603	0.466701	0.570811	1,58	4,66%	77,95%
13	SG15	0.328941	0.441557	0.564017	1,49	4,39%	82,35%
14	SG06	0.356492	0.441473	0.530208	0,91	2,68%	85,03%
15	SG02	0.342685	0.427510	0.534785	1,07	3,15%	88,18%
16	SG07	0.300710	0.401960	0.506272	0,63	1,86%	90,04%
17	SG09	0.315248	0.383333	0.456682	0,65	1,92%	91,95%
18	SG04	0.291185	0.371589	0.462338	1,20	3,54%	95,49%
19	SG05	0.247849	0.321267	0.406146	1,53	4,51%	100,00%

Fig. 8. Finally recommended non-metallic product segments

Therefore, the company balances potential benefit and risk for category segments and, at the same time, handles the effort and costs to assess LCC attractiveness and conduct supplier identification and screening activities instead of just looking at savings.

6 Conclusions

In this paper, we have introduced a complex decision-making problem, the selection of non-metallic category segments by an original equipment manufacturer to be sourced from low cost countries taking into account conflicting criteria, potential benefit and the risks involved.

We have made provision for all the stages of the Decision Analysis cycle using the GMAA system, a user-friendly decision support system based on an additive multi-attribute utility model and that accounts for incomplete information about the problem parameters. We have achieved a final recommendation on the basis of the ranking of non-metallic category segments. Best ranked segments accounting for 60% of the total annual expenditure of parts in the segment not sourced from LCC are those to be recommended.

Acknowledgments

This paper was supported by the Spanish Ministry of Education and Science projects TSI2004-06801-C04-04 and MTM2004-21099-E.

References

1. Jiménez, A., Ríos-Insua, S., Mateos, A.: A Decision Support System for Multi-Attribute Utility Evaluation based on Imprecise Assignments. Decis. Support Syst. 36 (2003) 65–79
2. Jiménez, A., Ríos-Insua, S., Mateos, A.: A Generic Multi-Attribute Analysis System. Comput. Oper. Res. 33 (2006) 1081-1101.
3. Ríos Insua, D. (ed.): Sensitivity Analysis in MultiObjective Decision Making. Lecture Notes in Economics and Mathematical Systems Vol. 347, Springer-Verlag, Berlin (1990)
4. Keeney, R.L., Raiffa, H.: Decision with Multiple Objectives: Preferences and Value-Tradeoffs. Wiley, New York (1976)
5. Raiffa, H.: The Art and Science of Negotiation. Harvard University Press, Cambridge (1982)
6. Stewart, T.J.: Robustness of Additive Value Function Method in MCDM. Journal of Multi-criteria Decision Analysis 5 (1996) 301-309
7. Ríos Insua, D., French, S.: Framework for Sensitivity Analysis in Discrete Multi-Objective Decision-Making. Eur. J. Oper. Res. 54 (1991) 176-190
8. Mateos, A., Jiménez, A., Ríos-Insua, S.: Solving Dominance and Potential Optimality in Imprecise Multi-Attribute Additive Models. Reliab. Eng. Syst. Safe 79 (2003) 253-262
9. Jiménez, A., Ríos-Insua, S., Mateos, A.: Monte Carlo Simulation Techniques in a Multi-Attribute Decision Support System. Proceedings of the 12th IASTED International Conference on Applied Simulation and Modelling, ACTA Press (2003) 85-90

Decision Support Systems for Portfolio Optimization

María A. Osorio Lama, Abraham Sánchez López

School of Computer Sciences, Universidad Autónoma de Puebla,
Ciudad Universitaria, Puebla, Pue. 72560, México

{aosorio, asanchez}@cs.buap.mx
http://www.cs.buap.mx/~aosorio

Abstract. We present the elements for Decision Support Systems for Portfolio Optimization, including the financial investment elements for a knowledge database and the elements for a Portfolio Optimization Model. We take into account the tax impact and the investor's scenario role into the models, and modify the classic Markowitz model, augmenting constraints that consider them and proposing stochastic linear and stochastic quadratic models. The efficient frontier is calculated using the augmented model. Finally we present the elements for an integrated knowledge and preference systems with the Optimization model.

1 Introduction

We are in the midst of a revolution in investment management. An unprecedented globalization of financial markets, advances in the electronic transmission of data, the accessibility of inexpensive yet extremely powerful computer hardware and software, and the migration during the past decade of so-called quants and computer wizards to Wall Street have all contributed to this revolution according to Aiken [1]. The body of finance theory and empirical evidence related to rational investment decision making has become so large that any future advances are expected to be incremental. Thus, the coming decades will likely bring a consolidation and accelerated application of this knowledge (see Doyle [2] and Min [3]).

The computerization of investment decision-making activities requires software systems that integrate mathematical models, a source of data, and a user interface. Such systems are generally referred to as decision support systems, or DSSs. That form of DSS whose database includes relevant theory, facts, and human knowledge and expertise is called knowledge-based system, also referred to as an intelligent system or expert system (ES). Specific techniques such systems employ to acieve their goals are drawn form the field of study known as artificial intelligence (AI) (see Pomerol [7]).

The renewed interest in applying knowledge-based systems to business decisions can be attributed mainly to the plummeting costs of hardware and software. Financial applications are now viewed as ideal proving grounds for new AI concepts and products, because in the realm of finance, significant, rapid, and easily measurable economic benefits are often possible.

© A. Gelbukh, R. Monroy. (Eds.)
Advances in Artificial Intelligence Theory
Research on Computing Science 16, 2005, pp. 13-22

Initially knowledge-based systems were viewed as tools to enable nonexperts to make decisions as effectively as would one or more experts in a particular field, or domain.

How would and intelligent system function in the portfolio investment domain? Consider the problem of how to allocate a large sum of money among stocks, bonds, real estate, and precious metals. Early ESs used for this purpose would probably have included in their data bases the knowledge of several human experts in each of these investment areas. Such a database is called a knowledge base. Today the knowledge base would be likely be built, at least in part, by a machine learning-based subsystems utilizing rule induction, genetic algorithms, or some other paradigm of learning automation. A money manager using such a system could, in theory, manage a portfolio including several asset classes more effectively and at less cost that could any of the individual domain experts. The most obvious advantages of integrating some form of computer intelligence into the portfolio decision-making process over continually consulting with a team of experts include permanence, usually a much lower cost, and a greater consistency of results.

We present the financial investment elements for a knowledge database in section 2, including definitions taxes impact and investor's scenarios. In part 3 we describe the elements for a Portfolio Optimization, taking into account the tax impact and the investor's scenario into the constraints in the model, modifying the classic Markowitz model, proposing stochastic linear and stochastic quadratic models. We also present the efficient frontier calculated with the augmented model. In section 4, we present the elements for an integrated knowledge and preference systems with the Optimization Model, and the conclusion and an example in section 5.

2 Financial Investment Elements for a Knowledge Database

We present the definitions and elements that should be present in the knowledge database, considering taxes ant the investor's scenario. They were first introduced in the models by Osorio et al [5, 6].

2.1 Definitions

Portfolio: A set of assets available for the investor.
Assets: The assets considered are Equities, Bonds, Cash and Properties in the United Kingdom or abroad, available for the constitution of a portfolio distribution.
Returns (Dividends or Income): Percentage of returns in the form of dividends for equities, bonds or properties and income for cash assets.
Capital Gains: Percentage of growth in the capital value of the assets included in the portfolio.
Net Redemption Value: Total amount of money received at the end of the horizon, when a wrapper is encashed, and taxes are paid.
Wrapper: A wrapper is a set of assets (Equities, Bond, Cash and Properties) with a specific set of rules for taxation on regular basis and in different investor scenarios.

2.2 Wrappers

Generally, wrappers can be divided on three general classes:

Offshore Bond: An offshore bond is a foreign non-income producing asset for a national investor. Life insurance companies in offshore centres usually offer offshore life insurance bonds. The major difference between national and Offshore insurance bonds is the location of the insurance company managing the funds and the tax treatment of the bond. The main advantage of an offshore bond is that the investment grows more or less free of tax, although the funds may suffer withholding taxes on their income. This is often knows as 'gross roll-up' and, over the long term, the compounding effect of this can make a substantial difference to the eventual overall return, despite the higher tax on final encashment.

Onshore Bond: Money is invested in a wide range of assets. These may include equities, bonds, cash and commercial property, the value of which will fluctuate. Onshore bonds have several important features as an investment medium. It is possible to switch from one asset to another without any personal capital gains tax charge arising. Usually tax on their income and capital gains is payable at a reduced rate equal to basic rate tax.

There may be capital gains tax advantages as the price of units in the funds allows for the future liability to tax on capital gains.

Unit Trust: Unit trusts offer an opportunity for capital growth with a much greater spread of risk than that usually associated with direct equity investment. A unit trust pools the resources of its individual investors and fund managers, and invests the total within a defined range of investment markets. The fund is divided into equal proportions called units, the price of which is normally quoted on a daily basis. Unit trusts normally pay dividends to investors twice a year.

The value of a unit trust is determined by the price of these units, which is calculated daily by reference to the value of the securities and other assets held in the fund. The value of those units can go down as well as up and investors may not get back their original investment.

There are general and specialized trusts and the permitted areas of investment will be set out in the trust deed. Funds can be geared to produce an income and/or capital growth. The type of units will determine how the dividends are received. With income units, they are paid out direct to the unit holder. With accumulation units the dividends are reinvested in the unit trust and contribute to the growth of the capital value of the funds held and thus increase the unit price. a specific set of rules for taxation on regular basis and in different investor scenarios.

2.3 Taxes

Taxes are payable in different ways for different wrappers and assets. Tax is evaluated in view of specific situations and the wrapper utilized. The main taxes applicable in this paper are:

Income tax: This tax is annually paid in the Unit Trust wrapper for the returns received that year.

Capital Gains Tax: Profits realized on the disposal of certain types of assets and wrappers are subject to capital gains tax. These include direct holdings of shares and property, as well as unit trust. However, in some countries, there is some mitigation of this tax in the form of taper relief. Taper relief is a gradual reduction of the amount of tax payable, dependent on the length of time an asset has been held.

Tax within bond: Tax charged within a fund. This is the case for the wrapper Onshore Bond where a tax charge arises annually within the fund, which provides the investor with a 'credit' for the basic rate of income tax.

Tax on Gross Returns: This tax is paid on the encashment of Onshore or Offshore Bonds wrappers. On Offshore bond, no tax arises on income or gains until exit, but on encashment, all income and gains are taxed at income tax.

Table 1. Type of Taxes

Wrappers	Annual	When Encashed	Withdrawals from
None	Income Tax	Capital Gains Tax	Income (NO) Capital Gains
Offshore Bond	No	Tax Offshore End	Allowed (NO) Returns (Tax Onshore)
Onshore Bond	Tax Onshore	Tax Onshore End	Allowed (NO) Returns (Tax Offshore)
Unit Trust	Income Tax		Income (NO) Capital Gains

2.4 Investor's Scenario

Investor events are the combination of different real life situations that require specific taxation rules and can affect the net redemption value of a wrapper. These events are:

Withdrawals: Withdrawals can be made anytime but are grouped in an annual basis in our model, for tax purposes. Withdrawals are split between the different wrappers and assets. In Unit Trusts, returns coming from dividends or income are subject to immediate tax and may be withdrawn without any further tax, but the capital gains are subject to capital gains tax.

Gift: A gift will not cause any taxation in Onshore and Offshore bonds but will incur in income tax and capital gains tax in Unit Trust.

Death and Inheritance: In the case of death with a testament, there is an inheritance tax on the total value of the wrapper in any wrapper. No Capital Gains tax is paid in wrapper Unit Trust, in the year of death.

Emigration: It only makes a difference in Offshore Bonds, where relief is given on a pro-rata basis for any time spent as non-national resident for tax purposes during the duration the bond is held as an investment, and there is a potential to 'empty' tax-free if the bond is encashed outside the nation after one year's non residence. tax.

Table 2. Taxes according to the Investor's Scenario

Scenario	None	Onshore Bond	Offshore Bond	Unit Trust
Gift	Tax on Dividends, Income, Capital Gains	No Tax	No Tax	Tax on Dividends, Income, Capital Gains
Death+ Inheritance	Inheritance Tax. No tax on Capital Gains	Tax on Gross Bond Returns + Inheritance Tax	Tax on Gross Bond Returns + Inheritance Tax	Inheritance Tax. No tax on Capital Gains
Withdrawals	Tax on capital gains for the withdrawal	A percentage deferred	A percentage deferred	Tax on capital gains for the withdrawal
Emigration	No Exception	No tax on Gross Bond Returns	No tax on Gross Bond Returns	No Exception

3 Elements for a Portfolio Optimization Model

From a global point of view, we should consider the elements shown in Figure 1 in a portfolio optimization model.

3.1 Constraints

Initial Allocation. At the beginning, the initial investment is distributed among instruments within each wrappers.

Cash Balance Equations. Subsequent transactions do not alter the wealth within the period in question. Therefore the following constraints specify to balance the portfolio for each wrapper at each node. These constraints basically balance the cash by reallocating money among assets within each wrapper. In this way, sales fund the purchase of other assets in the same wrapper.

Wealth Balance. The wealth for each wrapper is described by the wealth of the previous year plus the increased value of the assets in that wrapper after paying the corresponding annual taxes. The transaction also allows selling or buying assets within the same wrappers, with the corresponding transaction costs.

Cumulative Taxes. Cumulative tax is paid at the end of the investment horizon when the investment is encashed. At the beginning of the investment plant, the cumulative tax for each wrapper is assumed to be zero.

Cumulative Returns. Returns include the income or dividend after income tax and the capital gain tax are deducted.

Diversification Constraints. Any diversification restriction imposed by the investor or the bank's advice can be expressed by the percentage upper bounds for each asset within each wrapper.

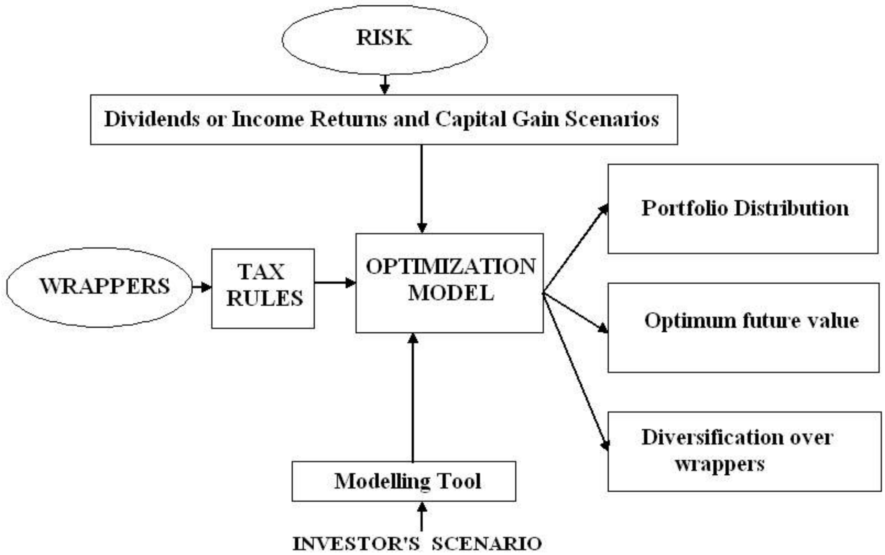


Fig. 1. Elements for a Portfolio Optimization Model

3.2 Uncertainty Representation and Scenario Trees

These models require a coherent representation of uncertainty. This is expressed in terms of mutivariate continuous distributions. Hence, a decision model is generated with internal sampling or a discrete approximation of the underlying continuous distribution. The random variables are the uncertain return values of each asset on an investment. The discretization of the random values and the probability space leads to a framework in which a random variable takes finitely many values. At each time period, new scenarios branch from the old, creating a scenario tree. Scenario trees can be generated based on different probabilistic approaches as simulation or optimization as presented in Gulpinar [4].

3.3 Multistage Optimization Models

Using the constraints described in 3.1 and the uncertainty in the model represented by scenario trees, we can define different optimization models. These models will all be multistage because they use the wealth generated in the previous period in order to

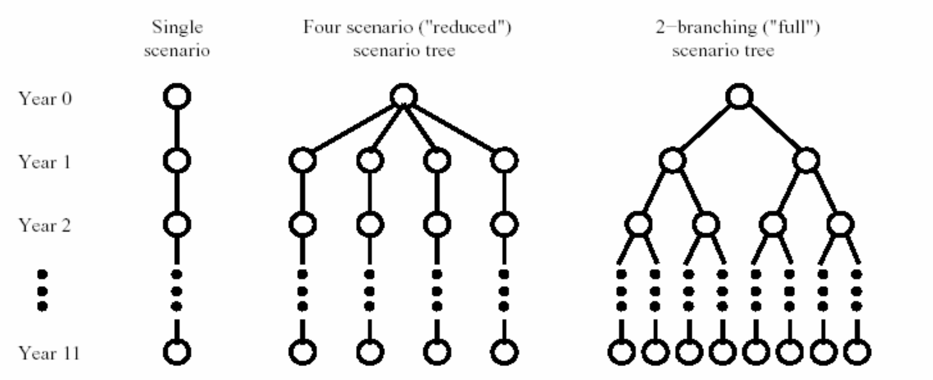


Fig. 2. Scenario Trees for Multiperiod Optimization

represent the constraint in the next period. If take into account the uncertainty of the assets return, based in the history of each asset and represent it in scenario trees models will be stochastic, otherwise they will be deterministic. According to the objective function, models generated under these concepts can be classified as:

A Stochastic Multistage Linear Programming Model. The expected post-tax wealth for each wrapper is calculated as net redemption value at the end of the investment period. In all time periods, taxes are accumulated and deferred, which must be paid on encashment of wrappers. The net redemption value for wrappers is then computed by deducting the accumulated taxes from the final portfolio. The total expected post-tax wealth at the last time period of the investment horizon is the sum of net redemption values of all wrappers and is the *mean*. Stochastic linear programming models which does not incorporate the quadratic variance term, computes the maximum expected return as the mean. This is the objective function for this model.

A Stochastic Multistage Quadratic Programming Model. Gathering together the variance terms of all wrappers, at all time periods will give the total variance in the model. Stochastic quadratic programming models with only wealth balance constraints were originally defined by Markowitz. This model can be enhanced considering all the constraints mentioned in the previous and risk in conjuction with the mean variance which is the objective function for this model (SQP).

Both models can be mixed if we allow the user to withdraw money from the investment's initial amount even instead of forcing the withdrawals only from the returns generated in the investment.

3.4 The Efficient Frontier

The Efficient Frontier is defined as the set of efficient points. Enlarging the universe of assets within wrappers from which the portfolio selection is made never results in a lower efficient frontier, since new securities can always be included at a level of zero. By including in a portfolio new assets whose returns are not highly positively correlated with those of other assets or wrappers, investors may develop significantly improved risk-return combinations. This explains the current trend toward global investing and the inclusion o real estate an other nontratidional assets in the portfolios of major institutional investors.

Which portfolio is chosen from among those on the efficient frontier will depend on the investor's utility function, which represents preferences with respect to risk and return. Different investors facing and identical efficient frontier are likely to choose efficient portfolios having at least somewhat differing levels of risk and expected return.

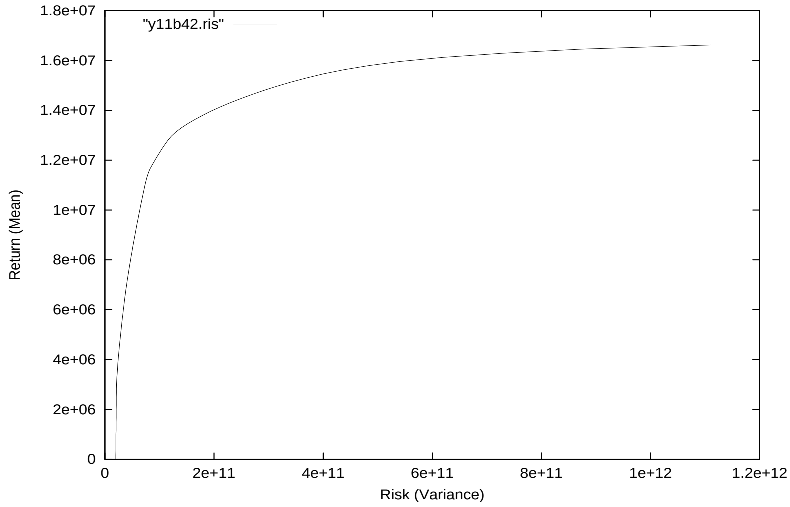


Fig. 3. Efficient Frontier: Mean vs Variance time.

4 Integrating Knowledge and Portfolio Optimization Models

Finally the elements in the portfolio optimization models and the knowledge databases can be integrated in the scheme showed in Figure 4. The scheme presented is similar to the scheme first introduced by Trippy and Lee [8], but instead of the prioritized variables he suggests, we generated additional constraints with enriched elements to manage a wide range of withdrawals in different wrappers for the portfolio elements considered, and took into account the investors' scenario first introduced by Osorio et al [5].

5 Conclusion

We will consider a real life case with the data in Table 3.

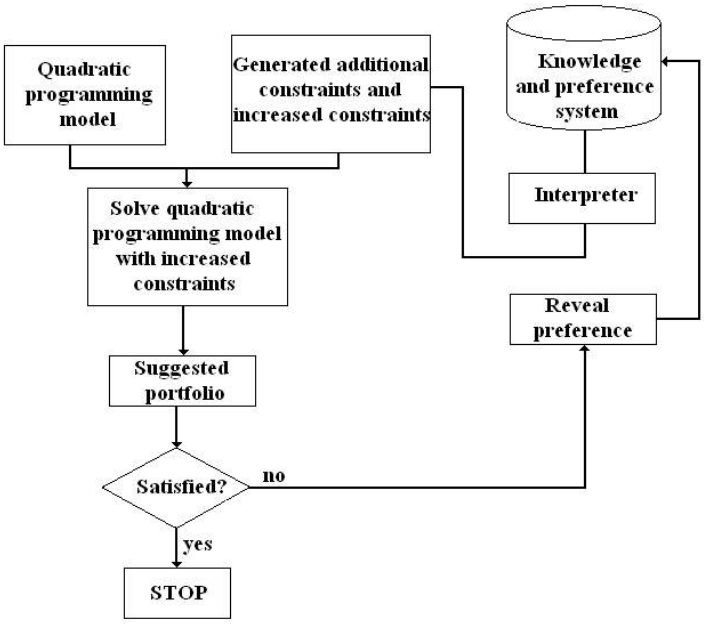


Fig. 4. Integrating Knowledge and Portfolio Optimization Models.

Table 3. Data for the example

YEAR	WITHDRAWAL	CGT	TAXES	%
1	£ 500,000	40%	Tax on Offshore Bond (end)	40%
2	£ 500,000	40%	Tax on Onshore Bond (annual)	22%
3	£ 500,000	40%	Tax on Onshore Bond (end)	18%
4	£ 500,000	38%	Income Tax (Cash)	40%
5	£ 500,000	36%	Income Tax (GBP Equities)	25%
6	£ 500,000	34%	*COSTS*	
7	£ 500,000	32%	Transaction Cost (all wrappers)	1%
8	£ 500,000	30%	Annual Cost (all wrappers)	1.15%
9	£ 500,000	28%	Initial Cost (all wrappers)	0%
10	£ 500,000	26%	*BOUNDS*	
11		24%	On all assets	43%

In this example, an investor sold a factory and would like to invest £8,000,000 and to live from this money the next eleven years. He would like to get an annual withdrawal of £500,000. How would he have to invest his money in order to maximize the money he will receive back after eleven years? To solve this question, we tested our approach using several structure for the trees. Table 4 shows the Net Redemption Value obtained with each and the number of constraints and variables that used each tree.

Table 4. Results for the example

MODEL	LP	MIP	LP
TREE	Reduced	Reduced	Full
NODES	44	44	4094
Constraints	1370	1610	112,574
Variables	3250	3730	300,654
Binaries	0	120	0
NRVW	11,324,784.76	12,201,189.91	11,459,678.32
CPU Secs	18.73	3925.84	63,424.32

As a conclusion we can say that optimal investments strategy generally entails a diversification over different assets. The more exact model is integer (MIP) and requires the solution of an LP in every node of the searching tree. The linear model only solves one LP. Our experiments show that the MIP model generally yields better returns and is therefore preferable despite the higher computational cost.

Acknowledgements. Research supported by SEP-BUAP-VIEP Project No. III 45-04/ING/G.

References

1. Aiken, M., Liu-Sheng, O., Vogel D.: Integrating Expert Systems With Gorup Decision Support Systems. ACM Trnasactions on Information Systems, Vol. 9, No. 1, January 1991, USA (1991) 75-95.
2. Doyle, J., Dean, T. et al: Strategic directions on Artificial Intelligence. ACM Computing surveys, Vol. 28 No. 4. USA (1996) 653-670.
3. Min, H., Eom, S.: An Integrated Decision Support System for Global Logistics, International Journal of Phisical Distribution & Logistic Management, Vol. 24, No. 1 MCB University Press USA (1994) 29-39.
4. Gülpinar, N., Rustem, B., Settergren, R.: Optimization and Simulation Approaches to Scenario Tree Generation. Journal of Economics Dynamics and Control, Vol. 28, Issue 7 (2004) 1291-1315.
5. Osorio, M.A., Settergren, R., Rustem, B., Gulpinar, N.: Post Tax Optimal Investments, Financial Engineering, e-Commerce ans Supply Chain. Kluwer Academic Publishers (2002) 153-173.
6. Osorio, M.A., Gulpinar, Settergren, R., Rustem, B.: Post-Tax Optimization with Stochastic Programming. European Journal of Operational Research, Vol. 157 (2004) 152-168.
7. Pomerol, J.C.: Tutorial on Artificial Intelligence and Human Decision Making, Euro XIV Conference on Operational Research, Jerusalem, Israel (1995).
8. Trippy, R., Lee, J.: Artificial Intelligence in Finance and Investing: state-of-the-art technologies for securities selection and portfolio management. IRWIN Professional Publishing, USA (1996).

Location-based Support for Commerce using Multiagent Negotiation

Ramón F. Brena, Luis M. Fernández, Jesús H. Domínguez, José L. Aguirre

{ramon.brena, A00789695, A00779374, jlaguirre}@itesm.mx
Center for Intelligent Systems
Tecnológico de Monterrey, Campus Monterrey
Monterrey, México

Abstract. Location-aware information systems is one of the most rapidly developing areas in IT. Location technologies, like GPS and cell-triangulation, offer to mobile devices possibilities for accessing information depending on its physical current location, giving the opportunity for new location-aware information services. In this paper we propose a method and a multiagent architecture for supporting impulse commerce (clients buy as they physically move through stores) by providing a negotiation brokering service. Mobile devices would act as personalized shopping guides for buyers, and merchants could "push" offers to clients, according to their actual in-stock merchandise and their sales policies. A basic working prototype illustrating our proposal is reported.

1 Introduction

Location-aware mobile information systems is one of the most rapidly developing areas in IT [1–3]. This is understandable because it combines both a need (to relate information systems with the physical world) and the technological feasibility for physically locating devices. Though location technologies are not mature enough to become mainstream, some of them are making their way to become both usable and economically practical. In particular, GPS devices and cell-triangulation technologies are being actively developed and make rapid progress [4]. We can expect in the near future the availability of inexpensive and widely available position-location mobile devices, possible integrated to other mobile devices like cell phones or PDAs. This would offer to mobile devices possibilities for accessing information depending on its physical current location, giving the opportunity for new location-aware information services.

While hardware location technology is advancing, the software counterpart is lagging behind. Though thousands of new useful services could benefit from location-awareness, just the most elementary of them have been developed, like restaurant location [5].

One of the most promising areas for location-aware information services is impulse commerce [5], where buyers wander through physical stores, eventually entering one of them to buy something. In this context, a mobile position-aware device, connected to an information system, could act as a "shopping guide",

© *A. Gelbukh, R. Monroy. (Eds.)*
Advances in Artificial Intelligence Theory
Research on Computing Science 16, 2005, pp. 23-32

providing the user information about the surrounding stores. This kind of information service will be for sure available in the short term, together with other location-associated services like hotels, restaurants etc., linked to a mapping service.

We consider extremely important the information personalization for specific users, giving them better focused -and thus more valuable- information. As we will see later in this paper, individual client profile management should be a key component of a useful location-aware information service.

But beyond static location-aware information services, we envision the possibility of more dynamic services, where a real-time interaction is possible between the stores (including restaurants, hotels, etc.) and the potential clients. These dynamic services would have several advantages over their static counterparts, as:

- It would be possible for the store to give information about actually available items, such as cloth sizes in stock;
- It would be possible for stores to propose special offers;
- More flexible pricing policies would be practical for stores, like reducing one price when buying another item;
- A higher degree of personalization would be possible for clients.

In this paper we propose a method and a multiagent architecture for supporting impulse commerce (clients buy as they physically move through stores) by providing a negotiation brokering service. Mobile devices would act as personalized shopping guides for buyers when they want so, and merchants could "push" offers to clients, according to their actual in-stock merchandise and their sales policies.

The structure of this paper is as follows: after this introduction, we present LOBSTER (LOcation-Based SupporT for commERce) in section 2; then a working prototype is reported in section 3 and, after, some comparison with related works in section 4. We conclude in section 5 reporting some final conclusions.

2 LOBSTER description

Imagine that a hypothetical John Smith has a cell phone with GPS-like location capabilities. John uses this device as a shopping guide, because it is connected to the LOBSTER service, which allows it to locate in real-time products he needs, according to his profile, and bargains as he travels by car through the city.

The mobile shopping guide shows on its screen a simplified, but physically accurate, map of vendors nearby, represented by special icons, presenting on screen just merchants fitting John's personal interests, stored in a *personal profile* that John uploaded to the system. This personal profile, which includes things as clothes sizes, food tastes, prices ranges, and many other preferences, is continually enriched with John's traveling habits and buying history, but John can choose to keep it private or either to deactivate it. The cell-phone like shopping assistant can work both in "pull" mode, when he asks, for instance, for the nearest chinese restaurant [1] and in "push" mode, where the merchant agent, who is alerted of the potential customer presence, advertises its product or even proposes a special deal. In his personal profile, John indicated whether he wants to be alerted about products (and which products), as well as special offers. In particular, John established this morning that he would not receive any advertisement from LOBSTER except if it is an offer about for digital cameras or shoes. In the personal profile it was established John's shoe size, so he will only be alerted of a store actually having his shoe size in stock at that particular moment. LOBSTER participating stores have a permanent connection with the system so they can give warranty about their actual merchandise in stock. John can make temporal profile modifications when, for instance he needs to find a gift for somebody else –who will of course have different clothes sizes, etc.

When John approaches a shoe store having his size and preferred style, his shopping assistant phone vibrates and shows on screen the store's location. Then, John can click on the store's icon to read the offer's details.

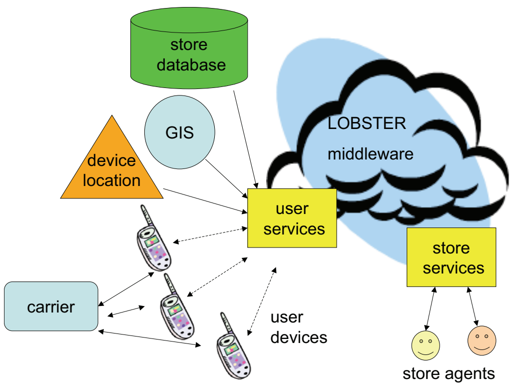


Fig. 1. Lobster's general architecture

As you can see from the above example, this vision requires much more than hardware location technologies. It requires an entire brokering service, which would be a middleware accessed both by the user's cell phone carrier, and by connected stores. We decided that it is necessary to provide clients and merchants the maximum negotiating capabilities and flexibility, and we think that this can be achieved by giving stores and clients their respective negotiating agents, taking advantage of agent technology's flexibility [8]. Those agents communicate with the LOBSTER middleware brokering service, which eventually manage costs of usage, both for stores and for clients. In figure 1 we show the main components of the LOBSTER service, including, of course, cell phones,

[1] This level of service has already been implemented, for instance in [6, 7, 4].

location hardware, a GIS system for storing and managing store's locations, and LOBSTER middleware, including client services and stores services.

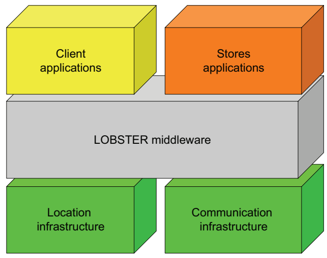


Fig. 2. Lobster's layered software architecture

In very broad terms, the system works as follows:

- The cellular phone sends its relative geographic position to a communication system (usually administered by the telecommunication company). The way in which this position is determined is not discussed here and it is assumed that it is already part of the service offered by the carrier.
- This position is received and the preferences, likes and dislikes the user may have are matched to those services, goods and store products, within a predefined range of action, that are being offered.
- Using this filtering process, sets of stores that comply with the requirements the user has are created. There are as many sets as products and services the user wants.
- With this sets, a negotiation process can take place, as long as the user's profile says so. Stores send their offers, based on their own policies, to the user. This offers are ranked and evaluated, setting up the utility that such values represent to the client and to the participant stores, so that the client can make a decision about what to do with them.
- If no decision is made then the stores have the option to send a better offer or do whatever their policies state.
- Finally, once a client has selected an offer, this one is saved by the system and the transaction is made for this specific product.

In order to fulfill these tasks, an agent-based architecture was designed implementing not only the basic agent architecture but also the rules for negotiation, evaluation and decision making in order to maximize utilities [9]. In the following sections such characteristics of our proposal are presented.

3 A multiagent Prototype

We have developed a working prototype for simulating the LOBSTER main ideas. This prototype is based on a multiagent architecture and has been implemented using a multiagent programming platform [10]. The multiagent architecture tries to simulate the real life conditions found in an everyday purchase action [4]. As such, there are four kinds of agents in our system, each one of them with a specific role and following a distributed paradigm [11]. In other words:

CellPhone Agent It represents the person in the negotiation process, using the preferences already established by the user. It also keeps track of the negotiations that go on and the geographic position where the user is. There are as many as registered users are.

Store Agent It represents the store that may evetually make an offer to the person who is within the range of work, following bargain policies according to the ones the store has. Similarly to the previous one, there are as many as registered stores are in the system.

Broker Agent It evaluates the offers on behalf of the user sending the information to the cellular phone for the person to make a decision. It also informs the stores about the results of the evaluation so that they can make better offers. This agent is only created once a negotiation process is instantiated and there are as many as products and interests the user has.

Server Agent Basically, it is in charge of the papework in the system. It matches the profiles the users have to those of the stores, creating as such the sets for the negotiation so it is the one that has direct access to the database of the system along with the tasks related to data maintenance. It also keeps track of the negotiations and their outcomes.

Name Agent	Instances		Roles and Tasks Assigned
CellPhone Agent	n	plays →	PositionInformer, PersonalProfileManager, PurchaseOpportunitiesController
Store Agent	m	plays →	StoreProfileManager, StoreNegotiator,
Server Agent	1	plays →	ProfilesManager, NegotiationInitiator, NegotiationCloser
Broker Agent	$\sum_{i=1}^{n} p_i$	plays →	UtilityRanker, NegotiationController

n	Number of cell phones running in the simulation
m	Number of stores running in the simulation
p_i	Number of interest products per cell phone i

Fig. 3. Lobster's Agent Model

In the proposed system these roles are represented in Figure 3. From this, it is not hard to see how the communication among agents and the flow of information goes; this is shown in Figure 4. It is important to notice that this represents the middleware, the software that would go under the machinery of the communication system. This is done so since such part of the system is the core for any further development in the field of value-added service in LBS technology [12].

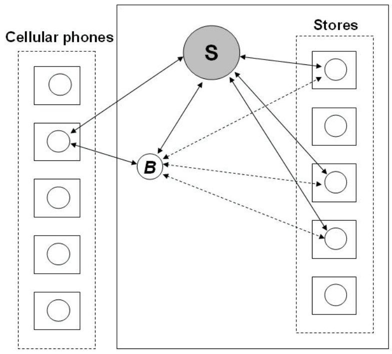


Fig. 4. Lobster's Architecture

3.1 Negotiation, Evaluation and Decision Making Processes

Since this report presents our first attempt to a working commerce-middleware; the negotiation, offer evaluation and decision making processes are quite simple. However, they are built in such a way that allow further development, enhancing as such the negotiation capabilities, offers filtering and decision making stages in an easy and scalable way.

In the present prototype all the above tasks are performed as described below:

Negotiation As mentioned earlier, the negotiation starts once that at least one store that matches the user's profile has been found; then, the *Broker Agent* is instantiated and waits for offers. This agent auctions "the client" asking for offers and, then, for even better ones. Every time new offers arrive, a new negotiation takes place since new proponents may appear; so new bids are received and evaluated for the user to make a better decision.

Evaluation It is performed by the *Broker Agent* using the offers received from the stores and normalizing them in order to give a score to every offer. Such scoring is sent to the user so that it is displayed on the cellular phone's screen, varying degrees of color to represent the evaluation given to each offer.

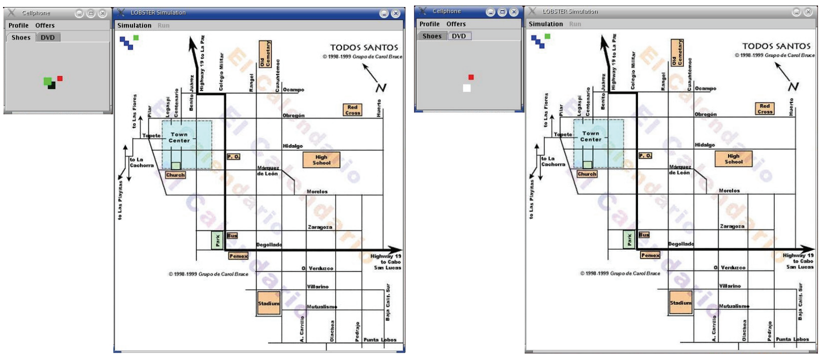
Decision Making The user, of course, is the one who makes the decision, choosing from continuing receiving more offers or accepting one to stop receiving more proposals. In any case, the outcome of the decision is stored in the database of the system and is administered by the server.

In the following sections, the results of executing this system in a simulated environment are presented and analized.

3.2 Prototype testing

In order to check the validity of our approach, a simulated environment was built. This environment represents any given city as shown in Figure 6(a) where the three squares in the up left position represent the stores and the one in the middle represents the user (i.e., the cellular phone). As explained earlier, these entities are running in the system as agents.

The *Cellphone Agent* sends its position to the *Server Agent* for it to match the profile the user has with those profiles of services and products stores have. If stores that match the user's profile are found, then the negotiation is begun and the process is displayed in the cellphone screen as shown in Figure 5(a) and 5(b); opening as many tabs as matched products are in the negotiation set.



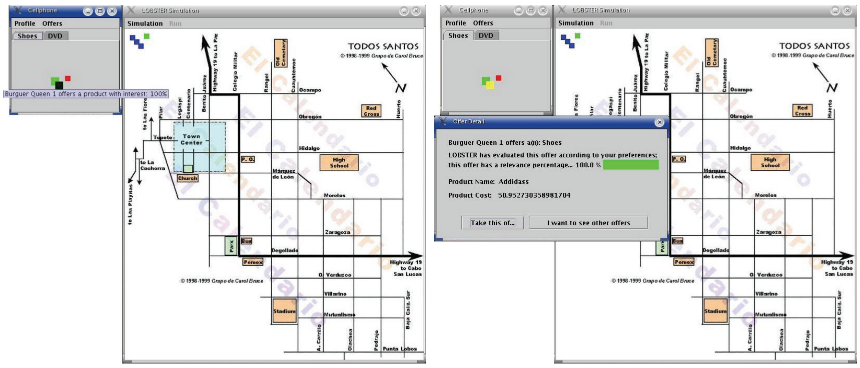
(a) Incoming Offers for Shoes (b) Incoming Offers for DVD

Fig. 5. Offers Received by the User

The relevance of these offers are represented in the cellphone screen varying the degrees of color, going from lighter to darker colors, and by messages on the screen, see Figure 6(a). The user can access this pictured information and see himself the complete message of the chosen product, 6(b).

The offers come in as the user moves adding any new store that fits the services and products list the user has. Every cycle a new set of evaluated (i.e., negotiated) offers is displayed on the screen for the user to check.

Moreover, and being this a dynamic system, the user can update the preferences he has by accessing the database (i.e., through the *Server Agent*); adding or deleting products or services, or changing his degrees of interest for each entry. The capability of updating profiles is also added to the stores that are in the



(a) Simple Displayed Information (b) Complete Displayed Information

Fig. 6. Displayed Offers' Information

system (i.e., *Store Agents*), by changing the policies, proper of the store, that administer the offers and the negotiation procedures. All these database utilities are done in real time and considered for future negotiations in the system.

Furthermore, new and more participants, both clients and stores, can be added to the environment and, thus, be part of a future negotiation by setting each agent's profile through the user's interface. Finally, and for simulation purposes, the system offers extra tools that help visualize the system running, such as running the agents a chosen number of times, following a step-by-step execution or setting the speed in the system.

The observed results after running the system showed that our approach offers not only opportunities to the person (i.e., client) but also to the store. This is accomplished thanks to the negotiation in which agents engage once an opportunity is found. In general, a more personalized service is what has been added to the everyday purchase process; following the traditional bargain that is applied by humans, though. This feature makes the negotiation a powerful tool in the system since allows people pay the true value of what they are looking for and makes stores set up policies where client-oriented services are of high importance [13].

The agent-based brokering proved to be a powerful tool, providing information that many times is hidden to the normal person. This new information helped user (i.e., buyers) make a faster decision when deciding what to do about an item. Moreover, the system allows people to cancel any ongoing negotiation and set his agent *off* when he may not be interested in anything.

4 Related work

This paper relates to work on location-based services in general, and specifically work within learning personal agents with wireless devices [14], and e-commerce applications [15]. Location-aware applications are becoming increasingly popular as a consequence of the growing availability of consumer-oriented wireless networks and the emergence of locating technologies [8]. This fact is increased because of the wide range of services that rely on users' location information [3]. As such our paper is also related to these ones differing mainly that not only location is taken into account [6] but also the profile users and stores keep in the database.

There are numerous ways to exploit location to provide more relevant information, or derive new services. It can be particularly powerful when combined with other user profile information to offer personalized and location sensitive responses to customers [3], distinguishing between emergency services, mobile network operator services, and value-added services (VAS), focusing on the latter category as the primary e-commerce opportunity. It is in the VAS category that our work is in; however, differing [8] in the sense that our system is open to add a multi-attribute store profile and users' preference list enabling multiple negotiations.

Compared to the above mentioned proposals, our system provides:

- A *push* mode for bargains and special offers, which complements the more widespread "pull" proposals;
- A flexible negotiation architecture, which gives to individual stores the possibility of programming original negotiation strategies;
- Highly personalized services for the clients.

5 Conclusions

We presented an agent-based approach based on a middleware capable of brokering merchants –which are connected to the system in real-time– and clients –who can access in a highly personalized way the available offers from a mobile location-aware device.

We presented a first prototype of LOBSTER, in which the lower software and hardware layers are simulated, but which is useful for illustrating the capabilities of our architecture.

Being this our first prototype, there are still some things left to be done to enhance the capabilities of the Lobster System. Basically, these tasks are:

- Add more filtering characteristics in the user profile when looking for a product or service. This features may include cost, color, fabric, size, etc.; and personal settings such as age, gender, etc.; allowing the use of ontologies.
- Add more degrees of interest, in other words, the user should be able to choose intermediate levels of interest when setting up his profile. This will allow the "learning" of the profile by the system as time goes by.

– Using the previous features, create a more integrative utility function where this aspects are considered when ranking the offers and getting the highest *utility* for the client and the store. (e.g., finding an equilibrium among parts)
– Implement this system so that it can operate in a mobile device (i.e., cellular phones).

Acknowledgements

This work was supported by the Monterrey Tech's Research Grant CAT011.

References

1. Tsalgatidou, A., Veijalainen, J., Pitoura, E.: Challenges in mobile electronic commerce (2000)
2. O'Hara, K., Perry, M.: User centred opportunities for supporting consumer behaviour through handheld and ubiquitous computing (2003)
3. Steinfield, C.: The development of location based services in mobile commerce. Elife After the Dot.Com Bust (2004) 177–197
4. Varshney, U., Vetter, R.: Mobile commerce: Framework, applications and networking support. Mobile Networks and Applications **7** (2002) 185–198
5. Youll, J., Morris, J., Krikorian, R., Maes, P.: Impulse: Location-based agent assistance (2000)
6. Fano, A.E.: Shopper's eye: Using location-based filtering for a shopping agent in the physical world. In: Proceedings of the 2nd International Workshop on Autonomous Agents. (1998)
7. Youll, J., Morris, J., Maes, P.: Impulse: Location-based agent assistance. In: Proceedings of the 4th International Conference on Autonomous Agents. (2000)
8. Tewari, G., Youll, J., Maes, P.: Personalized location-based brokering using an agent-based intermediary architecture. Decision Support Systems **34** (2003) 127–137
9. Holtkamp, B., Gartmann, R., Weissenberg, N., Han, Y., Li, G.: Situation-aware service selection from large offers. Technical report, Fraunhofer ISST (2004)
10. Bellifemine, F., Poggi, A., Rimassa, G.: Jade - a fipa-compliant agent framework. In: Proceedings of PAAM99, London. (1999)
11. Rieger, A., Cissée, R., Feuestack, S., Wohitorl, J., Albayrak, S.: An agent-based architecture for ubiquitous multimodal user interfaces. In: Proceedings of the 2005 International Conference on Active Media Technology. (2005)
12. Loke, S.W., Zaslavsky, A.: Integrated ambient services as enhancement to physical marketplaces. In: Proceedings of the 37th Hawaii International Conference on System Sciences. (2004)
13. Hermann, F., Bieber, G., Duesterhoeft, A.: Egocentric maps on mobile devices. In: Proceedings of the 4th International Workshop on Mobile Computing - IMC 2003. (2003) 32–37
14. Rao, B., Minakakis, L.: Evolution of mobile location-based services. Communications of the ACM **46** (2003) 61–65
15. Ratsimor, O., Korolev, V., Joshi, A., Finin, T.: Agents2go: An infraestructure for location-dependent service discovery in the mobile electronic commerce environment. In: Proceedings of the 1st International Workshop on Mobile Commerce. (2001) 31–37

Ontologies for Student and Domain Models in Adaptive and Collaborative Learning System

José M. Gascueña, Antonio Fernández-Caballero, Pascual González

Laboratory of User Interaction and Software Engineering (LoUISE)
Computer Science Research Institute
University of Castilla-La Mancha, Albacete, Spain
{jmanuel, caballer, pgonzalez}@info-ab.uclm.es

Abstract. Any educational environment must use different sources of information such as the Student Model and the Domain Model to be able to adapt learning to the individual characteristics of every student. The fundamental components of these models are the learning style of the student and the learning objects respectively. In addition to learning individually, the student can also increase his knowledge collaborating with others. This way, it turns out to be useful to contemplate in the Domain Model activities that need the participation of more than one student. In this article the characteristics that we include in the Student Model and in the Domain Model using both ontologies are described.

1 Introduction

The educational community has not made use yet sufficiently of the new information and communication technologies to advance substantially. It is habitual to find subjects in the educational environments in which the profile of every student is not born in mind; that is to say, they present the same materials and activities to all the students. On the other hand, there is few possibility of re-using the educational materials due to their little granularity.

The Semantic Web is an initiative of the W3C, which pretends to construct a new web architecture that not only provides support to store the contents, but also associates a formal semantics. A learning/teaching environment may be considered a portion of the web where ontologies are an adequate technological approach to include semantic information on the contents taught. Persons (students and professors) and agents could communicate on this semantics basis. Agents reason about the contents and reproduce them in an intelligent way according to the users' needs.

Ontologies are useful in any environment, and especially in educational environments, as they enable people and/or software agents to share a common understanding of the knowledge structure. Moreover, they permit to reuse knowledge, that is to say, it is not necessary to develop an ontology from zero if we find another ontology that is available for use in the modeling of the current domain.

© A. Gelbukh, R. Monroy. (Eds.)
Advances in Artificial Intelligence Theory
Research on Computing Science 16, 2005, pp. 33-42

In the literature several attempts exist to classify learning objects. Wiley proposes a taxonomy that classifies the learning objects according to their possible combinations and their characteristics in terms of the number of elements, reusability and dependency grade [18]. This taxonomy defines five types of learning objects: single-type, combined-intact, combined-modifiable, generative-presentation and generative-instructional. Learning objects also can qualify according to their pedagogic use in instruction, collaboration, practice and assessment objects [2]. In these works no standard language is used to show these taxonomies in form of an ontology. Redeker proposes an educational taxonomy for learning objects (course, partial course, learning and knowledge unit) for the facilitation of generic sequencing strategies [11]. Silva describes an ontology-based metadata to achieve personalization and reuse of content in the AdaptWeb project [13]. DAML+OIL language is used to represent the ontology. Ronchetti and Saini [12] propose an architecture to aid students to find materials that present different points of view or different ways to explain concepts, but they does not make use of Semantic Web technologies.

At present, the Semantic Web is acquiring a big importance, being education one of its fields of application. In the Semantic Web, one of the main purposes is to represent the knowledge container within the Web resources in order to make it available for applications. Ontologies can be used for gathering the above mentioned knowledge, defining formally the concepts and relations that appear in the application domain. In our case, we create ontologies of the Student Model to represent the characteristics that define the profile of the student and of the Domain Model, which reflects the structure of the educational content, in which the fundamental components are the learning objects. This outline constitutes an interesting approach as it permits a uniform treatment of the problem that facilitates the interaction between both models. This way, a software agent might gain access to both ontologies to select and to show to the student the elements of the material educational that better fit to his profile. To represent the ontologies the Web Ontology Language (OWL) [19], the last standard language proposed by the W3C to represent ontologies in the Web, has been used, and the Protégé 3.0 framework [10] has been selected to edit / construct them.

Both ontologies should allow distance learning of subjects across the web, with the intention to reach two objectives. First, the subjects should adapt to the needs and peculiarities of every student. Second, learning objects should be defined with the sufficient granularity so that they could be re-used in different subjects. To achieve these aims, we model the profile of the student and his knowledge, as well as the knowledge of the educational domain according to this profile. The learning style of the student is one of the fundamental elements that constitute the profile of the student, since it determines what his preferences are in his learning process. In particular, we use the Felder-Silverman Learning Style Model (FSLSM) [3] to recognize the typical learning styles of the students.

The article is structured as indicated next. In section 2 we define what a learning object is and what are its main characteristics. In section 3 the concept of learning style is defined, with a special emphasis on the FSLSM. In sections 4 and 5 the Student Model and the Domain Model ontologies, respectively, are introduced. Finally some conclusions are offered.

2 Learning Object Definition

At present there is no definition commonly accepted of the term learning object inside an educational environment. For major confusion, in addition to the different definitions that we can find, there is no agreement on the term used to describe it (learning resource, educational object, information object, reusable information object, etc.) [7]. The definition of the LOM working group is too general [5].

We only consider in our research resources that may be transmitted through the Internet. Thus, Wiley's definition is the more appropriate one to our objectives. Wiley defines a learning object as "any digital resource that can be reused to support learning" [17]. The proposed definition is sufficiently narrow to define a reasonably homogeneous set of things (reusable digital resources), and it explicitly rejects non-digital and non-reusable resources. This definition includes anything that can be delivered across the network on demand, be it large or small (text, images, audio, video, animations, applets, entire web pages that combine several media types, and so on). Nevertheless, in spite of the different interpretations on what learning objects are, most educators would agree that learning objects incorporate the following features [8]: a) s*elf-contained* - each learning object is self-contained and can be used independently of other learning objects, b) *reusable* - learning objects are reusable and the same learning object can be used in multiple contexts for multiple purposes, c) *can be aggregated* - learning objects can be grouped into larger collections of content to create more substantial learning units, and, d) *tagged with metadata* - all learning objects are tagged with metadata that describes the learning object and allows it to be easily retrieved in a search.

3 Learning Style Definition

A learning style is defined as the unique collection of individual skills and preferences that affect how a person perceives, gathers, and processes information. A learning style affects how a person acts in a group, learns, participates in activities, relates to others, solves problems, teaches, and works [6]. We have chosen the FSLSM so that it is a part of the profile of the student because it has been used successfully in many computer-based educational systems (e.g., [9], [4], [16], and [1]).

Table 1. FSLSM learning dimensions

Definitions	Dimensions		Definitions
Do it	Active	Reflective	Think about it
Learn facts	Sensing	Intuitive	Learn concepts
Require pictures	Visual	Verbal	Require reading or lecture
Step by step	Sequential	Global	Big picture

As shown in Table 1, the FSLSM [3] distinguishes four dichotomous dimensions to learning styles (active/reflective, sensing/intuitive, visual/verbal, sequential/global), which gives place to sixteen combinations of learning styles. The student answers to a questionnaire called Index for Learning Styles (ILS) [14], which enables to determine his learning style.

4 Ontology for the Student Model

In the Student Model ontology the knowledge about the student is represented, that is to say, which is his profile and how he is acquiring the domain knowledge taught. This ontology, showed graphically in Figure 1, consists of the central class *Student* whose properties describe the personal information (name, surname, identification card, e-mail, birth date and sex).

Property *hasLearningStyle* indicates the learning style that the student has. The dimensions of the FSLSM have been represented as four properties of type *integer* in class *LearningStyle*. The properties *hasLearningGoal* and *learnsUnit* point at the subjects (class *Unit*) that the student wants to learn and to the subject that he is studying at present, respectively. The property *knowsSection* serves to indicate the sections that the student already knows. The classes *Unit* and *Section* are defined in the ontology of the Domain Model. The property *hasNC* reflects the type of network connection that the student has in a learning session. To know the speed of the student's network connection can help to decide what objects are most adapted to show the educational contents. For instance, if there is a slow connection it will not be adequate to show elements that are hard to download to the client computer.

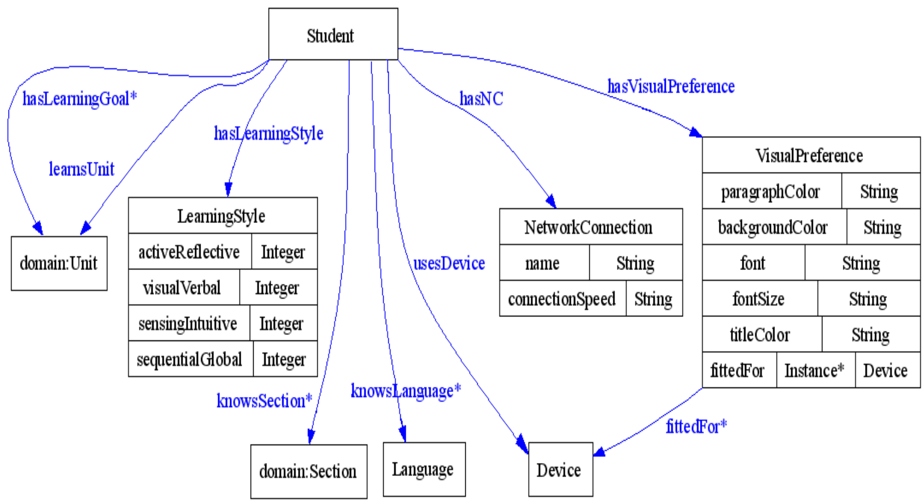


Fig. 1. Ontology for the Student Model

The property *hasVisualPreference* reflects how the student likes the pages to be shown (colours and characteristics of the letter, basically). The property *fittedFor* indicates that the visual preferences chosen by the student must be supported for the device used (PDA, standard monitor, etc). The property *usesDevice* indicates the device that the student uses to study the subject. These three properties will allow adapting the content in accordance with the visual preferences and the device in particular that the student uses.

The colour has been included as an important subjective distinctive feature of the profile of the student. It allows the user to process the information of a web page in a more effective way and simultaneously to increase his memorisation capacity. The properties *backgroundColor*, *titleColor* and *paragraphColor* in class *VisualPreference* reflect the background colour of the pages, the colour of the headers and words that are highlighted in the text, and the colour of the text of the paragraphs, respectively. On the other hand, the properties *font* (values Arial, Verdana, Courier, Times New Roman, etc.), and *fontSize* of class *VisualPreference* they are used to reflect the type and size of the used font.

Finally, property *knowsLanguage* of class *Student* reflects the languages that the student knows and dominates.

5 Ontology for the Domain Model

First we present the field to which the subjects belong. Our domain is framed inside the university context, although it might adapt easily to any other educational environment with a different organizational structure. As shown in Figure 2, a University is structured in departments (property *hasDepartment*), and every department is divided into a series of knowledge areas (property *hasKnowledgeArea*).

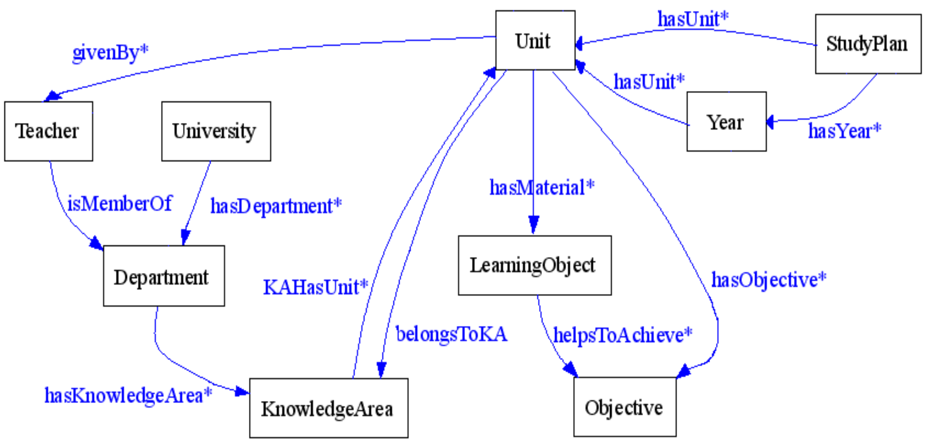


Fig. 2. Organizational structure

Every subject (*Unit*) belongs to a knowledge area of a department (property *belongsToKA*) and can be taught by several teachers of a department (property *givenBy*). In every unit a set of objectives have to be reached (property *hasObjective*). A collection of learning objects constitutes the material in order to teach the subject (property *hasMaterial*). Every learning object helps to reach a few objectives (property *helpsToAchieve*).

To describe the learning objects we will use metadata. As you may observe on Figure 3, a learning object (a) is created by one or several authors (property *createdBy*), (b) it has a set of key words that describe it (property *hasKeyword*), (c) it needs a certain network connection speed (property *requiresNC*), (d) it helps to reach a few objectives (property *helpsToAchieve*), (e) it may be visualized in certain devices (property *requiresDevice*), (f) it is located in a certain direction, and, (g) it is written in a given language -this way, we can locate the same object in several languages- (property *isLocated*). Also, a learning object has a description (property *description*), a type of interactivity -it can take values active, exhibition and mixed- (property *InteractivityType*) and a grade of difficulty -very easy, easy, average, difficult, and very difficult- (property *difficultyLevel*). The type *active* of interactivity applies for documents in which it is needed that the student interacts and/or performs operations (for example, simulations, exercises, test questionnaires), whereas *exhibition* is applied to documents whose objective is that the student gets the content (for example, text, images, sound).

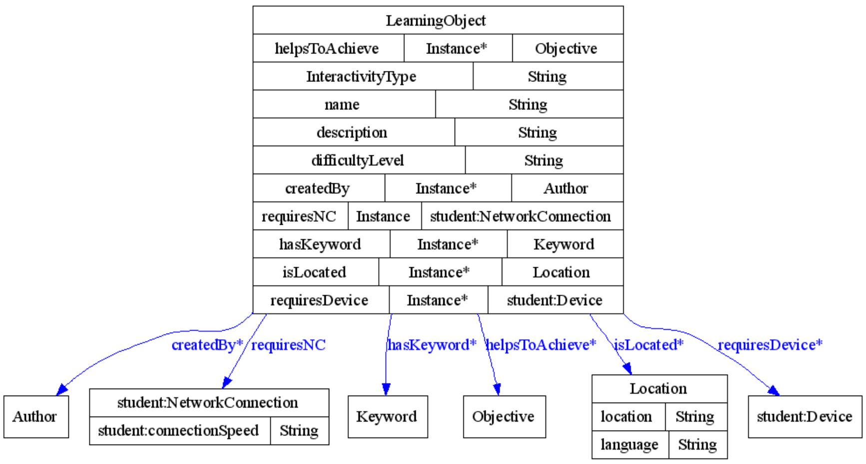


Fig. 3. Description of a learning object

To determine the grade of granularity of a learning object is a fundamental decision in any project. To obtain a high grade of reusability of a learning object is largely a function of the grade of granularity of the objects. That is to say, the more granular an object, the more reusable it will become [15]. Granularity is related to the relative size of the objects. For example the grade of granularity of a topic is minor than that of a section. To our judgment, we must fragment the educational contents into very little connected elements. In other words, we consider every element that can be a part of a page -text, images, video, and so on- has a meaning for itself and may be reused on different pages. Hence, if there are several elements with the same meaning we will be able to show the one that better fits to the preferences of the student.

Thus, we consider three types of learning objects of different granularity: *Topic*, *Section* and *Support* (see Figure 4). A topic has a thick granularity and it is formed by a set of sections (granularity increases). Sections are explained based on a collection of support material (granularity increases), choosing in every moment the objects that better adapt to the preferences of a beginner student. Thus we can reuse learning objects at three different levels, that is to say, we can reuse an entire topic, sections of a topic or fragments that explain a section.

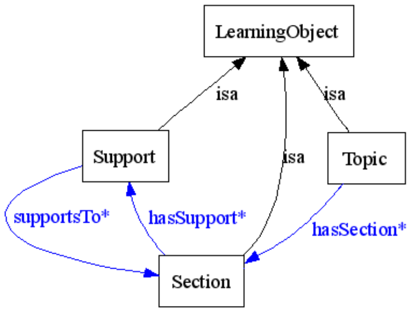


Fig. 4. Types of learning objects

The classes *TheoreticalExplanation* and *PracticalExplanation* represent the theoretical and practical explanations, respectively, that appear to the students (see Figure 5).

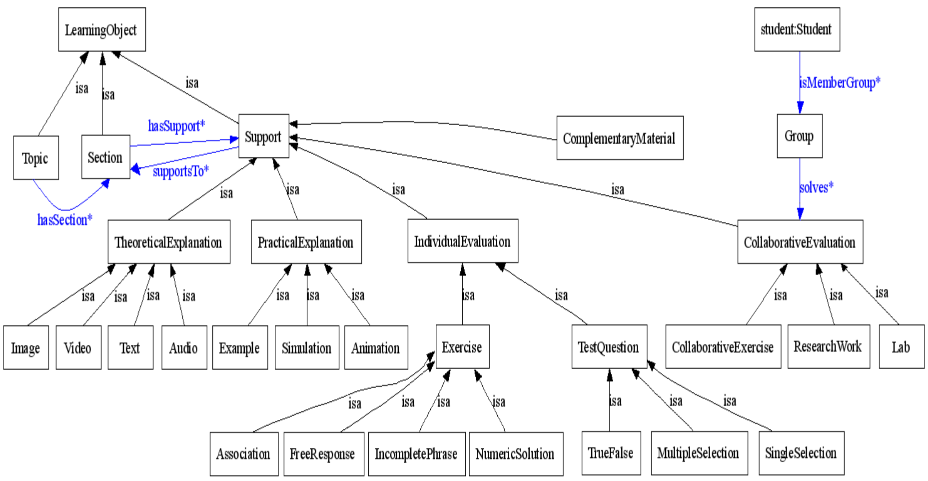


Fig. 5. Environment of class *Support*

This distinction allows showing to the sensory students first the practical applications of the theory and later the purely theoretical contents -and vice versa for the intuitive students. To compose the theoretical explanations several types of formats there are proposed (classes *Text*, *Audio*, *Video*, *Image*). This way, the theoretical explanations that appear to the verbal students are formed by text and/or audio, whereas videos and images are shown to the visual students. In order to realize practical explanations, examples, simulations and animations (classes *Example*, *Simulation*, *Animation*) can be used.

The individuals of classes *IndividualEvaluation* and *CollaborativeEvaluation* are used to evaluate the knowledge acquired by the student. The class *IndividualEvaluation* contains the exercises (class *Exercise*) and the test questionnaires (class *TestQuestion*) that a student has to solve without help of other partners, whereas the works (class *ResearchWork*), practices (class *Lab*) and some exercises (class *CollaborativeExercise*) of class *CollaborativeEvaluation* may be performed between several students.

The class Exercise contains the four following subclasses:

– *NumericSolution*. A statement in which a set of information are included is presented, and a question is posed so that the student has to answer with a numerical solution using the starting information.
– *IncompletePhrase*. A statement that the student has to complete in one or more points with a phrase, a word or a cipher is presented. There are two types of statements: one in which a list of elements is provided and the student must place the appropriate element in the correct place of the statement; and another in which the student must complete the statement without any type of help.
– *Association.* It consists of presenting a series of elements in two parallel columns where every word, symbol or phrase of a column can collaborate with the elements of another column. The student must establish the relations between the elements of two columns.
– *FreeResponse*. A statement appears in the form of a question and the student has to answer with one or several paragraphs. This type of exercises cannot be corrected automatically.

The class *TestQuestion* has several subclasses to highlight the different types of test questions that are shown to the student. They are the following ones:

– *TrueFalse*. The student has to choose among one of two alternatives.
– *SingleSelection*. A question of multiple elections with one single response. A situation or a problem and several options that provide possible alternatives are presented. There is only one valid option. A variant of this type of questions is that all options are valid but there is one that is better than all others.
– *MultipleSelection*. A question of multiple elections with multiple responses. A situation or a problem and several options that provide possible alternatives are presented. There are several valid options and the student has to select all the correct ones.

In addition to learning individually, a student can also increase his skills by collaborating with others. That is to say, learning in addition to being an individual process is also a social process. By means of the collaborative work it is possible to learn to work with others. To work of in a cooperative way allows individual knowledge to serve as a knowledge resource for all the components. This is why class *CollaborativeEvaluation* has been introduced as a new type of learning object. The property *isMemberGroup* of class *Student* indicates in what groups the student is included to realize the activities that need the participation of more than one student. The property *solves* of class *Group* indicates the problems that a certain group of students has to solve.

We can distinguish among several types of objects of collaborative evaluation. The class *CollaborativeExercise* represents the exercises that are proposed to several students to be solved in group. It turns especially out to be interesting to propose an exercise to a group in such a way that a part of the components of the group generate arguments in favour of the raised question whereas the rest generate arguments in against. Other examples of collaborative activities are the research works (class *ResearchWork*) proposed to the students so that they study in depth a topic, and the laboratory practices (class *Lab*) proposed to apply the skills and knowledge acquired in a real case.

Finally, class *ComplementaryMaterial* is added as a subclass of class *Support* to represent the additional materials accessible to the students that want to learn more.

6 Conclusions

The selection of the learning objects most adapted for every student is one of the interesting topics that are investigated in educational systems. The above mentioned selection is realized principally bearing in mind the students learning styles and preferences.

Another question to bear in mind is the reuse of learning objects, as creating teaching material is a task that needs a lot of time. To achieve a major reuse we think that the learning objects created must be as small as possible (image, video, text, audio, and so on), and from the smaller fragments more complex objects (sections, topics) should be constructed.

The new generation of the Web, the Semantic Web, appears already as a promising technology to provide the Web contents of meaning. The ontologies constitute one of the principal tools of the Semantic Web to specify explicitly the concepts of a concrete domain, its properties and its relationships. The OWL language has been used to represent the Student Model and the education Domain Model.

Acknowledgements

This work has been financed partially by the Junta de Comunidades de Castilla-La Mancha PBC-03-003 and the Spanish CICYT TIN2004-08000-C03-01 projects.

References

1. Aase, M., Kurfess, F. (2004) Utilizing learning styles for interactive tutorials. Proceedings of the 4th IEEE International Conference on Advanced Learning Technologies, ICALT 2004, 828-830.
2. ASTD & SmartForce (2002). A Field Guide to Learning Object. http://www.learningcircuits.org/NR/rdonlyres/17DB7DB7-0E67-480F-BA5C-4D0A2336529D/1091/smartforce.pdf
3. Felder, R.M, Silverman, L.K. (1988). Learning and teaching styles in engineering education. Journal of Engineering Education, 78(7), 674-681.
4. Hong H., Kinshuk. (2004). Adaptation to student learning styles in web based educational systems. In L. Cantoni & C. McLoughlin (Eds.) Proceedings of ED-MEDIA 2004 - World Conference on Educational Multimedia, Hypermedia & Telecommunications, AACE, 491-496.
5. IEEE 1484.12.1-2002, Draft Standard for Learning Object Metadata. http://ltsc.ieee.org/wg12/files/LOM_1484_12_1_v1_Final_Draft.pdf
6. Johnston, C., Orwig, C. (1999). Your learning style and language learning. http://www.sil.org/lingualinks/languagelearning/otherresources/yorlrnngstylandlngglrnng/YorLrnngStylAndLnggLrnng.htm
7. McGreal R. (2004). Learning objects: A practical definition. International Journal of Instructional Technology and Distance Learning, 1(9). http://itdl.org/Journal/Sep_04/article02.htm
8. Millar G. (2002). Learning objects 101: A primer for neophytes. Published by the BCIT Learning and Teaching Centre. http://online.bcit.ca/sidebars/02november/inside-out-1.htm
9. Peña, C.I., Marzo, J.L., De la Rosa, J.Ll., Fabregat, R. (2002). Un sistema de tutoría inteligente adaptativo considerando estilos de aprendizaje, IV congreso iberoamericano de informática educativa, IE2002, Vigo (Spain)
10. Protégé 3.0. http://protege.stanford.edu/
11. Redeker, G. (2003). An educational taxonomy for learning objects. Proceedings of the 3rd IEEE International Conference on Advanced Learning Technologies, ICALT 2003, 250.
12. Ronchetti, M. Saini, P. (2004). Knowledge management in an e-learning system. Proceedings of the 4th IEEEInternational Conference on Advanced Learning Technologies, ICALT 2004, 365-369.
13. Silva Muñoz, L. (2004). Ontology based Metadata for e-learning Content. Master Thesis. Federal University of Rio Grande do Sul, Porto Alegre, Brasil.
14. Soloman, B.A., Folder, R.M. (2005). Index of Learning Styles Questionnaire. http://www.engr.ncsu.edu/learningstyles/ilsweb.html
15. South, J.B., Monson, D.W. (2000). A university-wide system for creating, capturing, and delivering learning objects. Section 4.2 of the book The Instructional Use of Learning http://www.reusability.org/read/chapters/south.doc
16. Stash, N., Cristea, A., De Bra, P. (2004). Authoring of learning styles in adaptive hypermedia: Problems and solutions. Proceedings of the 13th International World Wide Web Conference, 114-123.
17. Wiley, D.A. (2001). Connecting learning objects to instructional design theory: A definition, a metaphor, and a taxonomy. In D. A. Wiley, ed., The Instructional Use of Learning Objects: Online Version. http://reusability.org/read/chapters/wiley.doc
18. Wiley, D.A. (2000). Learning object design and sequencing theory. PhD Dissertation. Department of Instructional Psychology and Technology. Brigham Young University
19. World Wide Web Consortium. (2004). “OWL Web Ontology Language Overview”.

Information Systems and Data Mining

Shallow Parsing and Information Extraction

Diego Uribe

Departamento de Ingenierías
Universidad Iberoamericana-Laguna
diego.uribe@lag.uia.mx

Abstract. From the set of modules that a full IE system comprises, the component to deal with the definition of text patterns for the identification of specific relational information has captivated the attention of researchers. Much research has been carried out in the context of subject/object relationships for the identification of the protagonists involved in target text events. Relatively little has been reported on learning rules for text extraction based on chunks as the main syntactic unit. In order to analyze the plausibility of an intermediate granularity such as chunks, our model implements a rule representation which has proved to be both effective and flexible. Regardless of the number of parameters (slots) the event represents, the rule representation is able to denote the local context which surrounds the target fillers. Experiments conducted in two different domains give evidence of the adaptability of the model and linguistic analysis of the corpus was carried out in order to illustrate how IE is a domain-specific task. We demonstrate how not only tuning the semantic lexicon to a new scenario plays a key role in the adaptation process but also how the syntactic analysis gets into trouble when the partial parser is confronted by a specialized corpus.

1 Introduction

Information Extraction (IE) is the activity concerned with the extraction of specific information through skimming natural language text. Actions or events and their protagonists commonly known as entities describe this target information. The basic architecture of this activity is made up of three modules: preprocessing, syntactic analysis and domain analysis [1]. While the first two modules are domain independent, that is, the preprocessing and syntactic analysis of the text can be implemented regardless of the concerned domain, the last module, domain analysis, is a module that draws on the knowledge which surrounds a particular area.

An important aspect in the learning process for the automatic detection of particular events from plain text documents is the rule representation. A rule representation based on subject-verb-object relations is a common scenario to facilitate the identification of the main protagonists in the target event. However, a more fine-grained way of representation such as noun groups and verb groups alters the search space, so that a suitable similarity metric and generalization procedure are demanded by the learning mechanism. In this paper, we approach

© A. Gelbukh, R. Monroy. (Eds.)
Advances in Artificial Intelligence Theory
Research on Computing Science 16, 2005, pp. 45-54

this problem through string similarity. Specifically, we turn this problem into the *longest common subsequence* problem in order to find the longest common subsequence among the chunks which constitute the training instances.

The analysis of two entirely diverse scenarios allows us to show two more interesting implications of a rule representation based on an intermediate granularity. First, the grammatical variation in the narration of the target event plays a key role in the identification of the event's parameters. While a common practice to cope with this problem is to make use of normalization [2], we can see how some scenarios do not require such approach. Secondly, the difficulties that the partial parsing exhibited with a specialized domain give evidence to suggest that the module of syntactic analysis is also domain dependent.

In addition to the syntactic elements which the rule representation is based on, the number of fillers that the learned rule is able to extract is another important aspect of the rule representation. According to the studied scenarios, a flexible structure which allows the identification of the context that surrounds the relevant phrases is mandatory. For this purpose, an extension of the representation implemented by RAPIER [4] seems worth of experimentation.

Finally we describe the construction of the domain-specific glossary known as semantic lexicon. In fact, the description of the specific sub-language for our particular domain relies on both the meaning of the words and the meaning associated with our syntactic constituents: noun and verb groups. In fact, the set of semantic word classes which make up the lexicon may be classified according to the content word represented by noun groups and verb groups: while noun groups are used for the identification of either the major objects or facts in the domain, the most important actions or relationships are spotted by the analysis of the verb groups.

In this way, this paper will argue that a through linguistic analysis of the corpus is needed to get around the peculiarities that a specialized domain entails. In other words, rather than considering the domain analysis module as the only domain dependent module, we show how the three modules which make up the basic architecture of IE demand a rigorous linguistic scrutiny.

In the next section we give a brief description of the studied scenarios. In section 3 we review the problem of a similarity metric for a rule representation based on chunks. In section 4 we show how the grammatical variation and the scattering of events in text are relevant properties of the scenario. We also show in this section, examples of how the partial parser struggled with a specialized scenario. In section 5 we describe how the definition of the lexicon is beyond a straightforward analysis of the content words. Finally, we conclude with a discussion in section 6.

2 Scenarios

For our experimentation, we have chosen two scenarios. One is a "classic" data set: Management Succession [3], and the other one is a specialized corpus: the football domain. Since the former corpus is a very well known scenario, we describe in this section the football domain. The corpus consists of a set of articles, commonly known as "clockwatch", which describe the main events of a football match: goals, substitutions, booked players, missed chances, etc. This collection of articles has been drawn from the BBC Sport website[1]. Our corpus contains 101 texts and, the average size is roughly 1000 words per match-description (109,370 words in total results in an average of 1082 words per text).

At first glance, our corpus seems quite simple but the linguistic complexity of the events is beyond shooting a ball. The linguistic complexity of the football texts is as worthy of linguistic analysis as those that were used in the Message Understanding Conferences, namely: Terrorism in Latin America, Join Ventures activities and Management Succession events. According to the five information extraction tasks, as defined by Cardie [5], football texts provide enough material for linguistic research for each of these tasks. For example, Named Entity Recognition is an imperative task for the identification of the protagonists involved in the events, that is, players, managers, referees and places as stadiums. There are also goal descriptions in which Coreference Resolution would be very useful for the elucidation of anaphoric references (frequently pronouns) that permit to identify the corresponding protagonists (or antecedents) in the event. The identification of relations between the entities, known as Template Relation construction, is another essential task that would be also very convenient for the recognition of the players involved, either when a goal is scored or when an incident has occurred. So, as we can see within the vastness of events (along with their corresponding peculiarities) that surrounds football, this domain also represents a significant linguistic challenge.

Table 1 shows some sentences which represent the set of target events: goals, substitutions and booked players; and the corresponding filled templates for each of the sentences are shown in table 2. In these examples we can note how the protagonists for each event play a specific role and denote a relationship between them. For instance, the template for the *goal* event illustrates a relationship between two players, the passer (PasserPlayer) and the striker (StrikerPlayer), when a goal is scored. In an analogous way, the template for the *substitution* event illustrates a substitution relationship between two players, the player who comes in (InPlayer) and the player who goes off (OffPlayer). The next event, *booking*, illustrates an offensive relationship between a player who is the aggressor (BookedPlayer or ExpelledPlayer) and another player who is the offended (InjuredPlayer) or the cause (Reason) for which the aggresor has been booked with either a yellow or red card.

3 Rule Representation

Given that, on the one hand, an event alludes to an action or a relationship between entities, and on the other hand, we are interested in extraction rules based on local context only, that is, multiple-sentence event descriptions are

[1] http://news.bbc.co.uk/sport1/hi/football/default.stm

Table 1. Four sentences which illustrate the target events

Goal
20:58 Juan Sebastian Veron plays a fabulous through-ball to Ole Gunnar Solskjaer, who takes a touch before firing under Deportivo's keeper from an acute angle.
Substitution
78 mins: The outstanding Gerrard is substituted for Bayern Munich's young midfielder Owen Hargreaves.
Booking
9 mins: England midfielder Scholes is booked for a reckless late tackle on Charisteas.

Table 2. Filled templates

Goal	`Minute`	`PasserPlayer`	`StrikerPlayer`
	20:58	*Juan Sebastian Veron*	*Ole Gunnar Solskjaer*
Substitution	`Minute`	`InPlayer`	`OffPlayer`
	78	*Gerrard*	*Owen Hargreaves*
Booking	`Minute`	`BookedPlayer`	`InjuredPlayer/Reason`
	9	*Scholes*	*Charisteas*

beyond the scope of this work, we must look for a frame which allows us to stand for the local context that encloses an event.

Among the multiple concept representations which have already been observed in the past (most of them based on frames in terms of classic syntactic constituents), we have devised a rule representation which synthesizes the features of two previous systems. LIEP's representation [6] describes the syntactic context of the path between the target entities, that is, the slot fillers represented as noun groups, as well as the semantic classes of the heads of the target nouns groups. Second, RAPIER's representation [4] denotes both syntactic and semantic information around a target noun only. In this way, a plausible rule representation is the integration of these previous abstractions. Thus, in our system, the linguistic constituents have been framed taking into account both schemes: the path between the target entities and the context that surrounds the entities to be extracted.

In fact, the local context that surrounds an event is represented by a structure which contains four components (or fields): *pre-filler*, *filler*, *link* and *post-filler*. *Filler* represents one of the target entities, that is, a slot in the template. *Pre-filler* corresponds to the text that precedes the first filler of the event, whereas *post-filler* denotes the text after the last filler. Finally, *link* is concerned with the path between the event's fillers. Thus, an action or event of n fillers is represented by $2n+1$ components or fields. The next example: "1617 Arsenal continue to punish Fulham as Thierry Henry scores his second goal from another pass from Sylvain

Wiltord.", is an instance of a goal event, and the corresponding representation is shown in table 3.

Table 3. Syntactic and semantic elements for an instance

Element	Chunk list
Pre-Filler	[1617_CD]] [[Arsenal_NNP]Team] ((continue_VBP)) ((to_TO punish_VB)) [[Fulham_NNP]Team] as_IN
Filler1	[[Thierry_NNP Henry_NNP]Player]
Link	((scores_VBZ)Score) [[his_PRP second_NN goal_NN]Goal] from_IN [[another_DT pass_NN]Pass] from_IN
Filler2	[[Sylvain_NNP Wiltord_NNP]Player]
Post-Filler	._.

A deeper analysis of the information in this table is worth our attention. First, we can see the syntactic and semantic constituents corresponding to the event represented by a specific training instance[2]. The plain text is submitted to a shallow syntactic processing known as chunking which produces a flat list of noun groups, verb groups and function words. These in turn will be processed by the entity recognizer and the semantic tagger respectively in order to provide an elementary interpretation of the syntactic constituents. Secondly, we note the particular components that make up the representation of a goal event. This particular structure contains two fillers, the *link* field between them and the *pre-filler* and *post-filler* fields respectively; therefore, the total number of components in the structure which represent this event is five.

3.1 Similarity Metric

How LEEP discerns the similarity between two positive training instances is a question that revolves around the underlying rule representation. More exactly, the granularity that underlies the meaning representation plays such a relevant role that a suitable criterion for similarity must be devised. For example, previous rule extraction learning systems for text such as AutoSlog [8] and Crystal [9] make use of unification as the computational technique for their similarity purposes according to their rule representation defined in terms of conventional syntactic elements as subject, object, preposition phrase, etc.

Given that the number of conventional syntactic categories is small and well defined, the use of unification as computational technique for similarity purposes seems appropriated for rule extraction learning systems for text whose rule representation is based on these conventional features. However, a finer grained meaning representation, as a result of the partition of a sentence into smaller

[2] Round brackets represent verb gropus whereas square brackets stand for noun groups. Word classes are shown between brackets.

linguistic units, that is, grounded in terms of chunks, gives rise to an undefined and less precise definition of the number of syntactic features to be analyzed. In fact, when the number of linguistic units to be considered in the similarity processing has been not only increased but also undefined, that is, the number of features to be considered is variable, the evaluation of the distance between two training instances is more intricate.

The challenge then is to find out a similarity metric that allows us to identify regularities among the linguistic units in which our rule representation is based on: noun groups, verb groups and function words only. Since the meaning of a sentence is based not only on the words that make it up, but also on the ordering and relations among the words [7], we try a similarity metric which takes into account the ordering among the linguistic features: *longest common subsequence.*

To accurately distinguish the protagonists involved in the event, previous systems such as AutoSlog [8] and Crystal [9] regard as a similar instance that one whose slots to be extracted must be contained in the same syntactic constituents as the seed's slots. For example, if the seed represents a substitution event such as *Owen is replaced by Heskey* in which *Heskey* represents the player who comes in and *Owen* the player who goes off, an event such as *Sven-Goran Eriksson replaces Gerrard with McManaman* could not be a candidate instance worth having similarity analysis, since the fillers to be extracted do not belong to the same syntactic constituents (while the subject in the former sentence represents a target player, the subject in the latter sentence represents a different concept: the manager).

But for us, instead of looking for similarity among specific syntactic constituents, we care about the order of the slots to be extracted as an effect of the rule representation implemented. By making use of the same example shown in the previous paragraph, the event *Sven-Goran Eriksson replaces Gerrard with McManaman* could be considered a candidate instance to similarity evaluation, as the order of the target concepts is the same than the seed. However, an event such as *Kanu replaces Thierry Henry* could not be a similar instance since even though it describes a substitution event, it does not exhibit the same order in the fillers (*Thierry Henry* is the player who goes off, whereas *Kanu* is the player who comes in.). In this way, in order to distinguish between the protagonists involved in an event, we are concerned with a crucial linguistic element in its description: the voice. To cope with this grammatical variation, inflectional information about specific verbs is included in the semantic lexicon. In fact, sets of verbs which make up a particular semantic class are represented by the inflected form of the class. For example, verbs such as *replace* and *substitute* are represented as "Replace" or "Replaced" according to the voice of the event.

4 Syntactic Analysis

4.1 Grammatical Variation

The boundless expressive power of the language is present in the Management Succession domain. The most straightforward way to notice this inherent characteristic of the language is the scattering of events in text. In fact, the spread of the target information is a common scenario in this domain (multiple sentences describe the event) as well as the number of combinations which describe the relative order of the fillers in the expression of the target concept (single sentence or local context). For example, in the particular case of instances which contain information about Organization, PersonIn and Position, there are multiple combinations which must be considered for the definition of the corresponding patterns.

Even tough a wider variability in the narration of the target event in the Management Succession domain was experimented, the football domain also gave evidence of this phenomenon. How did our system cope with this grammatical variation? What were the implications of the use of an intermediate linguistic granularity?

Grishman [2], in order to surpass the limitation of partial parsing, that is, the incapacity to capture paraphrastic relations between different syntactic structures, defines metarules which expand subject/object relationships into patterns for the active, passive, relative, etc. clauses. In our particular case, we do not make use of metarules for the expansion of subject/object relations since these syntactic constituents are unknown for us. Rather than to define a pattern for each syntactic structure, we learn rules according to the arrangement of the fillers on the training instances corresponding to each target event. In this way, instances whose syntactic structure is *"X succeeds Y"*, or *"X, who succeeded Y"* are put into the same training set, whereas *"Y was succeeded by X"* belongs to a different set. Indeed, this is the way in which our system, according to the specific arrangement of the fillers rather than to a particular syntactic structure, finds out the salient regularities of the target events.

4.2 Partial Parsing

An important difference between the domains examined by our system was the fact that the MUC-6 texts did not represent any problem for the chunker. There were no practically syntactic errors in the processing of this generic type of texts, as opposite to the football texts which can be considered as texts which use a specialized lexicon.

In fact, as an indication of the complexity linguistic in the processing of the football domain is some parsing errors in which the chunker has incurred, particularly in goal declarations compared to the precision exhibited in the parsing of substitutions and cautioned player utterances. For example, in instances such as *"Steven Gerrard equalizes for Liverpool"*, *equalize* is considered as noun; in *"Nigel Martyn made a superb save to deny Scholes"*, *save* is sometimes tagged as verb or preposition.

5 Domain Analysis

The description of the specific sub-language relies on both the meaning of the words and the meaning associated with grammatical structures. In this way, determining the semantic class of a relevant noun group or verb group revolves around scrutiny of their corresponding heads. Nevertheless, the simple analysis of the heads is not enough: modifiers such as adjectives and adverbs may alter the concept so these elements also play a crucial role in the semantic tagging.

For example, in the football domain, lexical terms such as *almost*, or *deflected*, alter substantially the meaning of a head considered in isolation. Table 4 shows how the adverb *almost* acts on the verb *score* in the verb group *almost scores* (so in this case the POS pattern is “Adverb Verb”), whereas in the noun group *deflected shot*, the noun *shot* is modified by the adjective *deflected*.

Table 4. Modifiers such as *almost* and *deflected* in a verb and noun group respectively

80 mins: United sub Michael Stewart almost scores a spectacular goal but is denied by a fine Ricardo save.
The Scots were inmediatly on the offensive and a deflected shot from Stilian Petrov flashed narrowly wide.

There are, however, other situations in which modifiers are not always in close proximity to the head of the noun or verb group as in the previous case. In these circumstances, we made use of the variance-based co-ocurrence discovery approach introduced by [10]. Under this approach, important combinations of words for our corpus such as *shoots-wide* exemplify the vital role played by the modifier *wide*. The next instance *2022 Fabio Rochemback shoots just wide from 20 yards for Barcelona...* is an example which illustrates this case.

We also identified handy collocations in this domain. Collocations such as *make way for* represent a specific and concrete meaning which can not be captured by the semantic generality represented by the verb *make*. The next instance *75 mins Van Nistelrooy makes way for Diego Forlan after suffering a slight knock which requires an ice pack* is an example of an event described in terms of this collocation that comprises more than one chunk.

Finally, dealing with lexical ambiguity is also part of semantic interpretation. For example, the most important word in our football-corpus, *goal*, is ambiguous between two senses: the event which represents to score, and an area on a playing field, usually marked by two posts with a net fixed behind them (see table 5). Even though we are disregarding another possible meaning for goal: aim or purpose, it would not be possible for our semantic tagger to label this ambiguous term with a specific semantic category. In other words, the phenomenon of *polysemy*: a single lexeme with more than one meaning, is also present in our football domain. How did we cope with this situation? We dealt with this ambiguity case by making use of word collocations through statistical analysis of our corpus [11], specifically the frequency-based approach. By using this straightforward statistical method, we could work out a regularity that allowed us to distinguish the first sense (to score) of this term: an ordinal number followed by the term goal. Put in another way, in this corpus is very common to come across expressions such as *third goal* (Most of these expressions mean a goal for a specific team, but however, there may be expressions such as: ... on Derby with his 16th goal of the season, which mean a goal for a specific player rather than for a team.).

Table 5. Illustration of lexical ambiguity of goal

Sense	Instance
Score	83 mins Sydney Govou scores a vital goal for Lyon in Group D.
Area	4 mins Aristizabal hits the post when one-on-one with Mexican keeper Perez Rojas and just seven yards out from goal.

Likewise, more cases of polysemy were manually discovered during the annotation process of the texts. Metaphoric expressions such as *drill* and *slot* to describe the action when the player scores a goal, as well as terms such as *fire* and *rifle* to describe the action when the player shoots the ball, prove the great diversity of expressions used in the description of the football events.

6 Discussion

In the football domain, the experimentation conducted by our system suggests a deepest research in the recognition of the wide variability of terms used in the description of the events (lexical variation), rather than the use of subject/object relations to generate syntactic variants of a pattern, as a more convenient alternative to improve the covering of the learned rules.

Two particular difficulties were detected for the correct identification of specialized terms. First, the use of metaphoric expressions is present no only to describe the action to score (for example, terms such as *goes-off* in *Darius Vassell collects the ball and hits a superb right-foot shot that goes in off the post*, but also to portray the relationship between the players. For example, the use of *read* to describe the connection is illustrated in: *Berbatov read Yildiray Basturk's angled pass and slipped the ball across Barthez to reduce the deficit*. Secondly, the parsing problems shown in section 4 give evidence of how syntactic analysis is a module that goes through limitations when the parser deals with a specialized domain. Training the parser is a paramount activity to cope with this sort of domains.

On the other hand, the story is rather different for the MUC-6 corpus. The main difficulty in this domain is the grammatical variation. For example, a close scrutiny of the texts was required since a considerable percent of the information in the filled templates was obtained beyond local context: event descriptions spread across multiple sentences [12]. Furthermore, in order to cope with the wide diversity in the expression of the target concept (the potential number of combination among the template's fillers), the use of metarules is rather than essential in this domain. In this way, to some extent, an improvement in coverage is plausibly expected.

In this way, our analysis suggests that a deep linguistic analysis is essential to adapt our modules to extraction tasks. Two domains give evidence of how the complexity of a scenario relies on the peculiarities of its expressions [13].

References

1. Appelt, D., Israel, D.: Introduction to Information Extraction Technology. A Tutorial Prepared for IJCAI-99, (1999)
2. Grishman R.: The NYU System for MUC-6 or Where's the Syntax? In Proceedings of the Sixth Message Understanding Conference Morgan Kaufmann Publishers, 167-175, 1995.
3. MUC-6: Proceedings of the Sixth Message Understanding Conference San Francisco, CA, Morgan Kaufmann Publishers, 1995.
4. Califf, M.E., Mooney, R.: Relational Learning Techniques for Natural Language Information Extraction. The University of Texas at Austin, (1998)
5. Cardie, C.: Empirical Methods in Information Extraction. AI Magazine 39(1):65-79, 1997.
6. Huffman, S.: Learning Information Extraction Patterns from Examples. Connectionist, Statistical, and Symbolic Approaches to Learning for Natural Learning Processing. Springer, 246-260, 1996.
7. Jurafsky, D., Martin, J.H.: Speech and Language Processing An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition. Prentice-Hall, Inc., 2000.
8. Riloff, E.: Information Extraction as a Basis for Portable Text Classification Systems. Ph.D. thesis. University of Massachusetts, Amherst, 1994.
9. Soderland S.: CRYSTAL: Learning Domain-specific Text Analysis Rules. University of Massachusetts, Amherst, (1997)
10. Smadja, Fraank: Retrieving collocations from text: Xtract. Computational Linguistics, Vol. 19, 143-177, 1993.
11. Allen, J.: Natural Language Understanding. The Benjamin/Cummings Publishing Company, Inc., Second Edition, 1995.
12. Stevenson, M.: Information Extraction from Single and Multiple Sentences. In Proceedings of the Twentieth International Conference on Computational Linguistics (COLING-2004), Geneva, Switzerland.
13. Huttunen, S., Yangarber, R., Grishman, R.: Complexity of Event Structure in IE Scenarios. In Proceedings of the 19th International Conference on Computational Linguistics (COLING-2002).

Identifying Information from Heterogeneous and Distributed Information Sources for Recommender Systems

Silvana Aciar, Josefina López Herrera, and Josep Lluis de la Rosa

University of Girona
Campus Montilivi, edifice P4, 17071 Girona, Spain
{saciar,peplluis}@eia.udg.es, josefina.lopez@udg.es

Abstract. With easy access to World Wide Web the users are overloaded of information. Recommender systems have emerged as research approach to address this problem. The users want to find what they need, when they need it and under the conditions that they want.
These conditions drives to the recommender systems to access to different sources to find relevant information to recommend.
In this paper we presented a set of intrinsic characteristics to determine the relevance of the sources to make recommendations.
These characteristics allow to have a representation of the information contained in the sources that is relevant to recommend and a set of criteria to select the most relevant.
A multi-agent system has been designed to obtain these characteristics. Preliminary results of recommendations made with the selected sources are presented in this paper.

1 Introduction

Today knowledge integration from various sources is gaining importance in areas such as Process Administration, Data Warehouse, e-commerce, Knowledge Administration and Marketing. The information integration allows more knowledge to be acquired from disperse information.

However, identifying, accessing and integrating information from various sources is a difficult task, made complex by [1][13]:

- the dynamism,
- the geographic distribution and
- the heterogeneity of the sources

The agent paradigm has emerged as tool to manage the complexity in the integration systems. It is a promising proposal for information recovery and a solution to distributed problems in which mechanisms of cooperation and coordination are necessary [7].

Today the quantity of available information is enormous, almost immeasurable and the complexity of the systems grows if the agents have to access to all the information sources [12] [4].

© A. Gelbukh, R. Monroy. (Eds.)
Advances in Artificial Intelligence Theory
Research on Computing Science 16, 2005, pp. 55-64

In this paper the agents select and integrate relevant sources for a recommender system. A recommender system receives as inputs the user preferences and generates recommendations of products or services according to its preferences [11].

A measurement based on the intrinsic characteristics of the sources is used to establish the relevance to select the information sources.

This paper is organized as follows: In **Section 2** the intrinsic characteristics and the relevance measurement used to select the sources are defined. The MAS (Multi-Agent System) designed to select the relevant information sources is presented in **Section 3**. In **Section 4** is presented the case of study with the obtained results. Finally, conclusions and future work are presented in **Section 5**.

2 Intrinsic Characteristics of the Information Sources

In this section a set of characteristics of the sources is defined that allow to establish a relevance value to select information sources to make the recommendations.

The set of characteristics provides:

- A representation of the information contained in each source, and
- a set of criteria to compare the sources and to select the most relevant.

In Table 1 the characteristics defined are listed.

2.1 Selecting the Information Sources

The selection of information sources is based on the relevance value obtained from the intrinsics characteristics. Relevance R(S) is defined as quantity of information that a source S can contain for a recommendation:

$$R(S) = \sum (f_i \times p_i \times [\frac{1}{\sum_{i=1}^{n}(f_i)}]) \tag{1}$$

Where f_i = Weight of the characteristic i (*Completeness, Diversity, Frequency, ...*), $f_i >= 1$. The characteristic most relevant for the domain have the highest value. n = Total number of characteristic. p_i = value of the characteristics i, it is obtained with the equations from Table 1.

2.2 Making the Recommendation

Information to make recommendations is only obtained from the information sources with the highest values of relevance.

Recommendation-based contents were used to make the recommendation in[2][10][8] [6]. The system processes information from various sources and attempts to extract characteristics and useful elements about their content. Content-based filtering techniques can vary according to their degree of complexity. For

Characteristics	Description	Measurement
Completeness [14]	Number of users from one information source also found in another source	*Completeness* $= \sum(A \cap B)/\sum A$ $A \cap B =$ *Users existing in both sources* A = *Users existing in the source A* B = *Users existing in the source B*
Diversity [16]	Number of user groups The users are grouped according to a common criterion	*Diversity*$= H = \sum p_i * \log_2(p_i)$ $p_i = n/N$ n = *Number of users included in group i* N = *Total number of users in the source*
Ontology	Semantic representation of the information *contained in the sources*	*Relevant Attributes* $= \frac{a}{A}$ a = *Number of relevant attributes i in the source* A = *Number of relevant attributes j for the recommendation*
Timeliness [17]	Update of the information about the users interactions	*Timeliness* $= \sum w_i * c_i/N$ c_i= *Number of user that purchased i in a period of time i* w_i = *Weight of the period of time i* N = *Total number of user in the source*
Frequency [17]	Frequency of the user interactions	*Frequency* $=\sum w_i * f_i/N$ f_i = *Number of user in a ratio of p purchase frequency* w_i= *Weight of the a purchase frequency* N = *Total number of user in the source*

Table 1. Intrinsic characteristics of the information sources

example, a search based on key words is one of the simplest. A more complex technique is the based on the extraction of semantic content from information contained in documents. In the present work the information to make recommendations is the user behaviour, i.e. the purchase frequency, the quantity purchased, the last date that a purchase was made and the product purchased.

For example, the following information is found in the data-base for user X:
Last date of purchase = 10/05/2005
Amount = 1000
Frequency = high
Product = computer

A computer would not be recommended to this customer because it is an infrequent product and the purchase date is very recent.

The RFM algorithm (Recency, Frequency, Monetary)was used to obtain knowledge of the user behaviour. This algorithm divide the users according to their last date of purchase, the frequency of their purchases and the quantity purchased. Based on this information, decisions concerning whether or not to make a recommendation to deserter customers can be taken. Analysis of behaviour is the key to obtain favourable responses to the recommendations made to customers.

3 Multiagent System to Select Relevant Information Sources

A MAS has been designed to select the relevant information sources. The MAS is shown in Figure 1

- Each information source is managed by a **Manager Agent (MA)** that has abstract information about the content of the source, defined in terms of the intrinsic characteristics. In addition, the agent is responsible for measuring the relevance of the source.
- The **Property Agents (PA)** are in charge of obtaining the characteristics. They are task agents [13] and interact directly with the source. There is an agent for each property of the source and their task is to measure the intrinsic characteristic.
- A **Selector Agent (SA)** is responsible for selecting the most relevant sources in order to make the recommendation.
- A **Recommender Agent (RA)** is an Interface agent [13]that interacts with the user. Once the sources are selected, it makes the recommendation.

The adaptability to the environment in a MAS is one of the most important characteristics. In this system the agents adapt themselves and learn from the environment, updating the values of the intrinsic characteristics of the sources. The agents periodically calculate these values, so data change, the values of the characteristics also change. For the selection phase, the relevance measure R(S) is calculated from these values, and is therefore also updated. Based on these

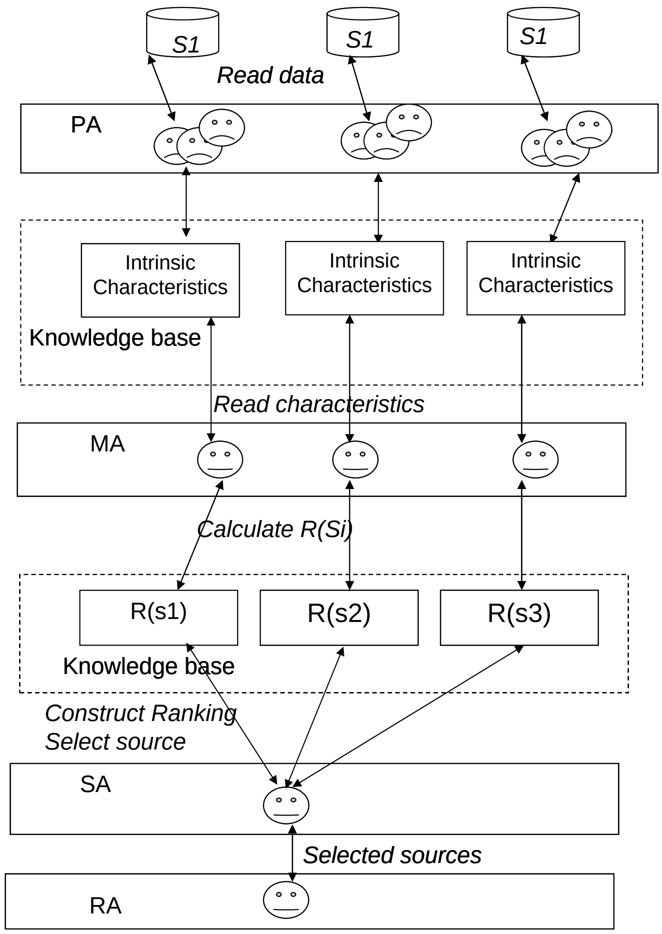


Fig. 1. Multi-Agent System to select relevant information sources

hypotheses, it can be said that MAS learn and adapt to their environment (preferences and tastes of the users). In general terms, planning in this kind of system can be viewed as a centralized planning for distributed plans [5].The agents do not work as a team as described in [3]. In complex societies, such as the administration of information from various sources, the amount of communication needed for agents to reach agreement on a joint decision would overcharge the system. This is the reason why the responsibility for taking decisions (selecting sources of information) is delegated to only one type of agent, the **SELECTOR AGENT**, and based on the relevance measure of the source. To obtain this objective, the other agents have to calculate the values of the characteristics allowing the relevance information provided by each source to be inferred.

4 Case Study

Eight data bases in the consumer package goods domain were used. These data bases contain information about clients, products and a historical record of the products purchased.

They contain real data from more than thousand clients and the purchases made by them during the period 2001-2002. There is data about clients who made purchases on the Internet (OnLine) and purchasing information about customers who personally shopped in the supermarket (Off-Line).

The different data bases have common users, who are easily identifiable because they have the same id in all the data bases.

At the beginning the agents have no knowledge of the sources. They do not know which is the most relevant to make the recommendations. The agents begin to randomly explore the information sources.

The MA from each source creates PA to obtain the measurements to calculate the relevance of each source. Table 2 shows the different values of the measures obtained by the PAs.

These measurements were obtained following the equations mentions in Table 1. To obtain the diversity measure user groups were established according to the zone in which they lived (Z), their sex (S) and the number of persons in the family (F).

The MA obtains the relevance measure R(S) for each source as can be seen in Table 3 .

With this measurement, the SA has knowledge of the relevance of each one of the sources to select the most relevant ones.

Once the recommendation is computed and the result obtained, the SA has knowledge of the relevance of the source and the result of the recommendation.

Figure 2 shows how the recommendation improvement increases as a result of integrate information only from relevant sources (S1, S2, S4, S5, S6, and S8).

In Figure 3 is shown the recommendation results from all sources (S1, S2, S3, S4, S5, S6, S7 and S8).

In this graphic can be observed that the result of recommendation decrease with the information integration from source less relevant (S3 and S7).

	f_i	**S1** p_1	**S2** p_2	**S3** p_3	**S4** p_4	**S5** p_5	**S6** p_6	**S7** p_7	**S8** p_8
Characteristics									
Ontology									
Relevant Attributes	9	0,80	0,50	0,20	1,00	0,60	1,00	0,10	0,80
Diversity									
Z	3	0,13	0,11	0,12	0,14	0,71	0,24	0,25	0,23
F	3	0,66	0,67	0,67	0,73	0,07	0,56	0,56	0,49
S	3	0,20	0,20	0,21	0,11	0,20	0,19	0,19	0,26
Completeness	5	0,10	0,60	0,30	0,30	0,57	0,33	0,30	0,70
Frequency	7	0,23	0,40	0,25	0,20	0,50	0,30	0,20	0,45
Timeliness	5	0,25	0,40	0,42	0,15	0,47	0,35	0,50	0,30

Table 2. Intrinsic characteristics of the sources in the consumer packaged good domain

Sources	S1	S2	**S3**	S4	S5	S6	**S7**	S8
R(S)	0,39	0,44	**0,29**	0,45	0,49	0,50	**0,27**	0,52

Table 3. Relevance of the sources

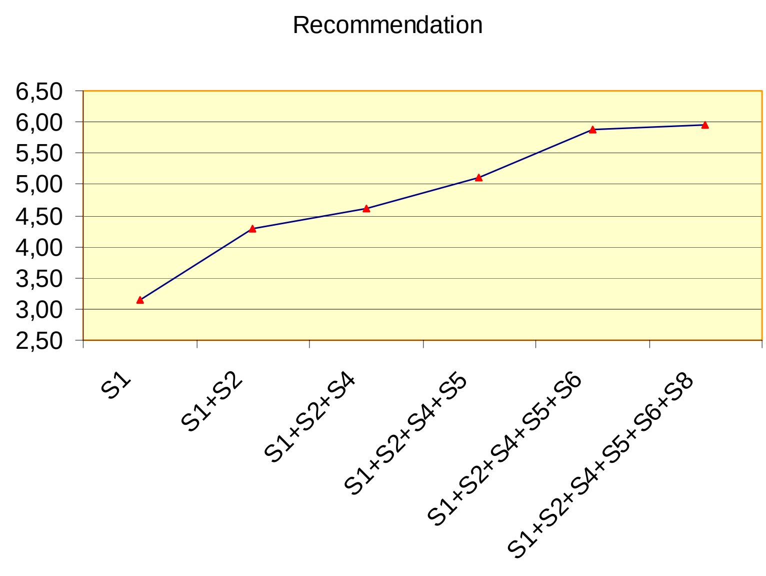


Fig. 2. Recommendation with the relevant sources

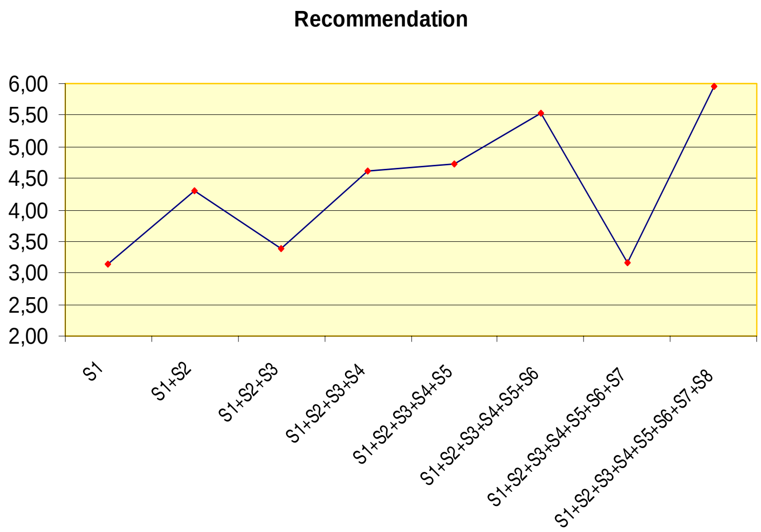


Fig. 3. Recommendation with all sources

5 Conclusion

This paper has introduced MAS to acquire and to integrate information from relevant information sources in order to improve the recommendation results. The sources of information are represented by the intrinsic characteristics. Based on them, the relevance of each source is obtained. Although very simple, this proposal produces significant results by improving the recommendation results. Future work should consider the need to compute recommendations integrating information from information sources in different domains, in which mapping between two or more ontologies of different domains will be necessary. It should also, evaluate the behaviour of the system when the information of a relevant source selected decays.

References

1. Arens, C. Y. Chee, C-N. Hsu, and C. A. Knoblock.V: Retrieving and integrating data from multiple information sources. Inter-national Journal on Intelligent and Coop-erative Information Systems. 2(2).1993,27–158
2. M. Balabanovic and Y. Shoham.:Fab: Content-Based, Collaborative Recommendation.Communications of the ACM,40(3),6–72,March,1997
3. P. Cohen, H. Levesque, and J. H. Nunes: On Acting Together.Proceedings AAAI-90,1990.

4. Edmund H. Durfee and Jeffrey S. Rosenschein: Distributed Problem Solving and Multi-Agent Systems: Comparisons and Examples. Thirteenth International Distributed Artificial Intelligence Workshop, 1994,pages 94– 104
5. Fisher, K:Problem Solving and Planning. In Klusch, M., Fischer, K., and Luck, M.2000.Working Notes of The Second European Agent Systems Summer School, EASSS.
6. Lieberman H.: Letizia : An Agent That Assists Web Browsing. Proceedings of the International Joint Conference on Artificial Intelligence,Montreal.1995
7. A.Moreno, D. Isern: Accessing Distributed Health-Care Services Through Smart Agents.Proceedings 4th International Workshop on Enterprise Networking and Computing in Health Care Industry (HEALTHCOM'02).2002,pages 34–41
8. Moukas, A. and Maes,P: Amalthaea: An evolving multi-agent information filtering and discovering system for the WWW.Autonomous Agents and Multi-Agent Systems.1998
9. Perugini, S., Goncalves, M. A.,and Fox, E. A: Aconnection centric survey of recommender system research. Journal of Intelligent Information Systems,volume 23[1],2004
10. Salton, G: The SMART Retrieval System - experiments in automatic document processing.T. Perntice-Hall, Inc,1971. Englewood Cliffs
11. Schafer, J. B., Konstan, J. and Riedl, J.: Recommender Systems in E-Commerce. EC '99: Proceedings of the First ACM Conference on Electronic Commerce,Denver, 1999. Pages:158–166
12. Shah, T. Finin and J.Mayfield.:Information retrievel on the semantic web. 10th International Conferenc on Information and Knowledge Management. ACM Press, 2003.
13. Sycara, K., Decker, K., and Williamson, M.: Modeling information agents: advertisement, organizational roles, and dynamic behavior. Technical Report WS-9602, American Association for Artificial Intelligence, 1996.
14. F. Naumann and J. C. Freytag.:Completeness of Information Sources. Technical Report HUB-IB-135, Humboldt University of Berlin, 2000.
15. X. Jin, Y. Zhou, and B. Mobasher:Web Usage Mining Based on Probabilistic Latent Semantic Analysis. ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD'04),Seattle,2004.
16. R.J. Hilderman, H.J. Hamilton: Principles for mining summaries using objective measures of interestingness. ictai, p. 0072,12th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'00). 2000.
17. P. Cykana, A. Paul and M. Stern: DOD Guidelines on Data Quality Management. Proceedings of the Conference on Information Quality, Cambridge, MA, 154-171. 1996

Model Selection in Data Mining: A Statistical Approach

José Alberto Frydman, Francisco J. Cantú Ortiz⋆
Jorge H. Sierra, Rubén Morales-Menéndez

ITESM Monterrey campus
Monterrey NL, México
afrydman@sbs.gob.pe, {fcantu,jsierra,rmm}@itesm.mx

Abstract. Choosing the best algorithm for a machine learning task is an open problem. Statistical techniques such as *paired t-test* and k-fold cross validation have been proposed, but they require independence assumptions that are not met in many data mining tasks and for large data sets these techniques suffer from the multiplicative effect. We propose Analysis Of Variance as an alternative for comparing learning algorithms in large databases. We show that this approach avoids the drawbacks of the *paired t-test* and *k-fold* cross validation. We describe several experiments on a DELVE database that show the advantages of our proposal in comparing the decision tree learning algorithms Gain Ratio, Gini and Chi Squared Automatic Interaction Detection as well as the naive Bayes algorithm.

Keywords: Machine Learning, Decision Trees, *Paired t-test*, ANOVA.

1 Introduction

The development of learning algorithms in Machine Learning (*ML*) is quite vast and varied since there are many ways of searching the space of hypotheses for constructing ways of learning called models. Learning algorithms for model construction include Decision trees, and Bayesian nets. One of the fundamental questions in *ML* is this: given a specific learning problem and context, which algorithm is the one that generates the best model?. Alternatively the question may be in this way: given several learning algorithms and a large database which algorithm will produce the most accurate predictions?. This article intends to give an approximation to the second question. Our starting premise is that the most important factor to look at when selecting an algorithm is its error in predicting unseen cases; however, we take into account that other factors like speed of learning and understandability of the generated model can be more important in some domains.

A model, also called a classifier, is a function that given an example, the function assigns the example a class among several classes. A learning algorithm

⋆ Corresponding author

© A. Gelbukh, R. Monroy. (Eds.)
Advances in Artificial Intelligence Theory
Research on Computing Science 16, 2005, pp. 65-74

is a function that given a database of instances and its classes constructs a classifier for assigning classes to unseen instances. The learning function is searched from a huge space of possible hypotheses.

Model validation means the evaluation of algorithms by comparing the accuracy of the models generated by the algorithms. Validation can be divided basically in three independent processes: **(1)** Measuring the error in the algorithm, **(2)** Improving the performance, **(3)** Comparing two or more algorithms. We propose a method for comparing more than two learning algorithms to select the best one depending upon the problem to be solved.

In this article we start from some premises related to points 1 and 2. Measuring the error deals with establishing a way of predicting the error in future cases; there are many different techniques to measure this, that can be chosen according to the number of examples. [16] suggests re-sampling techniques for data bases with less than $1,000$ examples, and the holdout method for data bases where more than $1,000$ examples can be used for testing. The Holdout method consists of separating the data in two sections: training and testing. We use the holdout method and justify why cross validation is not appropriate for model comparison.

For improving the performance of a model there are three strategies: a) manipulate the database reducing the training series, perform attribute selection or create new ones b) combine several models and weigh the results (Bagging or Boosting [1]) c) optimize the parameters of the algorithm. When comparing Data Mining (DM)algorithms it is important to recognize that any of these three strategies could produce a better result on one algorithm over another, and this fact would make algorithm selection a much more complicated process, [12]. However, [16] suggests the use of each algorithm in its best form, that is, if the algorithm is improved by some kind of manipulation, then it should be used. In this work we used the first approach.

We propose the use of *Analysis of Variance* ($ANOVA$)as an alternative test to the *paired t test*. Following the same line of reasoning, no costs for different errors were taken into account (false negative or false positive) because not all of the learning algorithms can deal with them. We worked only with the 0-1 loss function.

This work is organized as follows: In section 2 we present a review of works for comparing models of *Data Mining*, and we discuss some common problems. In section 3 we present the $ANOVA$ as a proposed solution to these problems. In section 4 we describe the experiments and the results. Finally, section 5 presents the conclusions.

2 Literature Review

Usually, model error is measured by a technique known as ten-fold cross validation since it has been shown that it gives an almost unbiased estimator of the true error [10]. Ten-fold cross validation consists on randomly splitting the database in ten sections. Then, one section is left aside at a time and the algorithm is trained with the 9/10 remaining, after that, the error is computed in the series that was left aside. Hence the learning algorithm is performed ten times, each time with a different training series (even though they share many examples) and each time tested in a completely different test series. Finally, the average of the ten estimated errors is taken to produce the estimate of the error for the classifier. Traditionally, when we need to compare the error of two learning algorithms A and B in a database D_0, it is customary to apply the following procedure based on k fold cross validation, [13]:

1. Database D_0 is split into k sections $T_1, T_2, \ldots, T_k$ (folds) where each section contains at least 30 examples (heuristic rule based on the central limit.)
2. For $i = 1$ to k , use T_i as the testing series and the remaining of the data as the training series S_i.
   - Measure the error of the algorithms, $E_A(S_i)$, $E_B(S_i)$ in the testing series T_i
   - Measure the difference in the computed error obtained by each of the two algorithms $\delta_i = E_A(S_i) - E_B(S_i)$
3. Compute the average of the differences of the errors obtained $\overline{\delta}$, where $\overline{\delta} = \frac{1}{k}\sum_{i=1}^{k}\delta_i$

This procedure is identical to ten-fold cross validation, with the only difference that instead of obtaining the error in only one algorithm, the difference in errors of the two algorithms is obtained. It is worth stating that the number k can be 10 or a larger number with the only restriction that there are at least 30 examples in each section (fold).

Now, if the two algorithms have the same performance it would be expected that the value of $\overline{\delta}$ is approximately zero. But, how close to zero? The well-known *paired t test* is generally used to determine if the value of $\overline{\delta}$ is small enough to conclude that the two algorithms have the same error. It is called *paired t test* because the two values obtained, $E_A(S_i)$ and $E_B(S_i)$, are statistically dependent (in this case they are dependent because the algorithms are trained in the same training series and are also tested on the same testing series). In the *paired t test* the test statistic is: $t = \frac{\overline{\delta} - \Delta_0}{S_D/\sqrt{n}}$ where Δ_0 tends to 0.

The null hypothesis states that the difference in error is practically zero whereas the alternative hypothesis states that the difference is less than, greater than or not equal to zero. However, one of the assumptions of the *paired t test* is that the **pairs of values** $(E_A(S_1), E_B(S_1)), \ldots, (E_A(S_k), E_B(S_k))$ are independent, assumption that is violated in the previously explained procedure. But how and where is this assumption violated?

The *paired t test* is based on the premise that all individual measures are independent (measures in this *paired t test* are differences δ_i on the k estimated errors of the two algorithms).

Unfortunately, cross validation (as well as bootstrapping) introduces some degree of **dependence** amongst these measures, and this produces a bias on the *t test* towards rejecting the hypothesis of equality of means. But, why are the measures dependent? Each measure depends on two quantities: a) the training series and b) the models constructed from the corresponding training series. Testing series are independent, but the models are not. Each algorithm develops a model in *10* training series, (assuming $k = 10$) which are almost identical.

Each training series shares 89% (8/9) of the data with all the other training series on the 10 iterations, therefore the training series overlap, and this introduces a dependence amongst the models generated from them, hence, the errors produced by these models will also be dependent.

But this is not the only drawback on using this test. When we want to compare more than two population means, the *paired t test* is not recommended for the following reason. Suppose that from a certain population five samples are considered from which their corresponding means are computed: $\overline{x}_1, \ldots, \overline{x}_5$. It is required to perform the following 10 *t tests* to compare the 10 pairs of sample means $(\overline{x}_1, \overline{x}_2), \ldots, (\overline{x}_1, \overline{x}_5)$. If more sample means are to be compared, more t tests should be performed; for example, a total of six samples increase the number of pairs to fifteen. However, when each sample mean is paired with another and a *t test* is performed for each pair of means, the possibility of it showing significant differences for some pairs of means is increased, even if the sample means proceed from the same population! This phenomenon is known in statistics as the multiplicative effect. Then, whenever comparing three or more sample means, amongst themselves, the *t test* is inappropriate. Another technique should be used to test the hypothesis of equality of means [1].

Up to here we have mentioned only the *paired t test*, however, there exist other approaches to comparing learning algorithms. For instance Dietterich [5] suggests the test $5 \times 2cv$ as the best test to compare errors (means) for algorithms that can be executed efficiently 10 times, he warns however that this test violates some of the independence assumptions. Other authors even question if the type I error is the one to compare (Type I error consists on falsely rejecting the hypothesis of equality of means). Also, due to how unstable learning algorithms are, any comparison between two or more of them must be done using multiple runs on different training series to remove the variability of the algorithm [4], [7]; this is the approach we take here. There is not a consensus amongst the research community about the method to use for comparison studies [11].

3 Comparing Algorithms using Analysis of Variance

We have mentioned two drawbacks about using the *paired t test*; on one hand the overlapping of the training series due to cross validation and on the other the multiplicative effect when comparing more than two algorithms. To correct the former, the method proposed here is applicable only for large databases[2], so that the holdout method can be used to measure error and hence eliminate the dependence produced by cross validation. Typically, the databases are huge (thousands or millions) in the data mining community. To avoid the latter, a statistical test should be found that does the same function as the *paired t test* but that can be used for more than two populations; such a test is the Analysis of

[1] Sample mean corresponds in this work to the algorithm error.

[2] In some domains 2,000 examples can be a large number whereas in another domain would be totally insufficient. We propose the use of learning curves to determine the number of examples needed to learn an underlying pattern.

Variance (**ANOVA**). ANOVA has been proposed for learning in support vector machines [15].

ANOVA is a statistical procedure for comparing the means of a population under the effect of different treatments. It is called a one-way ANOVA when only one variable effects the population, in a several-way ANOVA there are more variables effecting the population. For this work the treatments studied with ANOVA are the learning algorithms that have been trained with the training series, and the effects these treatments produce on the testing series will be studied, that is, the errors produced by the classifiers. Before applying ANOVA we must verify that its statistical assumptions are satisfied [3]. These assumptions are: **(1)** The observations in each sample are independent. **(2)** The underlying distribution in each population is normal. **(3)** All the populations have the same variance.

4 Experiments and Results

We present two experiments and for each four learning algorithms are compared; three decision trees generated with different split rules, (Gain ratio, Gini, and CHAID or Chi squared Automatic Interaction Detection) and the Bayesian classifier (Naive Bayes). Several approaches have been proposed for learning decision trees from databases [8]. To compute the errors obtained by each of the algorithms, we used the *DM* software Mineset 3.1 (Silicon Graphics). In the decision trees the level of pruning was established at 0.7(default value) and in the Bayesian classifier the Laplace corrector was set to automatic.

The database used in the experiments consists of $45,000$ entries and was obtained from the Delve archives, it can be found under the name Adult Dataset. Each entry presents 13 attributes; 6 continuous, 7 nominal, and it does not show any missing values. The dependent variable is Income, and can be classified in greater or less than $50,000$ US dollars annually. The purpose of this database is to classify if a person has an income greater or smaller than $50,000$ dollars (dichotomous variable). From now on we call a *sample* from the population the series of errors obtained by the algorithms in the database, being the population the universe of all errors that would be obtained from an infinite database from the same domain of application.

The first experiment consists of a two way ANOVA (one factor being the algorithms and the other as the training and testing series - the two of them together -), using blocking in the training and testing factor. We split at random the database in 20 sections, that is, each part consisted of $2,250$ examples. In each of them, we chose at random $1,250$ examples for training and $1,000$ examples for testing; in this way we got 4 samples of 20 errors each. See left side in Fig 1.

The numbers may seem arbitrary but there is a reason for this; first, the database was broken in 20 parts because the analysis of variance gets a better consistency when the number of observations is relatively large (20 is an acceptable value). The decision to assign $1,250$ examples for training to each subsections due to the fact that the error made by the algorithms when training with $1,250$ examples was almost the same as the error made with all the

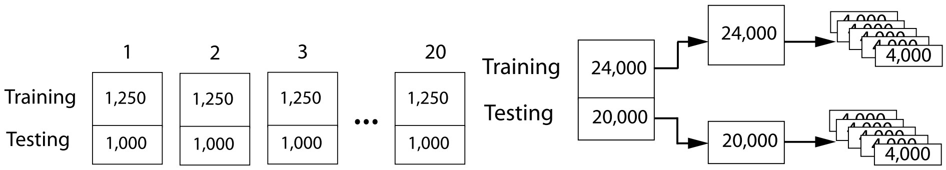


Fig. 1. Training and Testing examples : 1^{st} and 2^{nd} experiments

45, 000 examples (this was verified using learning curves), therefore 1, 250 training examples are a reasonable number to train the algorithms. Also, the 1, 000 test examples represent the minimum acceptable number of examples that are needed to compute with a good approximation the error in an algorithm (with only 1% deviation of the true error). To have an exact judgment on the error in a model, 5, 000 testing examples are needed (with a deviation in the computation of the error of 0.5%) [16]. Although the way the database was split may seem arbitrary, the reader may verify with has been said that there is a theoretical and also an empirical reason. The errors obtained with the four algorithms are shown in Table 1. See errors in Table 1 for these algorithms.

Table 1. Performance of the different methods

	Error					
Algorithm	1	2	...	19	20	Average
Gain Ratio	17.8	14.5	...	16.7	14.6	16.65 %
Chi-Squared	18.9	18.8	...	19	16.6	20.05 %
Gini	17.1	16.1	...	20.2	17.1	19.05 %
naive Bayes	18.5	17.3	...	17.5	18.4	17.92 %

The model generated in this first experiment is a model of mixed effects since we are considering a non-random factor (algorithms) and a random one (the training and testing series). This random factor represents the blocking. Blocking is done when an external variable is present and can affect the results of the test [2]. In the case of comparing algorithms, the external variable is the random selection of the training and testing examples. At first sight we can see that Gain Ratio is the treatment with less error and this is proven using analysis of variance. Once we have the observed errors (that conform four samples)we verify that the statistical assumptions of ANOVA are satisfied.

Assumption (1) is satisfied due to the form in which the samples were constructed. Both the training and testing examples are randomly chosen. All the samples have a size of twenty observations ($2,250 \times 20$). To check assumption (2) a statistical test known as Shapiro-Wilks was applied. In this test the null hypothesis is that the population is normally distributed. When applying the test to the 20 observations obtained from the treatment Gain Ratio we got a p-value of 0.4088 ($W = 0.9519$). As can be seen, the p-value is large and therefore the null hypothesis cannot be rejected. Similar results were obtained with the other three treatments (algorithms). So, assumption (2) is also satisfied in the four treatments. To test for assumption (3) 4 tests were conducted (in all four of them the null hypothesis is that the variances are equal). The tests: O'Brien, Brown-Forsythe, Levene and Barlett did not reject the null hypothesis, that is, we cannot reject the equality of variances in the average errors of the algorithms. It must be clear that assumption (3) states that the population variances must be equal regardless of the probability distribution of the data (the distribution may be binomial). The normality was tested by the classical tests.

Once we have verified that all the assumptions are satisfied, we can proceed to perform the analysis of variance. In Table 2 we can see the summary of the analysis of variance, as if there was only one factor. Since the p-value is less than 0.0001 we can safely reject the null hypothesis, which means the algorithms behave significantly different. In another results, not shown here, the treatments as well as the blocks are significant (we reject the null hypothesis), that is, they have an influence on the degree of error.

Table 2. Analysis of Variance like a whole

Source	DF	Sum of Squares	Mean Square	F Ratio
Model	22	235.629	10.7104	7.5926
Error	57	80.4065	1.4106	Prob>F
C Total	79	316.0355		<.0001

When the null hypothesis in the analysis if variance is not rejected, the analysis is concluded and no significant differences between the means (in this case average errors) were found [3]. But if the null hypothesis is rejected, it is of interest to know which means differ from the others. The method to continue with this analysis is called multiple comparisons. There are several tests in statistical literature to do this. One such tests is Tukey's test. For the case of comparing algorithms we would like to compare all pairs formed from the four treatments. When applying this procedure to the factor *algorithms* we get[3]: $Q_{0.05,4,57} = 3.75\sqrt{MSE/J} = 3.75\sqrt{1.41/20} = 0.995$.

As we may see, 0.995 is smaller than any difference between the averages of the four algorithms (17.92 − 16.65), (19.05 − 17.92), (20.05 − 19.05). Therefore, we can conclude that all four algorithms are significantly different amongst themselves: they are all different. It would be interesting to establish the reason

[3] MSE : Mean Square Error, Q: Critical Value of Studentized Range.

why this difference exists between the algorithms; a bias and variance analysis in each of them may help in finding the reason[4].

For the second experiment, we consider three factors instead of two: algorithms, training series and testing series. The three factor ANOVA has exactly the same assumptions as one way and two way ANOVA; and so, the results on verifying the assumptions in this second experiment is not shown here.

The database was split, one with 24,000 examples for training and the other with 20,000 examples for testing (both randomly chosen). Both, training an testing examples were also split into groups of 4,000 examples, that is, we got 6 groups of 4,000 training examples and 5 groups of 4,000 testing examples, look right side of figure 1. Each of the four algorithms was trained on the 6 training series and was tested in the 5 testing series, giving a total of 120 observations (4 algorithms x 6 training series x 5 testing series). This methodology produce 4 samples of 30 (6×5) observations each for the ANOVA experiment.

This experiment, as the first one, is a mixed effects one since the treatments are fixed, but the training and testing series are random. The interesting part of this experiment is that the interaction between the 3 factors is also considered (algorithms, training and testing), which was not present in the first experiment. The null hypothesis is the same as before; all errors are the same, Table 3[5].

Table 3. Three factors test of effects (A = Algorithm, T = Training, Te = Testing)

Source	SS	MS Num	DF Num	F Ratio	Prob >F
A	98.88	32.96	3	30.22	<0.0001
T	13.72	2.74	5	5.98	0.0025
A-T	6.03	0.40	15	3.10	0.0009
Te	9.70	2.42	4	2.77	0.07
A-Te	9.82	0.82	12	6.32	<0.0001
T-Te	3.72	0.19	20	1.44	0.1408

We can see that in the majority of the factors, the null hypothesis is rejected except in the factor testing, and in the factor interaction between testing and training ($T-Te$). This means that testing examples by themselves do not change the results. This may be due to the fact that the number of training examples is quite large (4,000) as to assure the goodness of the approximation to the true error of the algorithm. In other words, using 4,000 testing examples would not change significantly the mean error of the algorithm. As was mentioned earlier, with 5,000 testing examples we can be completely certain that the computed error is the true error of the algorithm, so with 4,000 the approximation is almost perfect [16]. Treatment effects are highly significant, and we can conclude

[4] The bias of an algorithm is the systematic error that cannot be eliminated even if equally size infinite training series are obtained [6] .

[5] SS : Sum of Squares, MS: Mean Square, DF : Degree of Freedom.

that each algorithm has a different effect. Training examples present a significant difference also, from here we can see that when varying the training series the generated classifiers are significantly different (even the classifiers generated with the same algorithm). This behavior is what Breiman called an "unstable" algorithm [4]. The summary of the analysis is shown in Table 4. We can see that the hypothesis of equality of means is also rejected (p-value < 0.0001). The average errors obtained in this experiment are: Bayes (17.52%), Chi S. (18.36%), Gain R. (15.92%), Gini (17.82%). Note that the Gain Ratio is the best.

Table 4. Analysis of variance as a whole for three factors

Source	DF	SS	Mean Square	F Ratio
Model	59	141.87	2.40	18.56
Error	60	7.77	0.13	Prob>F
C Total	119	149.63		<.0001

We perform two independent experiments. None of them is better than the other, they are simply two ways of approaching the ANOVA for the same problem; the reader can choose the one that seems more appropriate. There is no agreement amongst the AI community about how to compare learning algorithms [9]. Even statisticians have some problems in agreeing on the correct approach in complex experimental design problems [14].

5 Conclusions

The results obtained for both experiments are the same: in first place, algorithms are significantly different and, in second place, Gain Ratio is the best algorithm to predict if a person has an income greater to $50,000$ dollars. Therefore, we conclude that ANOVA is able to compare several learning algorithms and select the best algorithm for this particular database. In this case the best algorithm is Gain Ratio. It may happen that algorithms don't show differences, and in such a case, we could chose the fastest algorithm or the easiest to interpret (for example, a decision tree would be preferred over a neural net) .

ANOVA is an alternative technique that does not violate the statistical assumptions of commonly used tests. Three statistical assumptions for ANOVA must be satisfied and also a sufficiently large database should be used (to be able to break it into smaller databases). Although these requirements may seem burdensome and restricted only to domains that satisfy them, it is worth trying this technique when the degree of error is the most important factor to deal with. We propose a method to determine the best algorithm in a particular problem.

Experimental design was not developed with computational algorithms in mind, therefore, any application of the statistical techniques for this purpose must be seen as a mere approximation to the solution of the problem, but not as the only form to compare algorithms.

Choosing the best algorithm for a particular problem is an exploratory process; it depends directly on the knowledge the analyst has of the algorithms and the domain of application, hence involving some science and also some art. Fundamental research in *ML* is inherently empirical, because the performance of *ML* algorithms is determined by how well their underlying assumptions match the structure of the world [4].

There is not a general theory about how to choose an algorithm a priori. What we would like is to find a series of measures obtained from the data from which we could predict with a certain degree of confidence which algorithm will behave the best performance[11] but currently, this is not possible. If we want to prove that an algorithm is in general better than another (this is perhaps one of the fundamental questions in *ML* [4], [5]), a database used in the experiment should be chosen first, however, the question about which databases are representative is still unanswered.

References

1. E Bauer and R Kohavi. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine Learning*, 36:105–142, 1999.
2. G E P Box, W G Hunter, and Stuart. *Estadística para investigadores*. John Wiley and Sons Inc, 1988.
3. J L Devore. *Probabilidad y Estadística para Ingeniería y Ciencias*. Thomson and Learning, 2001.
4. T G Dietterich. Editorial. *Machine Learning*, 24(2):1–3, 1996.
5. T G Dietterich. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation*, 10(7):1895–1924, 1998.
6. T G Dietterich. Machine learning bias, statistical bias, and statistical variance of decision tree algorithms. *Machine Learning*, 1998.
7. T G Dietterich. Machine learning for sequential data: A review. In T Caelli, editor, *Structural, Syntactic, and Statistical Pattern Recognition*, volume 2396, pages 15–30. Springer-Verlag, 2002.
8. W Fan, H. Wan, P S Yu, and H. S. Lo. Inductive learning in less than one sequential data scan. In *Proc IJCAI*, 2003.
9. A Feelders and W Verkooijen. *Learning from Data: AI and Statistics V, chapter On the Statistical Comparison of Inductive Learning*. Springer-Verlag, 1996.
10. R Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proc IJCAI*, 1995.
11. R Kohavi. Data mining using MLC++. *Machine Learning*, 6(4):234–245, 1996.
12. D Michie, D J Spiegelhalter, and C C Taylor. *Machine Learning, Neural and Statistical Classification*. Ellis Horwood, 1994.
13. T Mitchell. *Machine Learning*. McGraw-Hill, 1997.
14. S L Salzberg. On comparing classifiers: A critique of current research and methods. *Data Mining and Knowledge Discovey*, 1:1–12, 1999.
15. G. Valentini and T. G. Dietterich. Bias-variance analysis of support vector machines for the development of svm-based ensemble methods. *Machine Learning Research*, To appear, 2004.
16. S M Weiss and C A Kulikowski. *Computer Systems that Learn*. Morgan Kaufmann, 1991.

Use of an ANFIS Network for Relative Humidity Behaviour Modelling on the South Region of Jalisco, México

Marco Gamboa[1], Marco Castro[2], and Francisco Herrera[3]

[1] Instituto Tecnologico de Ciudad Guzman, Departamento de Sistemas y Computacion, Av. Tecnologico #100 Cd. Guzman, Jalisco, Mexico
mgamboah@hotmail.com

[2] Instituto Tecnologico de la Paz, Departamento de Sistemas y Computacion, Boulevard Forjadores de Baja California 4720 La Paz, Baja California Sur, Mexico
ma_cl@hotmail.com

[3] Universidad Central "Marta Abreu" de las Villas, Facultad de Ingenieria Electrica, Carretera Camajuani S/N, Santa Clara, Cuba
herrera@fie.uclv.edu.cu

Abstract. This paper describes the application of an adaptive network using a neuro-fuzzy inference system called ANFIS for the identification of a relative humidity prognosis model on the south region of the state of Jalisco, México. The algorithm uses measurements obtained from meteorological stations as input-output signals and human expert's considerations on an if-then fuzzy knowledge base. The obtained model represents the system's dynamics. Figures showing the obtained model are presented.

1 Introduction

Currently, fuzzy modelling along with other techniques, such as neural networks, are being applied to identify systems not amenable to classical modelling techniques, due to the lack of precise, formal knowledge about them, their highly nonlinear behaviour, their degree of uncertainty, or their dynamic characteristics. One of the reasons for this is the capability of fuzzy systems to integrate information from different sources, such as physical laws, empirical models, or measurements and heuristics [1], and their ability to deal with qualitative information rather than solely relying on quantitative data.

Fuzzy modelling systems can be viewed as a special kind of Knowledge-Based Systems that uses a set of IF-THEN rules based on linguistic labels that represent the expert's knowledge about the system [2]. The part of the system responsible for interpreting the rules on the fuzzy knowledge base to, given the past input values, obtain the predicted output values is called an inference system. The proposed

© A. Gelbukh, R. Monroy. (Eds.)
Advances in Artificial Intelligence Theory
Research on Computing Science 16, 2005, pp. 75-84

model to determine relative humidity behaviour has its origins on the work by Takagi and Sugeno on fuzzy systems identification [3].

For a system with m inputs and one output on each instant of time t, it is possible to obtain a sample of the input and output data on that instant z(t) given by:

$$z(t) = [y(t)\ u_1(t)\ \ u_2(t)\ .\ .\ .\ \ u_m(t)]. \quad (\mathbf{1})$$

The model's size and complexity depends on the amount of input and output variables employed, and it is called the Order of the model.

Assuming a model structure defined by:

$$y(t) = g(\varphi(t),\theta). \quad (\mathbf{2})$$

Where y(t) represents the system's output on the instant t and the regressor vector φ(t) contains the ordered values of the input and output variables on t and on past instants (t-1, t-2….t-k.)

$$\varphi(t) = [\ y(t\text{-}1) \ldots y(t\text{-}k)\ \ u_1(t) \ldots u_1(t\text{-}k_1) \ldots \ldots \ldots u_m(t) \ldots u_m(t\text{-}k_m)\ ]. \quad (\mathbf{3})$$

The fundamental hypothesis is the existence of an unknown computable predictive function *g(.)* that applied to a set of regressor vectors allows us to determine the value of the future system outputs. Parametric model estimation techniques have been used to find a g'(t) function that can approximate within a certain precision degree using the available data so that:

$$y(t) = g'(t) + \ \ 't. \quad (\mathbf{4})$$

whereas v't represents the noises and effects produced by all those causal variables whether quantitative, qualitative, visible or invisible that where not included on the regressor set.

These estimation techniques are based on the assumption of a pre-established nonlinear form (Artificial Neural Networks and Fuzzy Rule Based Models) with excellent results. However, in prognosis work, the main focus of the scientific community has been on the use of connexionist systems (Artificial Neural Networks). And less attention has been paid to the exploration of adaptive fuzzy inference systems such as [4].

The present article describes the application of the ANFIS model over a series of meteorological data obtained on several stations located on the south portion of Jalisco State, and the way the identification algorithm is applied.

2 Structure of ANFIS

In classic set theory, a subset *S* on the universe *U* can be defined as a function that relates each element *x* of *U*, with an element of the discrete set {0, 1}[5].

$$S : U \bullet \{0, 1\}. \quad \textbf{(5)}$$

Where 0 is used to indicate than *x* does not belong to *S* at all and 1 indicates that *x* belongs completely to *S*.

On the other hand, in fuzzy set theory [6] μ is a membership function that can be valued on the continuum range [0, 1]:

$$:U \bullet [0, 1]. \quad \textbf{(6)}$$

where μ(u) represents the degree in which $u \in U$ belongs to the fuzzy set S. This is a generalization of the classical concept of a set (sharp set). μ is a set of ordered pair for discrete variables and a real function for continuous variables.

Certain families of functions are conventionally used to define the membership function, given their coincidence with the linguistic meaning of the most commonly used labels. Among the more frequent are:
The sigmoid function

$$S(u;\gamma,c) = \frac{1}{1+\exp[-\gamma(u-c)]}. \quad \textbf{(7)}$$

The Z function:

$$Z(u;\gamma,c) = 1-S(u;\gamma,c). \quad \textbf{(8)}$$

And the generalized bell function:

$$\Phi(u;\alpha,\beta,\gamma) = \frac{1}{1+\left|\frac{u-\gamma}{\alpha}\right|^{|2\beta|}}. \quad \textbf{(9)}$$

There are 3 main elements that constitute a fuzzy system: the input-output variables, the fuzzy set rule and the fuzzy inference system that defines the way the IF-THEN rules are used to obtain the output values given the current linguistic input variables values.

A neurofuzzy ANFIS network can be viewed as a fuzzy system conformed by Takagi-Sugeno rules [2], [7], [8]. On a system with two input variables x and y the ANFIS architecture can be represented as:

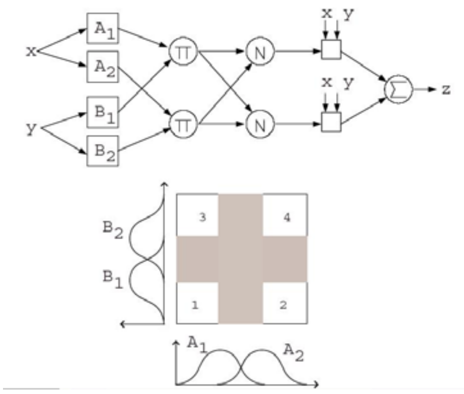


Fig 1: ANFIS architecture.

With a fuzzy rules base with the general structure:

if x is A1 and y is B1 then z = p1x + q1y + r1 (**10**)

if x is A1 and y is B2 then z = p2x + q2y + r2

if x is A2 and y is B1 then z = p3x + q3y + r3

if x is A2 and y is B2 then z = p4x + q4y + r4.

Where the x domain is delimited by fuzzy sets A1 and A2 and the y domain is delimited by fuzzy sets B1 and B2.

Four stages constitute the inference task: the first one is to calculate the membership functions µA1 (x), µA2(x), µB1 (y) y µB2 (y) using 9. The second task is to obtain the inference product wj = µA(x) · µB(y) for each rule. The third task is to obtain a weighted sum of each rule contribution to the outputs. And finally to evaluate the system's solution given by the following expression:

$$\sum_i \overline{w_i} f_i$$

In the rules base (10) the expert's knowledge among with other external considerations are represented in the form of fuzzy rules. It is to be noted that in 10 the antecedent part of the rule includes the same variables as the consequent; also, every variable domain is delimited by at least two fuzzy sets (Takagi-Sugeno form).

Although generally it is possible to use more complex functions in the consequent portion of the rules, usually a simple linear function is utilized to combine the entry variables.

The amount of fuzzy sets assigned to each independent variable, determines the ability of the ANFIS [4] algorithm to approximate non-linear functions and the algorithm's training technique to determine the parameters of the consequent functions [9]. However every time we increase the amount of fuzzy sets on the system, the amount of information required to train the network, and the computational cost of the process is considerably increased.

The fundamental aspect for the application of the ANFIS algorithm lies on the selection of the variables that will form the fuzzy rules, and the number of fuzzy sets assigned to each variable. The process of determination of an initial structure for the system has been investigated and data-mining techniques have been proposed by Jang [9].

3 The ANFIS Hybrid Training Method

Currently there are three main classes of fuzzy inference systems: Mandani, Sugeno and Tsukamoto[4] [11], which are fundamentally different on the structure of the consequent part of their IF-THEN rules, and because of this difference, they also differ from each other on their aggregation and defuzzification methods.

This paper centres on the Sugeno inference system. The ANFIS algorithm is the training routine that better adapts to this type of inference system. The learning rule is hybrid, given that it can apply the steepest descent method and back-propagation for non-linear parameters, and the least mean-squares methods on linear output parameters on the network.

The steepest descent consists on recursively obtaining a gradient vector on which each element is defined by the derived error function relative to a single parameter; this sole procedure can take a long while to converge. This series of iterations to find the gradient vector of an adaptive network structure is also called back-propagation because the gradient is calculated in the direction opposite to the output flow.

The combined use of both methods is effective for a fast identification of the adaptive network parameters. The iterations include a forward step and a backward step.

On the forward step the least mean square method is applied to calculate layer by layer, the output of each node on the net using the input signals vector, until a row corresponding to the matrix A and the vector Y, in the equation $A\Psi = Y$ is obtained. Where A is a matrix of m by n known functions of the input vector, Ψ the parameters to be esteemed vector of size n, and Y the size m output vector. This process is repeated for each pair of training data. The identification of the unknown parameter vector Ψ is obtained once the iterations for each row in A and Y are completed.

It is simple to determine that, when m=n and A isn't singular:

$$\Psi = A^{-1} y . \tag{11}$$

But, generally there are more input-output data pairs than parameters to adjust, thus m > n and a modelation error is generated so:

$$A\Psi + e = y . \tag{12}$$

and e is:

$$e = y - A\psi . \tag{13}$$

Our task consists in finding a value Ψ ($\hat{\Psi}$) that minimizes the sum of the square error defined by:

$$E(\Psi) = e^T e = (y - A\Psi)^T (y - A\Psi) . \tag{14}$$

Since E(Ψ) is in quadratic form, we have a unique minimum in ($\Psi \equiv \hat{\Psi}$) called the least mean squares estimator:

The back step uses back-propagation, where the error signals flow from the output nodes back to the input nodes. The gradient vector is calculated for each training data pair.

By the end of the processing of the training data, the non linear portion parameters are updated and the next forward step begins.

The proposed architecture uses five layers, the initial layer is conformed by the adaptive nodes that receive the input data. These nodes take that input data and apply the membership functions to each fuzzy set (high, low, tall,…etc.) to determine the degree to which they belong to those sets using a generalized bell function:

$$C1_i = \mu A_i(x) | i = \{1,2\} . \tag{15}$$

Where: $C1_i$ is the ith node output on layer 1, x is the node input and A_i is a linguistic label (low, medium, high, etc).
As the function parameters are changed, several membership functions are generated for each fuzzy set. Those parameters are called premise parameters.
The second layer is composed by fixed nodes also called "II nodes". The purpose of the second layer is to calculate the product of all the input signals ($C2_i$ = wi).

The importance of this layer is that each neurons output represents a percentile degree in which the rule is fulfilled.
The third layer calculates the fulfilment relation of the rule with the sum of fulfilment of all the rules. The output of this layer is a normalized this layers output (W_i^*) is called a normalized output.

$$\text{C3}_i = W_i^* = \frac{W_i}{W_1 + W_2} \quad \Big| \; i = \{1,2\} . \tag{16}$$

Each node on the fourth layer is an adaptive node with a function of the type:

$$\text{C4}_i = W_i^* fi = wi(Pix + qiy + ri) . \tag{17}$$

Where pi, qi and ri along with the set of parameters of those nodes, are called consequent parameters.

The fifth and final layer has an only node named $\sum$ that has as an output the sum of all its input signals.

$$\text{C5}_i = \sum_i W_i^* f_i = \frac{\sum_i W_i f_i}{\sum_i W_i} . \tag{18}$$

The forward step starts with the premise parameters and calculates the output of the nodes on the fourth layer and the consequent parameters of that layer are identified by the MCL method.

In the ANFIS architecture the values of the premise parameters are fixed, thus, the total output can be expressed as a linear combination of the consequent parameters.

This way, the hybrid training method guarantees that the consequent parameters are optimum under the fixed premise parameter condition. The hybrid training method converges a lot faster than the original back-propagation method because it reduces the search space. As a rule, the membership functions must be fixed while the process of training is taking place.

4 System Identification

The rules on a fuzzy system are initially determined by a human expert that uses his knowledge about the system for this task. However, when there is no expert available for this purpose, the number of membership function of each variable is assigned empirically. In such case the expert's task is limited to a validation of the initial considerations inferred by the algorithm, as well as the determination of the samples sizes used to train and to validate the model. It is also the expert's task to assess the quality of the identification.

The relative humidity readings on the south region of Jalisco, México, are used by the farmers to determine which products to seed, however these readings are obtained from the report generated by the Guadalajara City Airport, located 90 km away, thus the data has little reliability for the farming zone.

A meteorological station located a half kilometre of Ciudad Guzman (in the south central region of Jalisco) is equipped with sensors to monitor some weather parameters, among them, the temperature, radiation level and relative humidity. Those readings are stored in a database every 15 minutes.

A set of readings containing 34,945 measurements were obtained. Due to the periodicity of the data, 342 daily averages are obtained to form the training matrix (251) and the validation matrix (92). The variables included in the model were: Temperature (t-1), radiation (t-1), radiation (t), relative humidity (t-2), relative humidity (t-1) and finally, the relative humidity (t) as the output variable.

To establish the initial SID a structure of 10 membership functions, with the least possible amount of rules was tried, but with this structure, the identification presented unacceptable errors, due to the big amount of state variables with an influence on the system's behaviour that couldn't be modelled with a low number of membership functions.

The number of membership functions was increased to 20 with the generalized bell structure, and using the training data, a MatLab program using the Fuzzy ToolBox was developed. The initial phase of the program uses the genparam subroutine to automatically generate initial arbitrary values for the membership functions. Then the genfisl routine was used to make a grid fractioning of the input data. Out of this process the initial Sugeno fuzzy inference system is obtained. Once the initial membership functions are obtained, a 200 epochs training using the input-output data is performed through the Anfis subroutine. The results of this process are the modified fuzzy sets and the error plot.

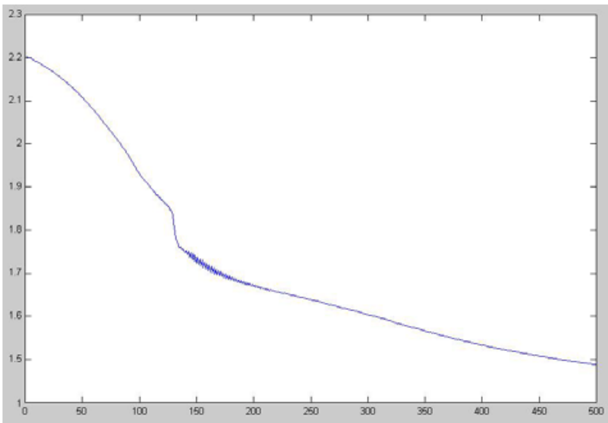


Figure 2

Still, the obtained error does not satisfy our criteria, thus a new training process is initiated using 400 epochs. Figure 2 shows that by the 130th iteration, the error is above 1.8, when the criterion is 1.5, but by the 400th iteration, the obtained value is 1.4. Since the grid partitioning defines that the number of rules is identical to the number of membership functions, we can obtain the 20 rules that identify the system with the showrule subroutine. On figure 3 the plotting of the identification results shows a total coincidence on the first 4000 chosen cases between the real process output (blue line) and the ANFIS identification output (green line).

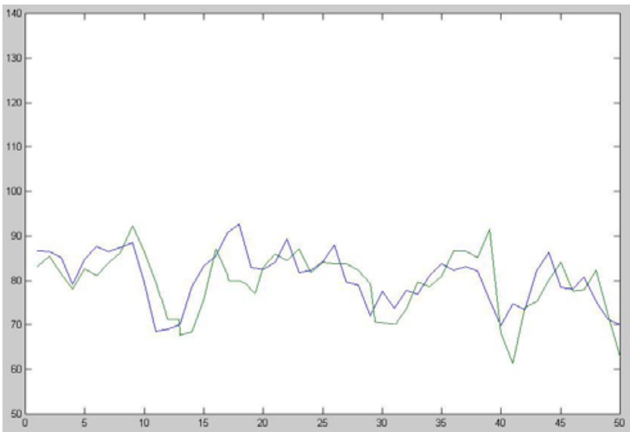


Figure 3

5 Conclusions

In this paper, the use of ANFIS identification with Sugeno fuzzy rules was discussed. Satisfactory results were obtained with the applied methodology to meet the needs of the farmers on the southern region of Jalisco. The results obtained are clearly better than the ones produced by the current system.

The advantage of this identification method lies on the fact that we need only a set of input/output measurements as initial data to identify non-linear systems. In the literature there are other examples of non-linear identification applications of ANFIS where a superior performance than classical methods and non hybrid artificial neural networks is obtained.

When looking to improve the identification structure it is convenient to pay attention to the effective fractioning (clustering) of the input space, this way we can reduce the number of rules and increase the speed of the identification process.

Thanks to their great adaptation possibilities, the generalized bell membership functions were successfully applied to very different input-output parameters.

A setback of this method is that when we encounter highly non linear systems, their identification demands an increased number of membership functions that make its on-line application very difficult due to the increased processing time needed to calculate the resulting parameters. On this case 20 membership functions where used and a time of 17 minutes of processing on a Pentium 3, were required to satisfy the error criteria.

References

1. Babuska, R.: Fuzzy Modeling for Control. 1 ed. International Series in Intelligence Technologies, ed. H. –J. Zimmerman. 1998, Boston: Kluwer Academic Publishers. 260
2. Wang, L. X.: Fuzzy System and Universal approximators. In: Poc. IEEE Int Conf. Fuzzy Systems, San Diego, Ca. Mar (1992)
3. Takagui T., Sugeno M.: Fuzzy Identification of systems and its applications to modeling and control". IEEE Trans. Syst., Man, Cybern, Vol. 15, 116-132 (1985)
4. Jang, J-S R. Anfis: Adaptative-network-based fuzzy Inference system. In: IEEE Trans. Syst., Man, Cybern, Vol. 23, 665-684 (1993)
5. Cox, E.: The fuzzy system Handbook, Second edition AP Professional, Paperback, 2nd Bk&Cd Edition, (1998)
6. Zadeth, L. A.: Fuzzy Sets. Information and Control, Vol. 8, 338-53 (1965)
7. Sugeno M., Kang T.: Structure identification on fuzzy model. Fuzzy sets and systems, Vol 28, 15-33 (1988)
8. Takagi, T., Sugeno, M. Derivation of fuzzy control rules from human operator's control actions. In Proceeding of IFAC Symposium on Fuzzy Information, Knoledge Reservation and Decision Analysis. Pg. 55-60 (1983)
9. Jang, J-S R.Input selection for ANFIS learning. In: Procededing of IEEE Int. conf. On Fuzzy Systems (1996)
10. Jang, J-S., Sun, C. T., Mizutani, E. : Neurofuzzy and Soft Computing. Math Lab Curriculum series. Prentice Hall (1997)

Information Security and Infrastructure

Multi-Agent System Design Based on Security Policies

Zeus Andrade Zaldívar, Ma. De los Ángeles Junco Rey, Jorge Adolfo Ramírez Uresti, José Arturo Tejeda Gómez, Enrique David Espinosa Carrillo

Instituto Tecnológico y de Estudios Superiores de Monterrey campus Estado de México.
Atizapán de Zaragoza, México.
{zandrade, ajunco, juresti, jtejeda, enrique.espinosa} @itesm.mx

Abstract. Security plays a major role in modern computer systems and multi-agent systems are not an exception. Security and multi-agent systems have been related on several works but little work has been done in relating security with agent oriented software engineering. In this paper we discuss the use of security policies to guide the design process for multi-agent systems and describe extensions to the GAIA methodology for this purpose. The proposed extensions are illustrated with the design of a file sharing system based on the MLS security model.

1 Introduction

Nowadays multi-agent systems provide a good alternative for the development of large and complex systems. Multi-agent systems are particularly useful when the system aims to recreate a real world organizational structure and when the presence of autonomous, intelligent, proactive, learning capable and social entities is needed to interact in distributed systems.

The majority of modern systems operate on open environments where they are susceptible to attacks, this makes necessary to take security as one of the more important elements of any modern system. A lot of research has related security and multi-agent systems; however, most of it had been focused on the final stages of the system development process, especially when the system is already implemented or even deployed. Identifying security requirements based on the system behavior instead of controlling the system behavior according to the security requirements is a hard task and commonly leads to the introduction of vulnerabilities in the system [1], [2], [3], [4].

Little work has been done focusing on the incorporation of security through the whole process of the system development. Mouratidis in [5] proposes extensions to the TROPOS methodology to accommodate security. Following this approach, in this paper we propose extensions to the GAIA methodology [6], [7] to accommodate security restrictions and we present recommendations to guide the design process according to security policies.

© A. Gelbukh, R. Monroy. (Eds.)
Advances in Artificial Intelligence Theory
Research on Computing Science 16, 2005, pp. 87-96

2 Extending the GAIA methodology

Security is established through policies and mechanisms that ensure integrity, privacy and availability of the resources that can be provided, required or present in the system. Security policies for multi-agent systems could be seen as sets of rules restricting the relationships between the system elements in terms of privacy, integrity and availability. The more important system elements are: Agents, users and resources.

There may be a lot of different types of rules according to the kind of entities they restrict or the properties they are based on. Rules can force an entity to perform an action, make an action triggers another, set conditions for an action, control the way an action is performed, control the number of entities and resources, or restrict the system behavior in many other ways. Thousands of different kinds of rules can be defined according to the specific needs and requirements of a system.

It is not realistic to think in a design methodology that takes all possible kinds of rules. For this reason in this work we take under consideration a limited set of rule types:

1. Restrictions on the effective capabilities of entities. Conditions that must be fulfilled to enable an entity to perform an action. These restrictions cover any kind of action performed by an agent or user such as reading, writing, creating or consuming resources and create, suspend, resume, migrate, and terminate an agent.
2. Restrictions over the interaction between entities. Conditions that must be fulfilled during the interaction of two entities.
3. Restrictions on the properties or attributes of entities. Conditions that restrain the attributes of an entity and that must be always fulfilled.

This set includes rules to represent and model policies like the multi level security [8] and the commercial model [9]. These models are representative on computer security, so the set of rules to represent them are also representative.

To illustrate the use of security policies on the design of multi-agent systems we use as a case of study a hypothetical distributed file access system based on Bell and LaPadula multi-level security model [8] (military security model). This security model is summarized in the following section.

2.1 Multi level security

This model establishes rules for two kinds of entities: objects, which are passive entities and cannot perform any activities, and subjects, which are active entities performing actions and accessing resources (objects). The model can be represented by the following rules:

R1. Subjects have only one level.
R2. Objects have only one level.
R3. All subjects have a set of labels.
R4. All objects have a set of labels.
R5. For every object there is a set of permissions for writing and reading.
R6. Reading is allowed if the subject's level is greater than or equal to the object's level and the subject's label set is a sub-set of the object's label set.

R7. Writing is allowed if the subject's level is lesser than or equal to the object's level and the subject's label set is a sub-set of the object's label set.
R8. An authentication mechanism is necessary
R9 . All subjects must be authenticated before accessing the system.

The primary objective of this model is to assure privacy. Through the mandatory and discretionary controls the model guaranties that information can be accessed only by authorized entities. Authorization is contained on the levels and labels on every entity.

2.2 Designing the system

The system to be consists of a shared file system based on agents and the multi-level security model. Although this system does not reflect the real complexity of most modern multi-agent systems, it has been chosen as a case of study because its simplicity allows a simple illustration of the design process based on security policies.

In the following sections we present the process of design for this system according to the phases established by the GAIA methodology and incorporating extensions and guidelines to preserve the security requirements (the security policy).

2.3 The analysis

In this phase of the GAIA methodology, the system specification is modeled through the identification of the organizational structure and the specification of preliminary roles, environment and interaction models. The security policy is part of the system specification so we propose to model it on this phase.

The first stage of this phase is to establish an organizational structure for the agents on the system. In many cases the security policy will define explicitly or implicitly the organizational structure or structures that should be used. For the case of study we identify a single organizational structure.

The GAIA methodology uses only the system requirements to identify and design roles, protocols and the environment, the first extension we propose to this methodology is to use a security policy to identify those elements. This is done through two new types of models, a restrictions model, which represents the security policy, and a behavior model, which models the system behavior restricted by the security policy.

The security policy restricts entities, so it must be modeled before modeling entities into roles and resources. The restrictions model must present all the rules or restrictions of the policy and the relationship between the rules and the entities (active and passive) of the system. The restriction model for the case of study is shown on figure 1.

The restrictions model presents all entities, resources, actions and interactions restricted by the security policy but there may be other elements present in the system requirements which are not restricted. The behavior model should represent graphically all restricted and unrestricted elements. On figure 2 and 3 we present a graphical notation and a behavior model for the case of study.

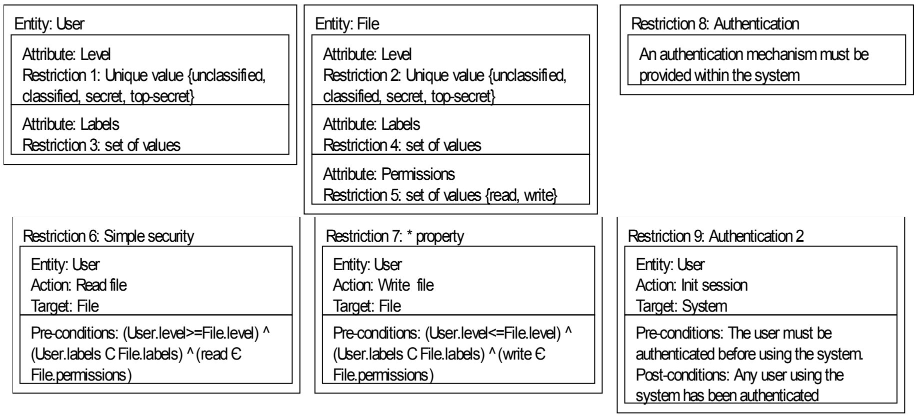


Fig. 1. Preliminary restrictions model

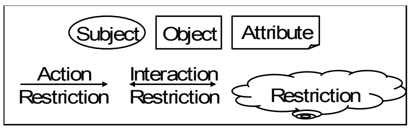


Fig. 2. Graphical notation for the behavior model

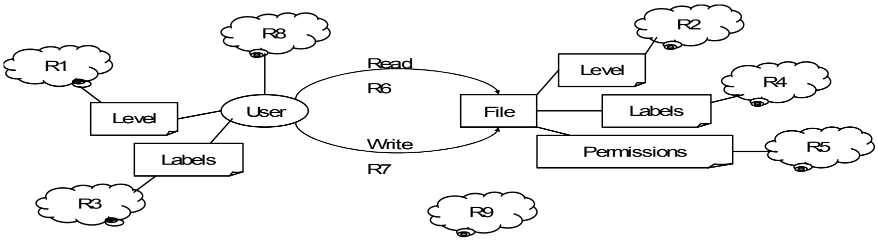


Fig. 3. Preliminary behavior model

Once we have established the restrictions and behavior models we can use them, along with the system specification, to generate the environment, roles and interaction models. Roles can be identified in the behavior model as subjects, resources as objects and interactions as double headed arrows connecting subjects.

An environment model, as proposed in [7], for the case of study is presented below:

```
READS: File, Users, Sessions
CHANGES: File, Sessions
```

The role model, as defined on [7], presents roles along with their description, actions, protocols, permissions and responsibilities. Permissions allow the designer to express the effective capabilities associated to the role and the safety responsibilities allow express conditions (invariants) that must always be maintained. However these two elements of the role schema are not useful to describe restrictions over the actions that the entity can perform. The second proposed extension is to use references to the restrictions defined on the restriction model instead of using permissions and safety responsibilities to describe the effective capabilities of the role. This extension does not apply to roles that are not restricted by the security policy. The following schema represents the role for the case of study.

Role schema: User
Description: Access the files in the system.
Protocols and Activities: InitSession, ReadFile, WriteFile, EndSession
Responsibilities: Liveness: USER = InitSession.(ReadFile | WriteFile)*.EndSession
Restrictions: R1, R3, R6, R7, R9

The interaction model consist of a set protocols defined by the interactions between roles. The interactions can be identified on the restrictions and behavior models, these interactions must be modeled as protocols. Restrictions over interactions are referenced in the schema for the protocols. The third proposed extension is to aggregate restriction references to the protocol schemas as defined in [7]. The interaction model for the case of study is shown in figure 4.

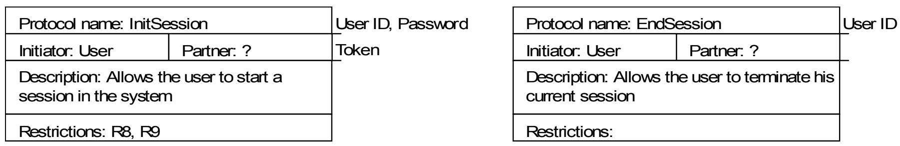


Fig. 4. Preliminary interaction model

The security policy, captured with the restrictions and behavior models define the security requirements of the system. The environment, role and interaction models are build parting from the restrictions and behavior models, the protocol and role schemas include references to the restrictions applied to the element they represent.

The last stage of the analysis is to establish the organizational rules. We propose only one recommendation for this stage, security restrictions that are not concerned to entities, their attributes, actions and interactions can and should be modeled as organizational rules. Organizational rules for the case of study are presented below.

```
IntiSession (User)[1]
IntiSession (User) → ReadFile (User)
IntiSession (User) → WriteFile (User)
IntiSession (User) → EndSession (User)
```

The GAIA methodology establishes as output of the analysis phase the preliminary models of environments, roles and interactions; we propose to add the preliminary models of restrictions and behavior (called preliminary because they are to be updated in the following phase).

2.4 The architectural design

This phase is focused on the establishment of the final structure for the system. On this phase all preliminary models are refined until they become final models.

The first stage of this phase is to define the organizational structure for the system. Security requirements (policies) diverge greatly from one system to another so the use of organizational patterns, as described in [7], may be restricted to systems that adopt the same security policy. The organizational structure for the case of study is presented underneath.

$\forall i$, User $\xrightarrow{\text{depends on}}$ Aunthenticator $\forall i$, User $\xrightarrow{\text{depends on}}$ FileManager

$\forall i$, User $\xrightarrow{\text{peer}}$ User $\forall i$, Authenticator $\xrightarrow{\text{peer}}$ FileManager

The next stage on the architectural design is to complete and refine the role model. The GAIA guidelines should be used during this stage; this means that new roles (organizational roles) must be added following the organizational structure, the new roles must be evaluated to verify that all restrictions on the security policy are fulfilled. The fourth proposed extension is the definition of security roles, this kind of roles are not explicitly needed by the organizational structure or the security policy but that are needed to satisfy some restrictions. Security roles can be identified through the protocols for which one or both entities (initiator and partner) were not identified in the preliminary interactions model, as an example, in the case of study we defined the protocol InitSession, which is restricted, in the preliminary interactions model the partner for the entity user was not identified and this identifies the need for a security role. Also, restricted actions like ReadFile and WriteFile imply the existence a security role which will be in charge of verifying the fulfillment of the restrictions associated to these actions. Below we present the role schemas for the new roles, which along with those defined on the preliminary role model integrate the final role model.

Role schema: Authenticator Description: Authenticates users trying to acces the system and manages a list of current users Protocols and Activities: InitSession, AuthenticateUser, EndSession Responsibilites: Liveness: AUTHENTICATOR = ((InitSession.AuthenticateUser) | EndSession $)^{\omega}$ Restrictions: R8, R9
Role schema: FileManager Description: Controls the access to the files in the system. Protocols and Activities: ReadFile, WriteFile Responsibilites: Liveness: FILEMANAGER = (ReadFile | WriteFile$)^{\omega}$ Restrictions: R6, R7

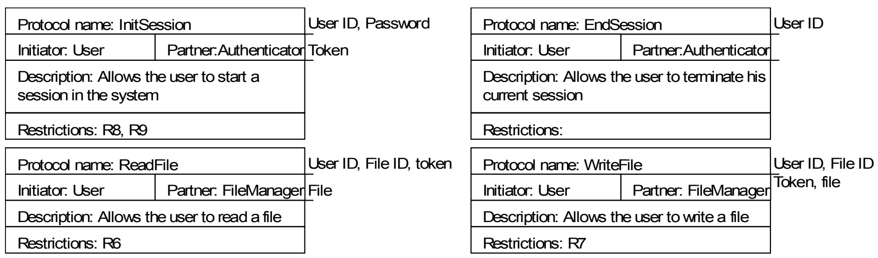


Fig. 5. Interactions model

The next stage is to refine the interactions model. The GAIA methodology provides the necessary guidelines to perform this action. New protocols may be added during this stage and it must be verified that they comply with all security restrictions present on the policy. In figure 5 we present the protocol schemas that complete the interactions model.

The aggregation of new roles and protocols could introduce inconsistencies with the security policy, thus a review and refinement of the restrictions and behavior models is necessary. The refinement of those models requires to evaluate all security restrictions and to modify them or create new restrictions to obtain a new security policy equivalent to the original but including all new roles and interactions. The refined models of restrictions and behavior are shown in figures 6 and 7.

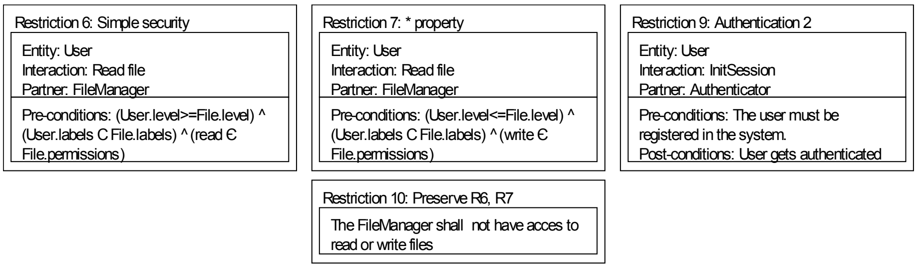


Fig. 6. Refinement of the restrictions model

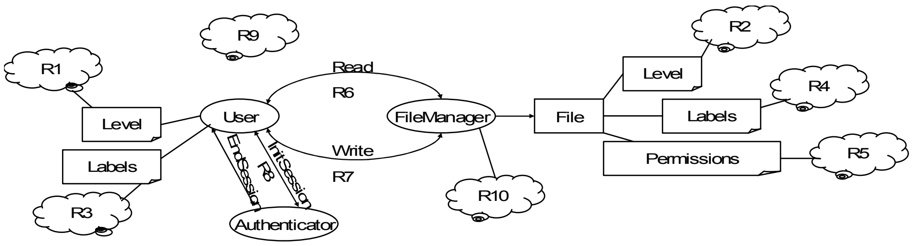


Fig. 7. Refinement of the behavior model

The design process must iterate on this phase until the roles, interactions, restrictions and behavior models become fully compliant with the system requirements and especially with the security policy.

2.5 The detailed design

On this phase the main objective is to find a match between agents and roles. Security restrictions play an important role during this phase because in many cases they define if it is possible for a single agent to play different roles. For our case of study, the agent model is presented below.

```
UserAgent  --plays-->  User
FileSystem --plays-->  FileManager, Authenticator
```

The fifth proposed extension is the inheritance of restrictions from the roles to the agent classes. Agent classes are designed to play one or more roles, so an agent class should inherit restrictions form the roles it plays. It is necessary to verify the application all restrictions, this can be done through a table containing all entities (agents and objects), their restrictions and how the restrictions will be applied. Table 1 presents these elements for the case of study.

Entity	Agent	Restriction	Aplication
User	UserAgent	R1	The attribute level restriction must be applied to the agent.
		R3	The attribute labels restriction must be applied to the agent.
		R9	The agent must perform an authentication procedure.
Authenticator	FileSystem	R8	The agent must provide an authentication mechanism.
		R9	The agent must authenticate any UserAgent agents and deliver a token if the authentication is succesfull.
FileManager	FileSystem	R6	The agent must verify the restriction before delivering the file to the UserAgent.
		R7	The agent must verify the restriction before replacing the file obtained by the user.
		R10	This agent should not be able to read or write files by it self.
File		R2	The attribute level and it's restriction must be applied to the object.
		R4	The attribute label and it's restriction must be applied to the object.
		R5	The attribute permissions and it's restriction must be applied to the object.

Table 1. System restrictions table.

The next step on the detailed design is to build a services model, on this model all activities, actions and interactions are detailed. This model can be constructed following the guidelines provided by the GAIA methodology. For the case of study, the services model is shown in table 2.

Service	Pre-conditions	Post-conditions	Inputs	Outputs
StartSession	The user must be registered in the system	The user gets a token and starts a session	User id and a password	A token for user
ReadFile	The token must be correct, the file must exist and R6 must be fulfilled	The file is readed by the user	User id, file id and a token	The file to read
WriteFile	The token must be correct and R7 must be fulfilled	The file is written by the user	User id, file id, new file and a token	None
EndSession	The user must have an open session in the system	The user session is terminated and the token invalidated	User id	none

Table 2. Services model

The detailed design is the simplest phase of the methodology, however, the implementation of the systems relays completely on the output of this phase, this implies that if any of the restrictions were lost during the previous phases or in this one, the implementation of the system will not be compliant with the security policy.

It is necessary to keep on the output of this phase the restrictions and behavior models because they are useful to guide the implementation process towards the compliance with the security policy.

2.6 The designed system

The obtained design consists on the final models: agents, services, restrictions and behavior along with the system restrictions table, the organizational rules and the organizational structure.

The security policy is present in all models. In the system restrictions table, every agent class has a set of restrictions inherited from the roles it plays and that must be implemented on the system. In the same way, all protocols have a set of restrictions to fulfill. The organizational rules preserve the rest of the security restrictions not directly associated with a role or protocol.

2.7 Implementing the designs

The GAIA methodology does not deal with the implementation process and being the basis for this work, the extensions proposed are limited to the design process.

There are many different agent frameworks which allow the creation and deployment of multi-agent systems. A combination of GAIA with the Jade framework [10] has been proposed in [11] and [12]. Jade also provides a security add-on based on rules [13] which is suitable for the implementation of systems designed following the extensions proposed.

Agent classes in the GAIA agent model can be directly mapped to Jade agent classes and its actions, protocols and services as Jade behaviors to be adopted by agents.

The security add-on for Jade includes features like authentication, permissions and secure messaging between agents. Authentication allows users to authenticate themselves with the system, secure messaging allows the use of cryptography to sign or encrypt messages during agent interactions and permissions allow or deny users and agents to perform certain actions like creating, killing, pausing or terminating an agent, sending messages to certain agents, accessing certain java classes, etc.

In most of the cases, rules from the restrictions model can be directly translated into rules from the Jade security add-on.

A software tool named MASSD has been created to allow an easy way to implement systems designed as proposed here. The application allows the user to generate all models here described and finally to automatically generate Jade code to implement agents and services.

3 Conclusions and future work

Security is one of the major issues to address when developing a multi-agent system. The overall system behavior must be controlled according to the security requirements of the system. On this paper we proposed the use of security policies to guide the design process of a multi-agent system and we presented a set of guidelines, recommendations, models, schemas and extensions to the GAIA methodology for this effect. We also established a relationship between the security policy, its restrictions and the elements of multi-agent systems and the necessary elements for modeling security restrictions within the GAIA methodology.

A system designed following the guidelines proposed and using the extensions described should be compliant with the security policy used. This contributes to reduce vulnerabilities in the system architecture. A real world test case for the benefits obtained with this methodology is part of the future work.

The presented work is focused on a representative set of rules or restrictions that may be present on a security policy, a generalization of the restriction types, the incorporation of validation mechanism to probe the preservation of the security policy and the study of different security models to establish security organizational patterns are part of the future work.

The systems designed following this work can be easily implemented using the Jade framework along with its security add-on. A software tool for automatic Jade agent code generation has been developed.

The work is being applied to generate a secure system architecture for the @lis technet project [14].

References

1. Lhuillier, N., Tomaiuolo, M., Vitaglione, G. Security in Multiagent Systems, JADE-S goesdistributed. exp: in search for innovation, vol 3. no. 3. Septiembre de 2003.
2. Jim Tam, J., Titkov, L., Neophytou, C. Securing Multi-Agent PlatformCommunication. Department of Electronic Engineering, Queen Mary, University of London.
3. Noordende, G., Brazier, F., Tanenbaum, A. A security framework for a mobile agent system. Ámsterdam University.
4. Kagal, L., Finin, T., Joshi, A. Developing Secure Agent Systems Using Delegation Based Trust Management. University of Maryland.
5. Mouratidis, H., Giorgini, P., Manson, G. Modelling secure multiagent systems. Proceedings of the AAMAS conference. 2003.
6. Zambonelli, F., Jennings, N., Wooldridge, M. Developing multi-agent systems: the gaia methodology. ACM transactions on software engineering and methodology. 2003.
7. Wooldridge, M., Jennings, N. R., Kinny, D. The Gaia methodology for agent oriented analysis and design. Journal of Autonomous Agents and Multi-Agent Systems, 3 (3):285–312, 2000.
8. Bell, D. E. LaPadulla, L. Computer security model: Unified exposition and multics interpretation. Technical report ESDTR-75-306. 1975.
9. Clark, D. D. Willson, D. R. A comparisson of militar and commercial security policies. Proceedings of the IEEE sysmposium on security and privacy. 1987.
10. Java Agent Development Framework. http://jade.cselt.it
11. Moraitis, P., Spanoudakis, N. Combining Gaia and JADE for Multi-Agent Systems Development. 4th International Symposium "From Agent Theory to Agent Implementation" (AT2AI4), in: Proceedings of the 17th European Meeting on Cybernetics and Systems Research (EMCSR 2004), Vienna, Austria, April 13 - 16, 2004.
12. Moraitis, P., Petraki, E., Spanoudakis, N. Engineering JADE Agents with the Gaia Methodology. In R. Kowalszyk, et al. (eds), “Agent Technologies, Infrastructures, Tools, and Applications for e-Services”, LNAI 2592, Springer-Verlag, 2003, pp. 77-91.
13. Java Agent Development Framework. Jade Security Guide. 2005.
14. @lis technet project. http://www.alis-technet.org.

An Artificial Manager for Security Policies in Organizations

Karen García-Gamboa[1] and Raúl Monroy[1] and Jesús Vázquez[2]

[1]Tecnológico de Monterrey, Campus Estado de México,
Km 3.5 Carretera al Lago de Guadalupe,
Col. Margarita Maza de Juárez, Atizapán de Zaragoza,
Estado de México, Mexico, 52926
Tel: +52 55 5864 5751
{A00472043,raulm}@itesm.mx
[2]Banco de México
Avenida 5 de Mayo, Col. Centro, Del. Cuauhtémoc
México D.F., Mexico, 06059
jjvazquez@banxico.org.mx

Abstract. Security policies are rules or conventions aimed at protecting the resources of an organization. Designing, implementing and maintaining security policies are all difficult error prone and time consuming tasks. We report on an e-policy manager, capable of expressing and reasoning about security policies, using (a subset of) first-order logic and closely following the work of [1]. The tool includes an interface that provides a graphical description of a set of policies. It also includes a link to an automated theorem prover, Otter [2], which is used to formally verify that the policies are consistent one another.

1 Introduction

Every organization should declare, and then adopt, a collection of security policies, for its resources to be protected. A *security policy* is a rule or convention that prescribes how to use a resource, keeping it away from an attack. If properly defined, these policies help ensuring the goals of security, namely: integrity, confidentiality, and availability of information. They may also help other purposes of the organization, including legal, regulation and contractual. Security policies should be applicable and easy to understand. A security policy should be like a program specification: it details what is to be done but not how.

Writing security policies is an error-prone task: policies are often ambiguous, they clash one another or they are simply outdated. Writing security policies is also very time-consuming, since it amounts to developing general plans that guarantee that the organization site will exhibit the intended behavior, even though it is under attack. This situation prompts the construction of a tool that can help a user designing and developing proper security policies.

© A. Gelbukh, R. Monroy. (Eds.)
Advances in Artificial Intelligence Theory
Research on Computing Science 16, 2005, pp. 97-106

This paper reports on one such a tool, we call *e-policy manager*, portraying the following features:

1. E-policy manager eases the capture of a security policy, while guaranteeing it conveys when someone is (not) allowed to carry out an action.
2. E-policy manager formalizes each security policy, using Halpern and Weissman first-order logic formulation [1], and then proves they are not inconsistent using the theorem-prover Otter [2].

E-policy manager deals only with security policies concerned with the protection of information files.

Paper overview

The rest of this paper is organized as follows: In Section 2 we describe how to express and reason about security policies using first order logic. We also give a toy example set of a set of inconsistent security policies and present how Otter [2] is used to pinpoint inconsistency. In Section 3 we present a graphical user interface for easily capturing security policies. In Section 4 we show results from a psychological validity test carried on the E-policy manager. In Sections 5 and 6 we respectively show related work and the conclusions drawn from our research.

2 Reasoning about Security Policies Using FOL

Formal methods advocate the use of techniques strongly based in Mathematics, with which we can formalize informally presented problems and provide rigorous arguments about their properties. To express and reason about security policies, we use (a subset of) first-order logic with equality over a vocabulary, as it has proven to be enough for this purpose [1].[1]

2.1 Permitting and Denying Policies

In this paper, security policies are either *permitting* (respectively *denying*) or *contextual*. A permitting (respectively denying) security policy conveys the *conditions* under which someone, the *subject*, is allowed (respectively forbidden) to perform an *action* on some *object*. Accordingly, the vocabulary is assumed to contain at least four collection of predicate relations, one denoting subjects (agents, processes, officers, etc.), other denoting objects (files, databases, applications, etc.), other denoting actions (read, write, execute, etc.), and other denoting constraints (time, roles, etc.)

A security policy therefore states a relation between a *subject* (users), an *object* (information files, databases, etc.) and an *action* (read, write, modify, etc.). The action is limited generally by the use of some access mechanism (a security mechanism). More formally, a (permitting) policy is a sentence of the form:

$$\forall X_1{:}T_1,\ldots, X_n{:}T_n.\ C \rightarrow [\neg]\mathit{permitted}(S, A) \quad (1)$$

[1] The reader is assumed to be familiar with the syntax and the semantics of First-Order Logic (FOL).

where C is a first-order formula, S and A are terms that, when valuated, return a subject and an action over some object respectively, and where [¬] indicates that ¬ may or may not appear in the formula. If ¬ does not appear in the formula, then the security policy is of type permitting. Throughout this paper, we shall respectively use $\forall X{:}T.\ P(X)$ and $\exists X{:}T.\ P(X)$ as an abbreviation of $\forall X.\ (T(X) \rightarrow P(X))$, read "all T's are P's" and $\exists X.\ (T(X) \wedge P(X))$, read as "some T's are P's".

Notice that we need some structure in order to be able to reason about the object upon which an action is carried out. To that purpose, we use function symbols. For example, we use *read*(f) to mean "read from file f". These are the function symbols of most frequent use:

- *read*(*f*) to mean "read from file f'";
- *write*(*f*) to mean "write onto file f'";
- *modify*(*f*) to mean "modify file f'";
- *delete*(*f*) to mean "delete file f'"; and
- *create*(*f*) to mean "create file f''.

2.2 Contextual Policies

A contextual policy is any first-order formula which states a specific view as to what counts as security within an organization. Four security properties may be included as an contextual security policy: i) integrity, ii) confidentiality, iii) availability and iv) authentication. Symbolically, these kinds of security policies do not change and so can be included or taken out at any time. Other contextual policies include *unicity*. For example, the predicate relating staff members and passwords is so that:

$$\forall X{:}staff.\ \exists Y{:}passw.\ (passw\_of(X, Y) \wedge \forall Z{:}passw.\ (passw\_of(X, Z) \rightarrow Y = Z))$$

2.3 The Environment

Called the *environment* [1], facts about the application domain, including properties of relations, are also captured using first-order logic. Given that E-policy manager was specifically designed to capture security policies of an organization, the environment comprehends relations for specifying objects such as department names, personnel members, etc. Employees, for instance, are all given a role and an affiliation, etc. For example, we write:

$$role\_of(X, Y)$$

$$affiliated2(X, Y)$$

to respectively mean "employee X, of type *staff*, has position Y, of type *position*" and "employee X, of type *staff*, belongs to department Y, of type *department*". Information has an owner and it is classified in terms of its relevance to the organization. For example, to respectively mean "information X, of type *info*, belongs to department Y, of type *department*" "information X, of type *info*, is in class Y, of type *class*" in symbols we write:

belongs2(*X*, *Y*)

of_class(X, Y)

Once completely captured, both the security policies and the environment, the context is all given to an automated reasoner. We have chosen to use Otter [2], since it is well-established both in industry and in academia. Using Otter, we verify that the model is not logically inconsistent, as well as querying the tool for specific capabilities of a user within the organization at hand.

It is well-known that first-order logic is semi-decidable. In our experiments, Otter was able to quickly find an inconsistency in the input security policies, (deriving the empty clause), if there was any, but usually spent a while, otherwise. Notice that, if the input security policies are not inconsistent, then Otter may run forever. Hence, sometimes we might not be able to find an argument supporting (respectively refuting) that the policies are consistent. To get around this problem, we would have to simultaneously apply Mace to the input security policies (see Section 5).

When asked a specific query about what a specific user can do, Otter also replied quickly. Our experiments confirm the theoretical results of [1]—our security policies are standard and not bipolar in their sense, and contain no inequalities in their antecedent—. If necessary, Otter can be manually configured so that a user (a security officer in this case) can select what methods should be applied in the search for the empty clause. If it derives the empty clause, Otter prints out a proof, which we use to automatically hint the user which security policies are thought to be in conflict.

Otter has been made to run in automatic mode, applying binary resolution, unary resolution or UR-Resolution, hyper-resolution, and binary paramodulation. Otter works searching for the empty clause, which in this case is an evidence of contradiction amongst the security policies.

2.3 An Example Set of Inconsistent Security Policies

In this section, we give a toy example of a set of inconsistent security policies and show how Otter is used to pinpoint inconsistency as well as a possible root of failure. The example is meant to illustrate that, since she is writing a number of policies, the security officer may not be aware of inconsistencies in a large document. Let us consider that the user introduces the following security policies:

1. Members of the IT department may read the information that belongs to that department, in symbols:

   $$\forall X{:}staff,\ Y{:}info.\ (affiliated2(X, it) \wedge belongs2(Y, it) \rightarrow permitted(X, read(Y))).$$

2. Members of the IT department cannot read the security procedures.

   $$\forall X{:}staff.\ (affiliated2(X, it) \rightarrow \neg permitted(X, read(security\_procedure))).$$

3. Security procedures are part of the IT department documents.

   $$\rightarrow belongs2(securiy\_procedures, it).$$

Once formalized, these policies are given to Otter for a consistency check. We conclude inconsistency, given that Otter finds the empty clause:

```
----------------------------------------- PROOF ------------------------------------------
1 []   x:staff   |  y:info
2[]    -affiliated2(x,it)   | -belongs2(y,it)   | permitted(x,read(y)).
3 []   x:info   | -affiliated2(x,it) | -permitted(x,read(security_procedures)).
4 []   x:info |  -belongs2(x,it)     |  belongs2(security_procedures,it).
-----> EMPTY CLAUSE at  0.25 sec ----> 4  [hyper, 1, 2, 3, 4]  $F. ---------------
--------------------------------------- end of proof ----------------------------------------
```

The example set of security policies is inconsistent, because policies 1 and 2 contradict one another. Policy 1 states that the staff members of IT are all allowed to read the IT information, while policy 2 states that they cannot read security procedures, which is IT information.

3 A Graphical User Interface

Writing, developing and maintaining security policies are all responsibilities of a security officer, who can not be assumed to be acquainted with formal methods. Formal methods require both significant skill and time (and therefore money) to use. To get around this problem, E-policy manager comes with a graphical user interface (GUI) that makes it easy to capture security policies by means of wizards and other graphical techniques. Also the GUI makes it easy both to correctly interpret the input security policies and to formalize them in FOL.

Through the GUI, a security officer can capture a set of security policies at a high, abstract level, by means of schemata. Although schemata restrict the expressiveness of our policy language, the output security policies contain the necessary ingredients for guaranteeing a simple but correct translation into our first-order language. Moreover, the GUI hides formal methods and knowledge representation out of the user, who no longer needs to be acquainted with these techniques.

3.1 Capturing a Permitting Security Policy

Given that a policy specifies the conditions under which a subject is allowed (respectively forbidden) to perform an action on some object, the GUI portrays a schemata, based on wizards, through which the user conveys several pieces of information. This involves the subject, e.g. users, processes, etc., the action, e.g. reading, writing, creating, etc., the object, an information file, the restrictions under which the action is permitted, e.g. a user role, a user affiliation, etc., and whether the action should be denied or permitted. The subject of a security policy can be a specific individual or a group of individuals that are related by some condition.

Figure 1 is a pictorial description of our security policy capturing schema. Users are allowed to write security policies giving objects the central role in the grammatical construction. Then, objects appear first, as illustrated in Figure 2.

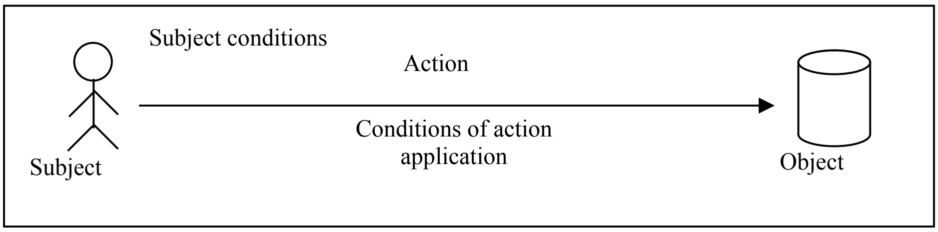


Figure 1. Security policy components

The GUI also allows the introduction of a modifier, we call the *purpose modifier*. If the purpose modifier is on, the interpretation of the security policy at hand is changed so that it now reads "*only* subject is allowed (respectively forbidden) to perform the associated action on the object under the given conditions".

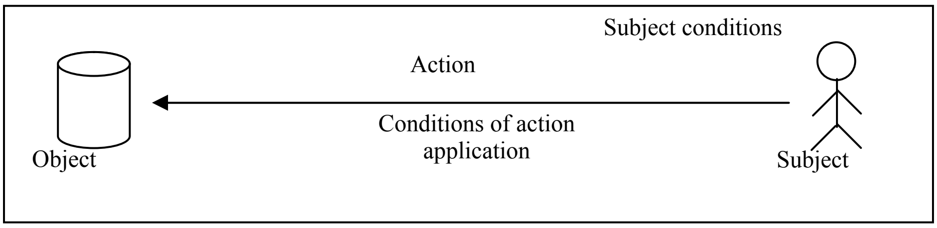


Figure 2. Security policies components (a different structure)

Using the GUI, a user is thus able to capture a security policy through a suitable schema. Schematically, the wizard enables the user to convey six pieces of information: i) the subject of the policy, ii) number of purposes (use only to denote exclusiveness, otherwise this field is left apart), iii) the type of policy (permitting or denying), iv) the action to be carried out, v) the object the action should be performed on, and vi) the constraints.

As an example policy capturing, consider that, after interacting with the wizard, the user has input the following information: `subject` =*staff*, `condition on the subject` = *officer*, `purpose modifier` = *on*, `policy type` = *permitting*, `action` = *read*, `object` = *passwords* (the password file), and `object constraints` = none, then the policy manager records the following formulae within the policy database:

$$\forall X{:}staff.\ (officer(X) \rightarrow permitted(X, read(passwords)))$$

$$\forall X{:}staff.\ (\neg officer(X) \rightarrow \neg permitted(X, read(passwords)))$$

As another example of security policy capturing, consider that, after interacting with the wizard, a security officer has input the following information: `subject` =

staff, `subject constraints` = *affiliated2*(*subject*, *it*), `purpose modifier` = off, `policy type` = *permitting*, `object` = *information*, `object constraint` = *belongs2*(*object*, *it*), `action` = *read*. Then, E-policy manager outputs:

$$\forall X{:}staff,\ Y{:}info.\ (affiliated2(X, it) \wedge belongs2(Y, it) \rightarrow permitted(X, read(Y)))$$

Each policy can be added or removed from the database by means of a wizard, which pops up a table containing all existing policies. A user only has to select an unwanted policy, by clicking on it, and then indicate policy removal or edition.

3.2 Linking the Environment with a Relational Database

The graphical user interface also drives a relational database. The user, making use of wizards, captures all the information related to the actual environment. For the application in mind, E-policy manager comes equipped with the following tables:

- Subject<identifier, name, department of affiliation, position>
- Object<identifier, name, department this object belongs to, class>
- Action<identifier, action description>
- Security_mechanism<identifier, access mechanism>

The relational database goal is not representing a security policy but providing the concrete information upon which the set policies act. Pretty much the same way in which policies are manipulated, the information hold in the database tables can be added, deleted, etc., via E-policy manager. When capturing a security policy, the schema (subject, action and object) can be filled with information retrieved from this relational database.

To provide flexibility to the user, we maintain a database of action synonymous and so a user may write (or select) “change”, “manipulate”, “alter”, etc. rather than the default “access”. To avoid slowing down the deduction process, we normalize all the synonymous of an action to a designated, default action name. This way, we do not perform additional, unnecessary applications of paramodulation or rewriting.

Using the capturing schema, each security policy is translated into both a first order formula and a semi-natural language expression. The formula is regarded as the formal model of the associated security policy. It is sent to Otter, if a consistency checking is to be performed. The expression is used for documentation purposes. It is part of the security policies manual of the associated organization.

The design of both the interface and the security policy schemata of E-policy manager were largely inspired in LaSCO [3]. LaSCO is an object oriented programming language which expresses a security policy by means of a constraint imposed on an object. Other policy languages were also considered, e.g. [4], but none of them provides as much a solid theoretical foundation as that of [1]. We have more to say about related work in section 5.

4 Testing Psychological Validity

E-policy manager was used to capture a number of security policies found in books or gathered from practitioners. Although they impose severe constraints on the policy language, schemata were found to be enough to capture all these security policies. Our experiments show that inconsistency checking may take a few milliseconds, if the security policies are contradictory, but may take a few seconds, otherwise. It is a theoretical result that FOL is undecidable; thus, in principle, inconsistency checking may not terminate at all.

E-policy manager was also evaluated for psychological validity. We run a test on six security officers, who answered a survey and interacted with the tool prototype. Our results from this experiment are encouraging. The answers provided by these security officers point that E-policy manager has achieved its two primary design goals, namely: i) to provide an graphical user interface that makes it easy to capture a security policy while guaranteeing it is correct in the sense of interpretation, and ii) to provide a means for formally verifying the security policies are consistent. Rather than an adverse opinion, we got words urging us to include an account for policies about other resources.

E-policy manager is available upon request by sending e-mail to the first author. It can also be downloaded from our URL, which we do not disclose here due to paper blind reviewing constraints. In the next section, we will review existing languages for the specification of security policies.

5 Related Work

A policy specification language aims at formalizing the intent of a policy designer into a form that can be read and interpreted by both people and machines [5]. It is especially designed to specify the relations amongst system entities in terms of actions and the conditions upon which these actions are denied or performed. There exist several policy specification languages. In what follows, we review the main features of 4 policy specification tools and associated languages: i) Keynote, ii) SPSL [5], iii) LaSCO [3], and the General Computer Security Policy Model [6].

Keynote and SPSL [5] are used to specify security policies about network applications. Neither Keynote nor SPL provide a visual tool for policy capturing. However, they are both equipped with a policy compiler, which produces a user profile that the intended application can use for denying or permitting the execution of an action. Keynote cannot be used to specify facts about the environment. Keynote does not scale properly, as it is difficult to foresee the state that results when enforcing a number of security policies. E-policy manager can be used to capture facts about the environment but was never thought as a mechanism for enforcing security policies.

LaSCO [3] is based on a model where a system consists of objects and events and works by conveying restrictions on objects. This language represents the policies by means of directed graphs which describe a specific state of the system (domain) and specific access constraints (requirements) and in mathematic logic. The nodes represent system objects and the edges represent system events. LaSCO [3] can be used to express a wide variety of standard and customized security policies, including access control and other history-based and context-dependent policies. Our work has been inspired in this language. For example, for the graphical user interface, we have adopted the use of graphs facilitating the security policies representation, as well as denoting information access control. LaSCO expressions can be translated into a low-level language for security policy enforcement. However, the tool does not involve the use of a mechanism to guarantee that the policies are consistent or that they meet certain properties.

Krsul, Spafford and Tuglualar [6] have presented a functional approach to the specification of security policies that allows policy stepwise refinement. The model makes the explicit assumption that policies and the value of the system objects are related. This model expresses policies as algorithmic and mathematical expressions. The specification policy explicitly lists the objects and attributes that are needed to enforce the policy. The model helps identifying the components that are relevant to the policy and hence provides a better policy understanding.

These languages are all adequate for the specification of security policies. However, they are not this effective, since, except for [6], they do not have a formal semantics, with which to reason about the security policies. Also security policy capturing using no visual aid has proven to be error prone, making it necessary to verify the written policies.

Halpern and Weissman have shown that (a subset of) first order logic is enough to express and efficiently reason about security policies [1]. They represent a security policy as a relation between three sorts, *Actions* (e.g. accessing a file), *Subjects* (the agents that perform actions) and *Times*. This contrasts with our work, where we denote a policy as a relation amongst *Subjects*, *Objects* and *Actions*. Halpern and Weissman are much more interested in using a user profile in order to enforce security policies; they argue that their security policy schema (which we have borrowed for our work) makes it easier for a user to write proper policies. They have not paid attention to checking policy consistency.

As we can see, our work is also based on Halpern and Weissman's. Indeed, in [1] it is mentioned these two researchers are working on developing a nice interface for security policy capturing, but no report has been published yet.

6 Conclusion and Future Work

To secure the most significant resources of an organization, it is necessary to have a set of appropriate policies. Managing security policies is not an easy task and currently it does not have computer support. The goal of our work is to provide a tool that supports this task and gives the bases for future research. The tool, called E-Policy manager, includes a graphical user interface that makes it easy to capture security policies and a module that formalizes these policies to be verified by a FOL automated theorem prover, Otter.

Our work in using FOL to formally represent and reason about security policies leaves plenty of room for improvement.

Further work includes simultaneously using Mace [8] and Otter, so as to obtain a conclusive result that takes into account the case of a consistent policy set (possibly making Otter run forever). Mace is a model generator which is usually used for searching a counterexample to an input conjecture. For our case, the counterexample will be (a subset of) the environment satisfying the input security policies.

Further work also includes using a natural language processor so as to allow a user to input security policies as he would in an informal document. This interface would significantly increment the acceptance of E-Policy manager from potential users.

References

1. Joseph Y. Halpern and Vicky Weissman. Using first-order logic to reason about policies. In *16th IEEE Computer Security Foundations Workshop*, Pacific Grove, 2003. IEEE Computer Society.

2. William McCune. Otter 2.0 In Mark E. Stickel, editor, *10th Conference on Automated Deduction*, volume 449 of *Lecture Notes in Computer Science*, pages 663-664. Springer, 1990.

3. James A. Hoagland. Specifying and enforcing policies using LaSCO, the language for security constraints on objects. *The Computing Research Repository*, cs. CR/0003066, 2000.

4. Renato Iannella. ODRL: The open digital rights language initiative. Technical report.

5. Minna Kangasluoma. Policy Specification Languages. Department of Computer Science, Helsinki University of Technology, 1999.

6. Ivan Krsul, Eugene Spafford, and Tugkan Tuglular. A New Approach to the Specification of General Computer Security Policies. COAST *Techical Report 97-13*., 1998, West Lafayette, IN 47907–1398.

7. Davies E., *Representation of Commonsense Knowledge*, Courant Institute for Mathematical Sciences, 1990.

8. William McCune. Mace 2.0 Reference Manual and Guide, *Mathematics and Computer Science Division Techical Memorandum No. 249*. Springer, 2001.

A Service-Oriented approach for integrating MultiAgent Systems with Web Services in a Collaboration application.

Eduardo H. Ramírez and Ramón F. Brena

Tecnológico de Monterrey
Centro de Sistemas Inteligentes
{eduardo.ramirez,ramon.brena}@itesm.mx

Abstract. Service-oriented architectural paradigm (SOA) improves the abstraction and flexibility on which information systems may be designed and integrated, thus allowing agent-based and other AI applications to be gradually and seamlessly adopted by large organizations. Web-services standards for semantics and interoperability realize the SOA approach and are key enablers for MultiAgent systems enterprise integration. In this paper we show how the SOA paradigm and Web Services technologies may be used for designing and integrating agent-based applications in a flexible and robust way. A case-study for a collaborative application is presented.

1 Introduction

The Service-Oriented architecture paradigm (SOA) improves the abstraction and flexibility on which information systems may be designed and integrated. SOA foundation ideas were introduced by Arsajani[2] and sharply defined as: "the architectural style that supports loosely coupled services to enable business flexibility in an interoperable, technology-agnostic manner. SOA consists of a composite set of business-aligned services that support a flexible and dynamically re-configurable end-to-end business processes realization using interface-based service descriptions[5]".

There exists three roles or actors in a SOA that are shown on figure 1. A *service consumer*, that locates a *service provider* in a *service registry* or broker, then binds the service description and finally performs an invocation to the provider. The loose coupling principle suggest the operations between actors to be realized interchanging as little information as possible, usually through message passing.

Even though the SOA architectural style is not bound to any particular implementation technology, the Web Services standards are becoming a natural and common choice. For the purposes of this work, we understand a Web Service as "a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL [24]). Other systems interact with the Web Service

© *A. Gelbukh, R. Monroy. (Eds.)*
Advances in Artificial Intelligence Theory
Research on Computing Science 16, 2005, pp. 107-116

in a manner prescribed by its description using SOAP-messages [25], typically conveyed using HTTP with an XML [23] serialization in conjunction with other Web-related standards."[26].

Close relationships between Web Services and Software agents[14] exist at the semantic and interoperability level. Since the conception of the Semantic Web[4] due to its semantic processing capabilities, agent technologies were strongly proposed as Intelligent Web Service consumers[10], being able to discover, compose and verify Web Services[27]. On the other side, agents are also a candidate technology to deliver knowledge-intensive Web Services[1][11]. However, in order to allow agents to become intelligent Web Service providers, a number of technical issues should be addressed.

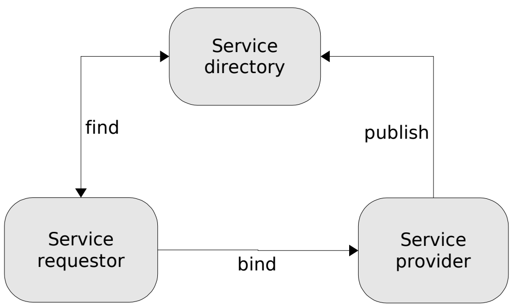


Fig. 1. Roles in a service-oriented architecture

2 SOA for MultiAgent Systems

The SOA approach and Web Services technologies could be used to overcome the technical limitations required to integrate a MultiAgent System into an enterprise service-oriented environment. In a similar way, the SOA ideas can be translated to the application internals to allow agents to provide a stable framework for building robust Agent-based applications. A number of recent works report how the SOA approach have been applied in several AI and multi-agent systems and components[17][19][27].

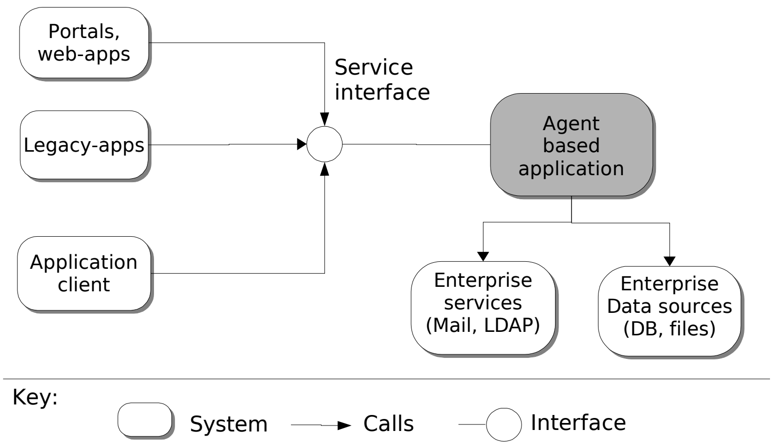


Fig. 2. Decoupled architecture top-level view

2.1 SOA for agent-based application integration

In conformance with the SOA ideas, the main architectural principle for integrating a MultiAgent system into an enterprise environment consists of decoupling applications through the exposure of "coarse-grained" service interfaces. As shown in figure 2, the underliying metaphor used to determine the design strategy was the aim to create a "black-box" in which agents can live and perform complex knowledge-intensive tasks. Neither enterprise applications nor its developers should be aware that a service is provided by agents if the system offers a standard SOAP endpoint as interface.

Several integration architectures for agent-based application exists[22],[7], however a particular approach, namely the "Embedded Web Service Architecture (EWSA)" has proven to be enough simple and effective for this purpose with a clear focus on agent-to-application communication. The proposal's main idea suggests embedding a Web Server into an agent-based application and defining a interaction model upon which software agents and Web-components[20] may communicate in order to provide services. Details on the approach can be found on related work[18].

Once the integration problem is solved, the task of designing an adequate set of services is a classic software engineering problem and remains a particular challenging activity by itself. Each agent-based application, may offer a different set of agent-based web services. Some examples of knowledge-intensive services from one of our agent projects[6] are:

Recommendation services A user's profile is represented by a set of points in the taxonomies, as each user could have many interests and could be located

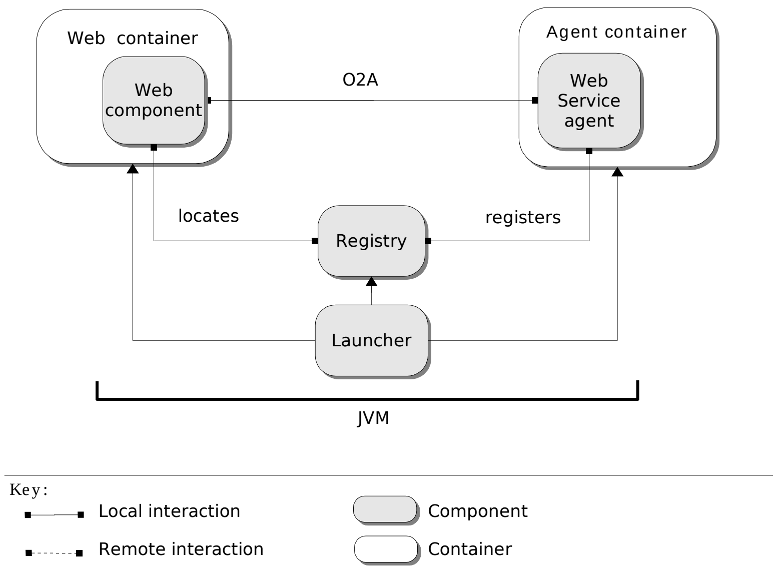


Fig. 3. Integration architecture component view

at different parts of the organizational structure. As the system keeps track of user interests and preferences it is able to recommend content to users on demand. Recommended content may be used in Portals or Web applications.

Content search and classification A service agent that searches the most relevant documents on particular data-sources can be constructed. The knowledge that guides the search is handled by the ontology agent where the keywords with which the search engine is invoked are defined. The documents obtained by the search are qualified by a fuzzy system and then the best ones are delivered to the users.

Subscription and alert services The system allows users to subscribe to changes in specific areas. Also, users may customize the media and frequency of notifications. Rules may be defined so as messages relative to certain topics are handled with higher priorities. A rule may state that several alerts may be sent to their cell-phone via SMS, and also define that interest-area messages be sent in a weekly summary via email.

Content distribution services Enterprise applications may deliver content to the system using its semantic-based content distribution services. When new content is received it is classified and distributed to users who may be interested. Users receive the notifications of new content as specified by their own rules.

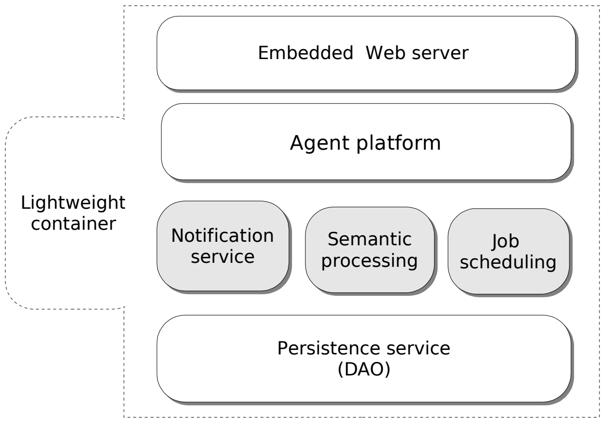


Fig. 4. Platform services framework, layered component view

2.2 A Service-Oriented framework for MultiAgent System design

Besides application integration, the service-oriented approach have gained momentum for intra-application component communication. The SOA paradigm have been reinforced by a new generation of non-invasive lightweight frameworks[16],[15] that leverage contemporary software design principles such and allow application developers to produce enterprise-level, yet simple applications. The use of a lightweight framework like Spring[12] helps agent project teams to decompose application in simple components that access and reuse a collection of platform services, thus simplifying programming and allowing a greater abstraction to the specific agent development.

A generic application architecture is shown in figure 4, where some general-purpose services may be provided by robust third-party tools. Each of the boxes in figure4 represents a platform service that is configured and instantiated by the lightweight container, the container also fulfills the "service registry" role described in section 1. Each platform service is composed of a particular kind of application components concerning a specific technical domain. The layer metaphor gives an insight about the abstraction level of each application tier, where high-level service components (i.e. web-components and agents), consume the services of low-level components (i.e. data-access objects).

In the presented example (figure 4), there exists generic (Web, Agent, Persistence) and custom services (shaded) like the Scheduling and Notification services. For this specific case they were defined and implemented as follows:

Embedded Web Server . Implemented using the Tomcat Web Server[13], it provides application with the capability of receiving HTTP requests and turning them to a particular Web-component or Servlet in conformance with the JSR-154[20] specification.

Agent platform . The execution environment for the agents provided by the agent platform in conformity with FIPA[8] specifications. Particularly implemented with the JADE[3] platform.

Scheduling service . Allows the agents and users of the applications to schedule of periodic or time-triggered execution of custom jobs using the Quartz Scheduler[21]. Job components are defined using an application specific hierarchy of tasks and events (i.e. CheckMail, CheckAppointments, StartWorkflow etc.).

Notification service . Allows the agents and web-components to access the enterprise notification infrastructure, namely the mail service using the JavaMail API and several SMS systems. Notification components are the drivers for the distinct enterprise notification providers.

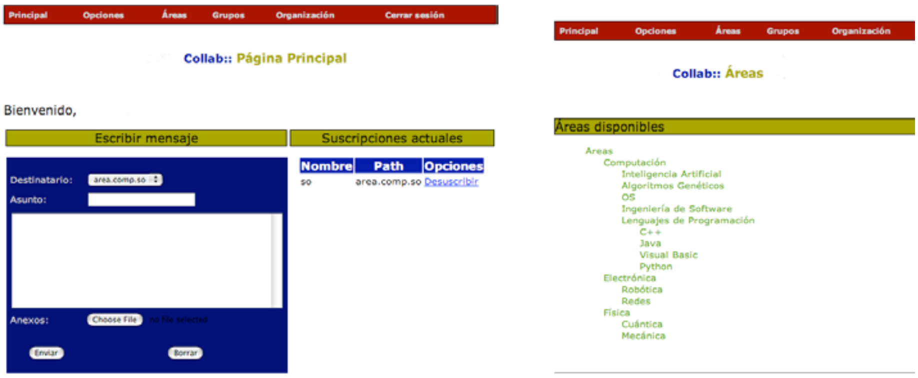
Semantic processing service . Provides access to a set of classification and inference tools like the Jess rule engine[9]. Allows agents to perform simplified queries to rule bases using native Java calls.

Persistence service . Implemented using the JDBC API and the components in conformance with DAO pattern, it is a low-level platform service that provides a simplified object access to enterprise data-sources to the servlets, agents, jobs and other components in the application. It also offers access to directory services such as LDAP for user authentication and profile synchronization.

Following this approach, the particular agent-model of the system is designed and implemented on the Agent platform, enabling agents with the capability of programatically invoking any of the underliying platform services. We believe that the use of such a framework becomes critical in agent projects where the agent developers don't share the same skill-set and concerns of enterprise application developers, then, the use of service-oriented principles (loose coupling, coarse grained interfaces) helps heterogeneous development teams to achieve a clear division of work. One major drawback of the model is that imposes some learning curve to the development teams which could lead to delays in the initial development cycles. Even tough the individual component developer learning scope gets reduced, all developers should have a clear understanding of the overall system architecture beyond de agent model, and of the libraries and interfaces that should be reused in order to interoperate with the underliying infrastructure services.

3 A collaborative application case study

Web-applications are widespread tools for collaboration support in large organizations, however, its traditional style of development and architecture hardly allows for a more sophisticated intelligent service to be included. For this kind of cases, the SOA-integrated agent system may be a knowledge service provider. A prototype groupware application was developed including the following standard sets of features:



(a) Message and file sharing (b) Group browsing

Fig. 5. Collaborative application prototype

- Users may create groups and join to existing groups. See figure 5(b).
- Users may browse the member list and conversations history of each group.
- Within groups users may exchange messages and share files. See figure 5(a).

3.1 Implementation of a recommendation service

Trivial features of a collaborative application could be enhanced by agent-based web services, for example, consider the simple scenario in which a user joins a workgroup using the collaborative application interface. After this event, the system will send recommendations about documents related to group activity and interests on a daily basis. Some of the internal services interaction needed to fulfill this case are:

- Once the user joins the group, the collaborative application will invoke the agent-based service using a standard coarse-grained interface.
- Periodical updates to the document base will be programmed to be triggered upon certain conditions using the agent-platform scheduling service.
- Information gathered from data-sources and repositories will be classified and filtered against the interests of the relevant groups using the semantic-processing and classification services.
- Relevant information will be distributed to users using the agent-based notification services, the system may leverage the use of several distribution media (Email, SMS) to alert users of new documents and other application events according to its relevance.

The components and services interactions for the use-case are shown in detail in figure 6. It should be noticed that the use of generic services for Scheduling, Notification and Classification allows the task-specific code (read documents,

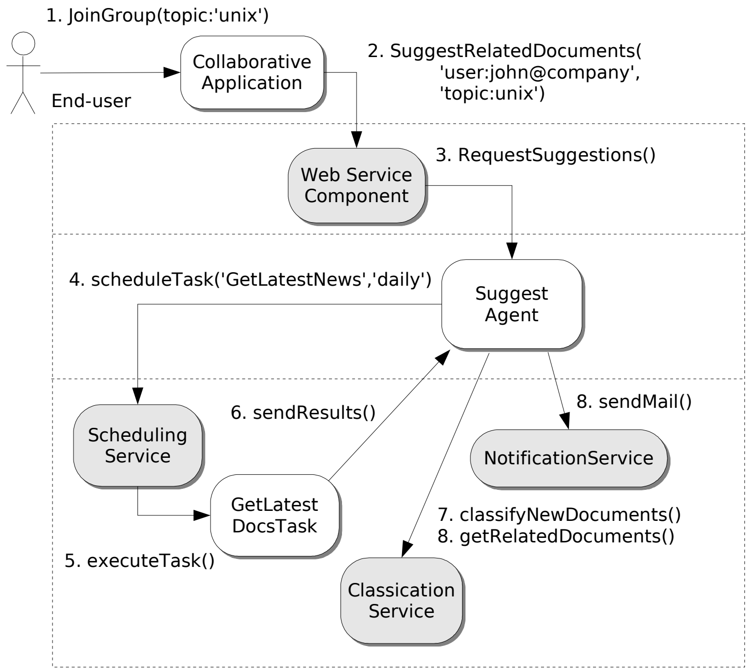


Fig. 6. Use case enhancement using agent based services, detailed view

send mail) to be encapsulated in the lower layers of the system, thus enhancing the abstraction level for the agent layer.

4 Conclusions

In this paper we have shown how the SOA paradigm and Web Services technologies may be used for designing and integrating agent-based applications as service providers in enterprise software environments. Also, using the same design principles we have described a generic framework for intra-platform services for implementing robust and reliable MultiAgent systems reusing high-quality open source components like Spring[12], Jade[3], Quartz[21] and Tomcat[13]. Also we have presented a case study on a simple collaborative application that shows the kind of feature improvements that can be done by the integration of agent-based web services.

Future works we intend to do on collaborative features enhanced by agent-based web services are:

– Taxonomy-based user profiles: User profiles are defined using several standard taxonomies like the organizational structure roles (which are assigned to user by management); workgroups that may be freely created upon project requirements; and interest areas to which users get subscribed according their professional or personal interests

- User-defined messages: Users of a workgroup or a content area may send custom messages. The message receivers are not selected by name rather than by their role in the organization or their interest areas. A semantic matching of message receivers may be produced using the web services.
- Semantic file sharing: Users of the system may share documents to taxonomy-based workgroups via the web-based interfaces. When a document is uploaded or updated, users subscribed to taxonomy child nodes will be notified according to their preferences. Example: A Unix tutorial is uploaded, so users in the general Computing area, the more specific Operating Systems sub-area, and the specific Unix area are notified of the addition. Also, when a new file is submitted, it may be classified and automatically distributed to a number of relevant groups outside the original receiver.

References

1. Alun Preece and Stefan Decker. Intelligent web services. *IEEE Intelligent Systems*, 17(1), Ene-Feb 2002. `http://www.csd.abdn.ac.uk/~apreece/research/download/ieeeis2002.pdf`.
2. A. Arsanjani. A domain-language approach to designing dynamic enterprise component-based architectures to support business services. In *39th International Conference and Exhibition on Technology of Object-Oriented Languages and Systems, 2001. TOOLS 39*, pages 130 – 141, Aug. 2001.
3. Fabio Bellifemine, Agostino Poggi, and Giovanni Rimassa. JADE - A FIPA-compliant agent framework, `http://jade.cselt.it/papers/PAAM.pdf`. In *Proceedings of PAAM'99, London*, 1999.
4. T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American*, 284(5):35–43, 2001. Essay about the possibilities of the semantic web.
5. Bernhard Borges and Kerrie Holley and Ali Arsanjani. Service-Oriented Architecture, Aug. 2004. `http://webservices.sys-con.com/read/46175.htm`.
6. R. Brena and J. Aguirre. Just-in-Time Knowledge Flow for Distributed Organizations using agents technology. In *Knowledge Technologies 2001 Conference, Austin Texas*, Mar 2001.
7. Dick Cowan and Martin Griss et al. Making Software Agent Technology available to Enterprise Applications, `http://www.hpl.hp.com/techreports/2002/HPL-2002-211.pdf`. Technical Report HPL-2002-211, HP Labs Technical Reports, 2002.
8. Foundation for Intelligent Physical Agents. FIPA Abstract Architecture Specification, `http://www.fipa.org/specs/fipa00001/`, 2002.
9. Ernest Friedman-Hill. Jess, the rule engine for the java platform, 2005. `http://herzberg.ca.sandia.gov/jess/`.
10. J. Hendler. Agents and the Semantic Web, `http://citeseer.ist.psu.edu/hendler01agents.html`. *IEEE Intelligent Systems*, 16(2), 2001.
11. Michael Hunhs. Agents as Web Services. *IEEE Internet Computing*, 6(4):93 –95, Jul-Ago 2002.
12. Interface21 Limited. The Spring Java/J2EE application framework, `http://www.springframework.org/`, 2005.

13. Jakarta Project - The Apache Software Foundation. The Tomcat Web Server v. 4.1, http://jakarta.apache.org/tomcat, 2003.
14. N. Jennings and M. Wooldridge. Software agents. *IEE Review*, 42(1):17 –20, 18 Jan. 1996.
15. Rod Johnson. *Expert One-on-One J2EE Design and Development*. Wrox Press Ltd., Birmingham, UK, UK, 2002.
16. Rod Johnson. J2EE development frameworks. *Computer*, 38(1):107–110, 2005.
17. G.F. Laforga and R.F. Romero. A services-oriented architecture applied to artificial neural network. In *9th International Conference on Neural Information Processing, ICONIP '02*, volume 5, pages 2650 – 2654, Nov, 2002.
18. E. Ramirez and R. Brena. Web-enabling MultiAgent Systems. *Lecture Notes in Computer Science, IBERAMIA 2004*, 3315, 2004.
19. H. Schuschel and M. Weske. Automated planning in a service-oriented architecture. In *13th International IEEE Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, WET ICE 2004*, pages 75–8, June 2004.
20. Sun Microsystems, Inc. JSR-000154 Java(TM) Servlet 2.4 Specification (Final release), http://jcp.org/aboutJava/communityprocess/final/jsr154/index.html, 2003.
21. The OpenSymphony Group. Quartz Enterpise Job Scheduler, http://www.opensymphony.com/quartz/, 2005.
22. Whitestein Technologies, A.G. Web Services Agent Integration Project, http://wsai.sourceforge.net, 2003.
23. World Wide Web Consortium (W3C). Extensible Markup Language (XML) 1.0 (Second Edition), http://www.w3c.org/TR/2000/REC-xml-20001006, 2000.
24. World Wide Web Consortium (W3C). Web Services Description Language (WSDL) 1.1, http://www.w3c.org/TR/wsdl, 2001.
25. World Wide Web Consortium (W3C). Simple Object Access Protocol, http://www.w3.org/TR/SOAP, 2003.
26. World Wide Web Consortium (W3C). Web Services Glossary, Working Draft, http://www.w3c.org/TR/ws-gloss/, Aug 2003.
27. S.J.H. Yang, B.C.W. Lan, and Jen-Yao Chung. A New Approach for Context Aware SOA. In *The 2005 IEEE International Conference on e-Technology, e-Commerce and e-Service, EEE '05*, pages 438 – 443, March 2005.

Anti-Spam Filtering and Email Virus Detection

A Collaborative Learning Method For Spam Filtering

Hsiu-Sen Chiang, Jui-Chi Shen*, Dong-Her Shih, and Chia-Shyang Lin

National Yunlin University of Science and Technology, Department of Information Management, 123, Section 3, University Road, Touliu, Yunlin, Taiwan
Kaohsiung Hospitality College, Sung-Ho Rd., Shiao-Kang. Kaohsiung, Taiwan *
{g9023728,shihdh,g9223728}@yuntech.edu.tw

Abstract. Spam, also known as Unsolicited Commercial Email (UCE), is the bane of email communication. It has brought enormous cost for the companies or users that use Internet. Spam filtering has made considerable progress in recent years. The predominant approaches are data mining methods and machine learning methods. Researchers have largely concentrated on either one of the approaches since a principled unifying framework is still lacking. This paper suggests that both approaches can be combined under a collaborative learning framework. We propose a collaborative learning algorithm that parallelly uses three different machine learning methods. The resultant algorithm is simple and understandable, and offers a principled solution to combine content-based filtering and collaborative filtering. Within our algorithm, we are now able to interpret various existing techniques from a unifying point of view. Finally we demonstrate the success of the proposed collaborative filtering methods in the experiment.

1 Introduction

Unsolicited Bulk Email (UBE), also referred to as Unsolicited Commercial Email (UCE), is commonly called spam or junk mail. Spamming is the practice of sending mass mailings to large numbers of people who have no relationship with the sender and who didn't ask for such mail. Different reasons motivate spammers, but the spam exists primarily because of the low cost. Spam filtering denotes a family of techniques that help users to find the right emails while filtering out undesired ones. The filter can be implemented at either server side (mail transport agent, MTA) or the end user side (mail user agent, MUA).

In this paper, we parallelly employ three methods to the filtering work with our collaborative learning algorithm. It is not easy to judge whether a mail is spam or not by any single one method. That is because under a single one method, some features occurred in spam will also occur in legitimate mail. There are many technologies and researches on spam mail detection, most of them use a single client agent to filter mail, and lack of the ability of collaboration. The junk mail has the characteristic that massively spreads, but the present client filter agents don’t fully utilize this characteristic. Hence, we propose a distributed architecture to learn collaboratively with the knowledge of spam in improving the ability of client spam detection.

© A. Gelbukh, R. Monroy. (Eds.)
Advances in Artificial Intelligence Theory
Research on Computing Science 16, 2005, pp. 119-128

2 Related Work

Spam filtering has made considerable progress in recent years. Many data mining and machine learning researchers have worked on spam detection and filtering. The problem is popular enough that it has been the subject of a Data Mining Cup contest [4] as well as numerous class projects. Bayesian analysis has been very popular [1; 14], but researchers have also used SVMs [5], decisions trees [19], rule learning [3; 18] and even genetic programming [12]. Some widely used methods for anti-spam are list as below:

Blacklist. A blacklist spam filter can be a DNS-based (DNS-based Blacklists, DNSBL) or email-address-based blacklist. The mechanism behind the method is keeping the source of spammers in a database. The legal mail server can access to the database then deny receiving the messages from the source of spam. Blacklist is very useful at ISP level. But it has several weaknesses. First, more than half of the spam mail servers are not in the blacklist. Second, the effect of blacklist depended on the administrator of the blacklist. If the black is wrong, it is possible that filtering legitimate mails [11].

Signature Based Filtering. The method of signature based filtering is comparing incoming mails with the spam prior received. In order to know whether two mails are the same, the filter calculates “signatures” for them. Signature based filter rarely blocks legitimate mails, but the weakness is that spammers can add random stuff to each copy of spam and give it a distinct signature, so that they can trick the signature based filters.

Rule Based Filtering. Rule-based filters try to discover the patterns, e.g. words or phrases, malformed headers and misleading dates. For example, RIPPER is based on keyword-spotting rules, which is a rule set generated by users’ manual setting. SpamAssassin, popularly used open source spam filter, uses a large set of heuristic rules. But the main disadvantage of rule-based filters is that they tend to have high false positive rates. For example, SpamAssassin has a problem with false positives rates [3] [17] [18].

Text Classification Filtering. A text classification filter uses text classification technique to filter spam. There have been several studies in this application, which include keyword-based, phrase-based, and character-based. Naïve Bayes-based [8; 18] method is also another efficient approach of keyword and phrase-based. It is a probabilistic classification by using features extracted from emails. Additionally, Boosting [21], Support Vector Machine [5], Rocchio [10], and decision tree based on ID3, C4.5, or C5 [19] algorithm, can be identified as the representative methods to analyze keywords in email.

Multi-agent Based Filtering. Due to the massive-distribute characteristic of spam, Multi-agent-based Filter, a new architecture, was proposed. The main feature is that clients can exchange knowledge about spam. Metzger et al. proposed an architecture that combines signature-based filter, SVM text classification, and multi-agent system [15].

3 Methodology

In this section, we will describe three statistical methods for individual learning. Each algorithm has different features. Then we will describe the preprocessing for the collaborative filtering and how our collaborative filtering method works.

3.1 Individual learning method

We employed three wildly used methods in the collaborative spam filtering process for individual learning. These algorithms include Naïve Bayes, Fisher's probability combination method, and Chi-square by degree of freedom. Fig. 1 shows a typical individual learning model for spam filtering.

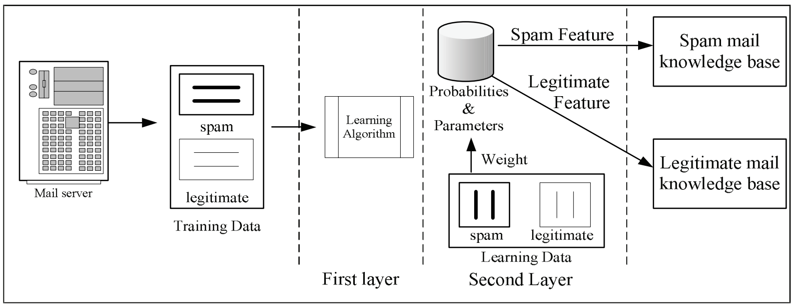


Fig.1. The individual learning model for spam filtering

Naïve Bayes Classifier. A Naïve Bayes classifier computes the likelihood that whether a mail is spam or not given the features that are contained in the mail [8]. The model, output by the Naïve Bayes algorithm, labels examples based on the features that they contain. We define *C* to be a random variable over the set of classes: legitimate and spam. That is, we want to compute *P(C/F)*, the probability that a mail is in a certain class given the mail contains the set of features F. We apply Bayes rule and express the probability as:

$$P(C/F) = \frac{P(F/C) * P(C)}{P(F)} \quad (1)$$

To use the Naïve Bayes rule we assume that the features occur independently from one another. If the features of a mail F include the features $F_1, F_2, F_3 \ldots F_n$, then equation (1) becomes:

$$P(C \mid F) = \frac{\prod_{i=j}^{n} P(F_i \mid C) * P(C)}{\prod_{j=1}^{n} P(F_j)} \quad (2)$$

Each *P(Fi /C)* is the frequency that features F_i occurs in a mail of class *C*. *P(C)* is the proportion of the class *C* in the entire set of mails. The output of the classifier is the highest probability class for a given set of strings. Since the denominator of equation (1) is the same for all classes we take the maximum class over all classes e of the probability of each class computed in equation (2) to get

$$\text{Most Likely Class} = \max_C \left(P(C) \prod_{i=1}^{n} P(F_i \mid C) \right) \tag{3}$$

Where, we use $\max_C$ to denote the function that returns the class with the highest probability. Most Likely Class is the class in *C* with the highest probability and hence the most likely classification of the example with features *F*. Then we applied equation (3) to compute the most likely class for the mail.

Fisher's Probability Combination Method. Robinson proposed a Bayes-like method that can release the independent assumption through Fisher's method to combine probabilities [20]. For each word that appears in the training data, we calculate:

$$b(w) = \frac{\text{number of spam containing the word } w}{\text{total number of spam}} \tag{4}$$

$$g(w) = \frac{\text{number of legistimate mail containing the word } w}{\text{total number of legistimate mail}} \tag{5}$$

$$p(w) = \frac{b(w)}{b(w) + g(w)} \tag{6}$$

p(w) can be interpreted as the probability that randomly chosen an email that containing word "*w*" will be spam. There is a problem with the probabilities calculated as above when some words are very rare in the training set. For instance, if a word appears in exactly one email and is a spam, the value of *p(w)* is 1.0.

The Fisher's probability combination approach lets us combine our general background information with the data we have collected for a word in such a way that both aspects are given their proper importance. In this way, we determine an appropriate degree of belief about whether, when we see the word again, it will be in a spam. We calculate this degree of belief, *f(w)*, as follows:

$$f(w) = \frac{(s \times x) + (n \times p(w))}{s + n} \tag{7}$$

s: the strength we want to give to our background information

x: our assumed probability, based on our general background information, that a word we don't have any other experience of will first appear in a spam

n: the number of emails we have received that contain word

In practice, the values for *s* and *x* are found through testing to optimize performance. Reasonable starting points are 1 for s and 0.5 for *x*.

In the proposed method, first, we should calculate $(\ 2)\ln(p_1\ \ p_2\ \ \dots\ \ p_n)$. Then, consider the result to have a Chi-square with 2n degrees of freedom, and use Chi-square Table to compute the probability. The "spammness" probability of a mail that contains specific w is:

$$H = C^{-1}[-2\ln \prod_w f(w), 2n] \tag{8}$$

where

H: the "spammness" probability of a mail

C^{-1}: the inverse Chi-square function, used to derive a p-value from a Chi-square distributed random variable

Chi-square by Degree of Freedom. O'Brein and Vogel suggested using an authorship identification technique known as Chi-square by degree of freedom method for spam filtering [16]. This idea is based on Pearson's Chi-square statistic. They argued that as over 90% world spam was the work of just 140 spammers [14], Methods should be devised to identify the "textual fingerprints" of these spammers [17]. Baayen et al. note that authors may have textual fingerprints on texts they produced [2]. At least writers who are not consciously charging their style of writing across texts will leave their fingerprints in the text. If this is the case, we could use authorship identification methods to identify these textual fingerprints and eliminate a large proportion of spam.

The Chi-square test is a non-parametric test of statistical significance. In order to carry out a Chi-square analysis, the sample must be randomly drawn from the population. Also the data must be frequencies as opposed to percentages. The measured variables must be independent and finally the frequencies must not less than 5 are disregarded [17]. The Chi-square statistic can be calculated by

$$\chi^2 = \sum \frac{(O_i - E_i)^2}{E_i} \quad (9)$$

where

χ^2 : The Chi-square value of an incoming mail

O_i : The observed value of an incoming mail

E_i : The expected value, calculated from the training set

3.2 Collaborative learning method

The proposed Collaborative Learning Model is based on three data mining algorithms. Each method contributed to the overall spam detection work through learning and collaboration. The whole collaborative filtering algorithms use the following steps to make recommendations to a user.

Construct posterior probabilities sets. We use training data to construct the posterior probabilities and parameters for each individual learning algorithm. According to the methods used, there are different posterior probabilities and parameters, in our case, spam and legitimate mail:

$$Pm_k c_x \mid C \leftarrow \{spam, legitimate\} \quad (10)$$

Where C are the set of classes and m are the set of learning algorithm.

Normalize the inconsistent scales. In order to collaborate to the output generated from the method we discussed previously, it is important to normalize the ratings of different scales to the same. We use the Gaussian Normalization Method for the collaborative work [9].

$$\hat{R}(x) = \frac{R_y(x) - \overline{R}_y}{\sqrt{\sum_x (R_y(x) - \overline{R}_y)^2}} \quad (11)$$

Where $\hat{R}(x)$ is the normalized value, $R_y(x)$ is the output value of each method, and $\overline{R}_y$ is the average output value of each method, derived from the weighting set.

Determining Weighting Value. Using learning data to determine weight value based on the result of individual learning. In each method, if the individual classifier makes right judgment to spam. It will be rewarded; otherwise, it will be punished. Because of the cost among the right and wrong decisions bring different cost to user. We give the reward value twice than the punishment. The learning weight generated from the subtraction of reward value and punishments, as shown in equation (12).

$$W_{mk} = (2 * va) - vp \qquad (12)$$

Combine weights and probabilities to take shape collaborative learning model. The collaborative learning model is constructed by combining the posterior probabilities sets and weight values from individual learning. Weight each probability of individual learning algorithm and use collaborative learning model to calculate the predicted rating in testing data. Individual learning process and use collaborative learning model to derive the predicted rating to incoming mail.

$$Max(Pc_x = \sum_{g=1}^{k} Pm_g c_x * W_{mg} \mid C \leftarrow \{spam, legitimate\}) \qquad (13)$$

The predicted rating $Pm_g c_x$ and the learning weight W_{mg} will depend on the collaborative learning algorithm. If the predicted rating of one class is higher than other classes, the system will recommend this class to the result.

We designed a simple collaborative learning algorithm as illustrated in Fig. 2, and the notations of the algorithms are shown in Table 1:

Table 1. Notations of the collaborative learning algorithm

T // Training data, L // Learning data, M // Individual learning algorithm
W // Weight, A // Attributes, C // Classes
va // Reward weight values, vp // Punishment weight values
$Pm_k c_x$ // the probability output by the individual learning process
W_{mk} // the weight derive from the learning data with the individual learning process
Pc_x // the summation of $Pm_k c_x \times W_{mk}$

Algorithm：A classifiable algorithm based on collaborative and learning
Input：T, L, M
Output：Collaborative learning model
Method：
$C \leftarrow \{c_1, c_2 ... c_x\}$, $T \leftarrow \{t_1, t_2 .. t_i\}$, $A \leftarrow \{a_1, a_2 ... a_j\}$, $M \leftarrow \{m_1, m_2 ..... m_k\}$, $L \leftarrow \{l_1, l_2 .. l_n\}$
//Create posterior probability for every method
    For each $m_k \in M$ do
        Gain $Pm_k c_x$
//Computing probability of each attributes in C
    For each $a_j \in A$ do
        For each $t_i \in T$ do
            IF $t_i \in c_x$ then
                $s = s + 1$

$P(C_x) = s / c_x$
End if
Next t
Next a
Next m
// Learning weight
For each $m_k \in M$ do
Gain W_{mk}
For $l_n \in L$ do
//Gain the filter result for each individual algorithm to compare probability (va , vp)
$$Max = (Pm_k c_x \mid C \leftarrow \{c_1, c_2 ... c_x\})$$
IF the result of individual algorithm is same as the result of the class C_x
$$W_{mk} = (2 * va) + vp$$
Else IF result of individual algorithm is opposite as the result of the class C_x
$$W_{mk} = (2 * va) - vp$$
End if
Next w
Next m
// Collaborative model
$$Max(Pc_x = \sum_{g=1}^{k} Pm_g c_x * W_{mg} \mid C \leftarrow \{c_1, c_2 ... c_x\})$$

Fig.2. The proposed collaborative learning algorithm

4 Experiment and Results

In this section, we will use the Spam Email Database from the UCI Machine Learning Repository to examine our proposed collaborative method.

4.1 Data set

The Spam Email Database was created by Hewlett-Packard Labs [22]. It had been used for the HP internal-only technical report and other spam detection studies. The database contains 4601 instances and 58 attributes. We estimate our results over new data by using cross validation. Cross validation is the standard method to estimate the likely predictions over unseen data in measuring the result of data mining or machine learning Mining [13]. We randomly choose 50% instances for the algorithm training, 25% for the weight generating, and the remaining partition is then used to evaluate our method. Then we repeated the process leaving out a different partition for testing each time. This gave us a very reliable measure about our method's accuracy over unseen data. The data set and its usage in our study is summarized in Fig. 3:

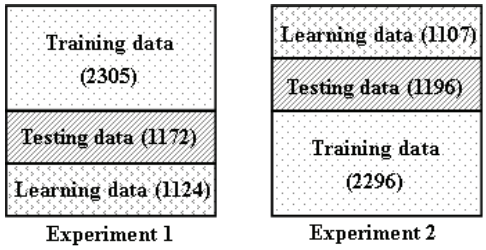


Fig.3. Cross validation of experiment 1 and 2

4.2 Results and Analysis

In order to evaluate the performance of three machine learning methods and collaborative learning that we proposed in spam filtering. We were interested in several quantities typically used in measuring the query result of information retrieval. These are:
True Positives (TP). The number of spam mail classified as spam.
True Negatives (TN). The number of legitimate mail classified as legitimate.
False Positives (FP). The number of legitimate mails falsely classified as spam.
False Negatives (FN). The number of spam mails falsely classified as legitimate.

The Detection Rate is defined as TP / (TP +FN), False Positive Rate as FP / (TN +FP), and Overall Accuracy as (TP +FN) / (TP +FP + FN + TN). The results of all individual learning methods and collaborative learning method are presented in Table 2 and Table 3. The voting scheme of three individual learning methods is also included for comparison.

Table 2. Results of the experiment (1)

	TP	TN	FP	FN	Detection Rate	False Positive Rate	Overall Accuracy
Naïve Bayes	365	679	52	76	82.77%	7.11%	75.43%
Fisher's	426	509	222	15	96.60%	30.37%	67.56%
Chi-square	404	628	103	37	91.61%	14.09%	74.57%
Voting	416	613	118	25	94.33%	16.14%	74.35%
Collaborative	422	646	85	19	95.69%	11.63%	77.17%

Table 3. Results of the experiment (2)

	TP	TN	FP	FN	Detection Rate	False Positive Rate	Overall Accuracy
Naïve Bayes	472	594	28	104	81.94%	4.50%	77.02%
Fisher's	568	457	165	8	98.61%	26.53%	74.06%
Chi-square	544	558	64	32	94.44%	10.29%	79.62%
Voting	557	543	79	19	96.70%	12.70%	79.48%
Collaborative	567	579	43	9	98.44%	6.91%	82.80%

As can be seen from Table 2 and 3, we can find that Fisher's method has the highest precision rate, but the recall and accuracy rates are not as good as others. And it suffers from the false positives rate. The Naïve Bayes has the lowest false positives rate and good in accuracy rate, though the detection rate is the lowest of the three. The Chi-square method generally has better performance than others. From the results of the experiment, the detection and false positive rate of collaborative learning scheme and voting scheme are down little less than those individual learning algorithms. But the accuracy rate of the collaborative learning is the best. Generally, collaborative learning has better performance than those individual learning algorithms or voting scheme.

5 Conclusion and Future Work

In order to deal with the huge amount of spam received day by day, powerful email filters with high reliability are needed. One problem of traditional text classification and signature-based methods is that they are based on a single method. In this paper, we introduced a collaborative learning scheme that can parallel filter spam with three different methods. In the proposed scheme, emails that are difficult to classify with a single one method, can be detected and filtered through the collaborative filtering architecture based on the collaborative weighting and learning. Thus, it has several advantages: first, the filtering system will not be affected by the failure of one method; second, the system can be deployed to P2P networks easily and provide higher reliability than centralized network; third, the training process is enhanced by the weight learning process, as we showed previously, the weight learning process provides a group learning mechanism for the three different filtering method and brings more accuracy result.

One of the most important areas of future work for this application is the development of more efficient algorithm. The current probabilistic method requires significant computing resources. Many incremental learning algorithms may solve this problem [6]. Another one is adding signature based function to our system. Although signature based methods have several weakness, but under the collaborative scheme, it helps building clusters for users' preferences.

Acknowledgement: The author would like to give thanks to the National Science Council of Taiwan for grant NSC 93-2213-E-224-038- to part of this research.

References

1. Androutsopoulos, I., Koutsias, J., Chandrinos, K. V., and Spyropoulos, C. D.: An Experimental Comparison of Naïve Bayesian and Keyword-Based Anti-Spam Filtering with Personal E-mail Messages. In Proc. of the 23 rd Annual International ACM SIGR Conference on Research and Development in Information Retrieval, Athens, Greece, (2000) 160–167

2. Baayen, H., Van Halteren, H., Neijt, A., and Tweedie, F.: An experiment in authorship attribution. In Journ´ees internationales d'Analyse statistique des Donn`ees Textuelles, (2002)
3. Cohen, William W.: Learning Rules that classify e-mail, the AAAI Spring Symposium on Machine Learning in Information Access. (1996) 18–25
4. Data Mining Cup 2003. Contest data, instructions and results (2003) available from: http://www.data-mining-cup.com/2003/Wettbewerb/1059704704/
5. Drucker, H., Wu, D., and Vapnik, V., N.: Support Vector Machines for Spam Categorization. IEEE Transaction on Neural Networks, Vol 10, No 5, (1999)
6. Giraud-Carrier, C.: Unifying Learning with Evolution through Baldwin an Evolution and Lamarckism: A Case Study. In: Proceedings of the Symposium on Computational Intelligence and Learning (CoIL-2000), MIT GmbH, June (2000) 36–41
7. Gomez Hidalgo, J. M.: Evaluating Cost-sensitive Unsolicited Bulk Email Categorization. In Proceedings of SAC-02, 17th ACM Symposium on Applied Computing, Madrid, ES, (2002) 615–620
8. Han, J. and Kamber, M.: Data mining concepts and techniques (USA, Morgan Kaufmann, 2001) page 284–287
9. Jin, R. and Si, L.: A study of methods for normalizing user ratings in collaborative filtering. Proceedings of the 27th annual international conference on Research and development in information retrieval, July (2004) 568–569
10. Joachims, T.: A probabilistic analysis of the Ricchio algorithm with TFIDF for text categorization. In Proc. of 14th International Conference on Machine Learning, (1997)
11. Jung, Jaeyeon, and Sit, Emil: An empirical study of spam traffic and the use of DNS black lists. Proceedings of the 4th ACM SIGCOMM conference on Internet measurement, October (2004) 370–375
12. Katirai, H.: Filtering junk e-mail: A performance comparison between genetic programming & naive bayes. (1999) available from: http://members.rogers.com/hoomank/papers/katirai99filtering.pdf
13. Kohavi, R.: A study of cross-validation and bootstrap for accuracy estimation and model selection. IJCAI, (1995)
14. Ludlow, M., Just 150 'spammers'blamed for e-mail woe. The Sunday Times, 1st December (2002) page 3
15. Metzger, J., Schillo, M. and Fischer, K.: A multiagent-based peer-to-peer network in java for distributed spam filtering. In Proc. of the CEEMAS, Czech Republic, June (2003)
16. O'Brien, C. and Vogel, C., Spam filters: Bayes vs. chi-squared; letters vs. words. In Proceedings of the International Symposium on Information and Communication Technologies, (2003)
17. O'Brien, C. and Vogel, C.: Comparing SpamAssassin with CBDF Email Filtering. In Proceedings of the 7th Annual CLUK Research Colloquium, (2004)
18. Provost., J.: Naive-bayes vs. rule-learning in classification of email. Technical Report AI-TR-99-284, University of Texas at Austin, Artificial Intelligence Lab, (1999)
19. Quinlan., J. R.: C4.5: Programs for Machine Learning. Morgan Kaufmann series in machine learning. Morgan Kaufmann, (1993)
20. Robinson, G.: A Statistical Approach to the Spam Problem. Linux Journal, March 2003. Linux Journal, Volume 2003, Issue 107, March (2003) Page 3
21. Schapire, R.E., and Singer, Y.: BoosTexter: A Boosting-based System for Text Categorization, Machine Learning. vol 39, (2000) 135–168
22. UCI Machine Learning Repository. Retrieved Jan 20, (2005) available from: http://www.ics.uci.edu/~mlearn/MLRepository.html

Analyzing the Impact of Corpus Preprocessing on Anti-Spam Filtering Software

J. R. Méndez[1], E. L. Iglesias[1], F. Fdez-Riverola[1], F. Díaz[2], 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. Because of the volume of spam e-mail and its evolving nature, many statistical techniques have been applied until now for the construction of anti-spam filtering software. In order to train and test filters, it is necessary to have a large e-mail corpus. In this paper we discuss several considerations that researchers must take into account when building and processing a corpus. After reviewing several text preprocessing methods used on spam filtering, we show the results obtained by different machine and lazy learning approaches when the preprocessing of the training corpus changes. The results obtained from the experiments carried out are very informative and they back up the idea that instance-based reasoning systems can offer significant advantages in the spam filtering domain.

1 Introduction and Motivation

Unsolicited e-mail messages, also known as spam, have become a serious problem for internet users. Recent studies show that between one-seventh and one-half of e-mail messages going into internet user inboxes are spam [1].

Nowadays anti-spam filtering software seems to be the most viable solution to the spam problem. It works attempting to automatically identify an incoming e-mail message using different approaches [2] and classifying it as either 'spam' or 'legitimate'. Spam filtering methods are often classified as rule-based or content-based (statistical). The first ones classify documents based on whether or not they meet a particular set of criteria [3]. The last ones do not require specifying any rules explicitly. They are primary driven by statistics that can be derived from the content of the messages (i.e., word frequency) [4].

In our work we identify two main types of content-based techniques: (*i*) machine learning (ML) algorithms and (*ii*) memory-based and case-based reasoning approaches. ML approaches use an algorithm to 'learn' the classification from a set of

© A. Gelbukh, R. Monroy. (Eds.)
Advances in Artificial Intelligence Theory
Research on Computing Science 16, 2005, pp. 129-138

training messages. On the other hand, memory-based and case-based reasoning techniques store all training instances in a memory structure and try to classify new messages finding similar e-mails on it. Hence, the decision of how to solve a problem is deferred until the last moment.

Content-based filters tend to be more successful than rules-based ones. It is more difficult for spammers to create a spam e-mail because they have no idea what type of e-mail each user is training their filters on. Furthermore, the lazy nature of memory-based and case-based approaches makes them specially suitable for dynamic environments as spam [5, 6].

In order to train and test content-based filters, it is necessary to build a large corpus with spam and legitimate e-mails or use a public corpus. Anyway, e-mails have to be preprocessed to extract their words (*features*). Also, since the number of features in a corpus can end up being very high, it will generally be necessary to choose those features that better represent e-mails before carrying out the filter training to prevent the classifiers from overfitting.

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 step of e-mail features extraction and the later selection of the most representative are crucial for the performance of the filter.

Our main goal in this paper is the evaluation and comparison of different feature extraction techniques used in text categorization. We will analyze what are their strengths and weaknesses when they are applied to the spam problem domain. Therefore, we will show the results obtained by different well-known content-based techniques when the preprocessing of the training corpus changes. The selected models go from the utilization of Naïve Bayes [7], boosting trees [8], Support Vector Machines [9] to three case-based systems for spam filtering that can learn dynamically: a Cunningham *et al.* system which we call *Cunn Odds Rate* [10], its improved version named ECUE [11] and the SpamHunting system [12].

The rest of the paper is organized as follows: Section 2 introduces previous work on machine learning and case-based e-mail filters. Section 3 describes the selected public available corpus for empirical model evaluation. Section 4 discuses several issues related with message representation and feature selection. Section 5 presents the experiments carried out and the results obtained and discusses the major findings. Finally, Section 6 exposes the main conclusions reached by the analysis of the experiments carried out.

2 Spam Filtering Techniques

2.1 Machine Learning Approaches

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 to classify spam. 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 Naïve Bayes algorithm [7]. Besides bayesian models, Support Vector Machines (SVM) and boosting techniques are also well-known ML techniques used in this field.

SVMs [9] 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. SVM has become very popular in the ML and DM community due to its good generalization performance and its ability to handle high-dimensional data through the use of kernels. 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 [13] for details).

Boosting algorithms [8] are techniques based on the use of weak learners; that is to say, algorithms that learn with a next error rate to 50%. 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. Different boosting algorithms have been developed for classification tasks, so much binary as multi-class. Among them we could highlight Adaboost [14].

Recently, several new ML models has been introduced for e-mail classification such as Chung-Kwei [15], which is based on pattern-discovery. As well as it is faster than other ML approaches, it becomes better on performance.

2.2 Case-based Reasoning Approaches

Case-based approaches outperform previous techniques in anti-spam filtering [11]. Case-based classification works well for disjoint concepts as spam (spam about *porn* has little in common with spam offering *rolex*) whereas ML techniques try to learn a unified concept description. Another important advantage of this approach is the ease with which it can be updated to tackle the *concept drift* problem in the anti-spam domain [6].

Delany *et al*. present in [11] a case-based system for anti-spam filtering called ECUE (*E-mail Classification Using Examples*) that can learn dynamically. The system use a similarity retrieval algorithm based on Case Retrieval Net*s* (CRN) [16]. CRN networks are equivalent to the k-nearest neighbourhood algorithm but are computationally more efficient in domains where there is feature-value redundancy and missing features in cases, as spam. ECUE classifier use unanimous voting to determine whether a new e-mail is spam or not. All the returned neighbours need to be classified as spam e-mails in order to classify as spam the new message. The ECUE

[1] WEKA is available from http://www.cs.waikato.ac.nz/ml/weka/
[2] YALE is available from http://yale.sourceforge.net

system represents the evolution from *Cunn Odds Rate* [10], a previous successful system of the same authors.

Also, in [12] a lazy learning hybrid system is introduced to accurately solve the problem of spam labelling and filtering. The model, named SpamHunting, 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 e-mail. 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 means of 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 Benchmark Corpus for Spam Filtering Research

As previously mentioned, it is essential to provide content-based filters with an appropriate corpus of e-mails for training and testing purposes. The corpus should be made up of both spam and legitimate e-mails. Each message should be marked as being either spam or non-spam. By training the filters on this corpus, they should learn the main characteristics that differentiate spam from legitimate messages.

Table 1. Temporal distribution of messages belonging to the SpamAssassin corpora of emails

	Jan 02	Feb 02	Mar 02	Apr 02	May 02	Jun 02	Jul 02	Aug 02	Sep 02	Oct 02	Nov 02	Dec 02	Sum 02
Spam	2	1	0	0	0	1	8	182	276	5	0	0	475
Error	-	-	-	-	-	-	-	-	-	-	-	-	26
Legitim.	0	44	0	0	2	5	157	561	1272	757	0	0	2798
Total													3299
	Jan 03	Feb 03	Mar 03	Apr 03	May 03	Jun 03	Jul 03	Aug 03	Sep 03	Oct 03	Nov 03	Dec 03	Sum 03
Spam	0	12	18	18	312	145	496	330	274	6	9	25	1645
Error	-	-	-	-	-	-	-	-	-	-	-	-	239
Legitim.	1	44	0	0	2	7	704	1334	1239	727	36	55	4149
Total													6033

Despite privacy issues related with the content of a message, there are several public available corpora of e-mails just as LingSpam[3], PU[3], JunkeE-mail[4], DivMod[5] or

[3] Available at http://www.iit.demokritos.gr/
[4] Available at http://clg.wlv.ac.uk/projects/junk-e-mail/

SpamAssassin[6]. In our work, we use the SpamAssassin corpora. It contains 9332 different messages from January 2002 up to and including December 2003 distributed as Table 1 shows. The error row shows the presence of messages with a corrupt date.

4 Message Representation

In order to increase message manipulation speed as well as knowledge representation ability, messages should not be stored in its primitive form, as they are in a original corpus. It is necessary to convert each message into a reliable message descriptor which can be easily assembled. In learning algorithms, training messages are usually represented as a vector $\vec{t} = \langle t_1, t_2, ..., t_p \rangle$ of weighted terms, T_i, much as in the vector space model in information retrieval [17, 18].

Features can be identified using a variety of generic lexical tools, primarily by tokenising the e-mail into words. At first glance, all that seems to be involved in it is the recognition of spaces as word separators. However, at least the following particular cases have to be considered with care: hyphens, punctuation marks, and the case of the letters (lower and upper case).

Lexical analyzer normally breaks hyphenated words and remove punctuation marks. However, in the spam domain many of these symbols are among the best discriminating attributes in a corpora, because they are more common in spam messages than legitimate ones. For this reason, hyphens and punctuation marks are not removed here. On the subject of case, the lexical analyzer normally converts all the text to either lower or upper case. It is also done here to reduce the number of terms.

In text categorization it is common to reduce the set of representative terms with very large collections [18]. This can be accomplished through the elimination of *stopword* (such as articles and connectives) and the use of *stemming* (which reduces distinct words to their common grammatical root). Since spam is a special form of text categorization, it could be applied here also.

Once carried out the lexical analysis over the corpus, the weight of terms in each message e, 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 calculated by Expression (1) or (*iii*) the *inverse document frequency* (IDF) given by Expression (2) denoting those terms that are common across the messages of the training collection.

$$t_i(e) = \frac{n_i(e)}{N(e)} \tag{1}$$

$$t_i(e) = \frac{n_i(e)}{N(e)} \log_2 \frac{m}{df(T_i)} \tag{2}$$

---

[5] Available at http://www.divmod.org/cvs/corpus/spam/

[6] Available at http://www.spamassassin.org/publiccorpus/

In Equations (1) and (2), $n_i(e)$ is the number of occurrences of term T_i in e, $N(e)$ represents the total number of occurrences of terms in e, m is the number of training messages and $df(T_i)$ stands for the number of training messages where the term T_i occurs.

5 Performance Evaluation

The final goal of our experiments is to measure the impact of applying different preprocessing steps over the corpus before training the models. The experiments have been done using Naïve Bayes, a SMO implementation of SVM, Adaboost, and the three previously commented CBR systems: ECUE, *Cunn Odds Rate* and SpamHunting.

Six well-known metrics [4] have been used in order to evaluate the performance (efficacy) of all the analyzed models: percentage of correct classifications (%OK), percentage of False Positives (%FP), percentage of False Negatives (%FN), spam *recall*, spam *precision* and *total cost ratio* (TCR) with three different cost values. The experiments were carried out at three different preprocessing scenarios: (*i*) applying stopword removal and stemming analysis, (*ii*) applying stopword but without stemming and (*iii*) without applying neither stopword nor stemming.

Typically, message representation scheme presented in Section 4 leads to very high-dimensional feature spaces. Several authors have noted the need for feature selection in order to make possible the use of conventional ML techniques, to improve generalization accuracy and to avoid over fitting of the models. Following the recommendation of [19], the *information gain* (IG) criterion is usually used to select a representative subset of features.

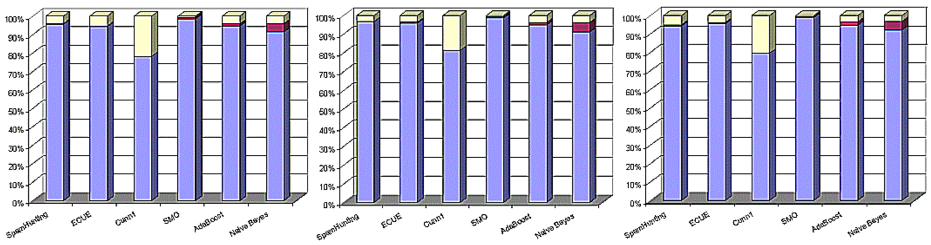
All the analyzed models except from *Cunn Odds Rate* and SpamHunting systems use IG to select the most predictive features as it has been shown to be an effective technique in aggressive feature removal in text classification [19]. For our comparisons, we have selected the best performance model of each technique varying between 100 and 2000 features. For *Cunn Odds Rate* model, we have maintained the original technique of selecting 30 words for representing spam e-mails plus 30 words representing legitimate messages. The algorithm employed for sorting the vocabulary is based on the odds-ratio described in [11].

SpamHunting terms selection is not made using the vocabulary of the whole corpus. Instead of this, each message has its own relevant terms. The relevant feature list of each message is computed as the minimum set containing the most frequent features of the specified e-mail, which frequency amount is greater than a specified threshold in the range [0,1]. As the best results have been obtained using the 30% frequency amount, we computed the relevant feature list as the most repeated features whose frequency amount is greater than mentioned threshold.

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.

5.1 Experimental Results

Figure 1 shows the percentage of correct classifications, false positive rate and false negative rate belonging to the six analyzed models over the defined scenarios. Analyzing Figure 1 we can realize that Naïve Bayes and SVM techniques get better performance with no stemming and no stopword removal (Scenario 3). When stopword and stemming are used (Scenario 1) SpamHunting, ECUE and Adaboost report better accuracy but generally worst results. However, applying only stopword removal (Scenario 2) in those models that do not incorporate IG for feature selection leads to a significant accuracy increment. In order to facilitate a deeper analysis, Table 2 shows the mean values over the 10 fold-cross validation for the scores presented in Figure 1.



(a) Scenario 1 (b) Scenario 2 (c) Scenario 3

Fig. 1. Comparative model performance varying preprocessing steps

Table 2. Mean value of correct clasifications, FPs and FNs with 10 fold-cross validation

		Naïve Bayes	**Ada boost**	**SVM**	**Cunn Odds Rate**	**ECUE**	**Spam Hunting**
Scenario 1	OK	852.5	881.9	920.2	730.4	880.2	892.9
	False Positives	43	12.2	7.8	0.5	3.6	1.5
	False Negatives	37.7	39.1	5.2	202.3	49.4	38.8
Scenario 2	OK	849.6	885.1	919.1	758.3	893	900.8
	False Positives	48.6	13.4	8.7	0.2	6.1	1.8
	False Negatives	35	34.7	5.4	174.7	34.1	30.6
Scenario 3	OK	857.9	883.4	922.2	742.1	889.8	880.3
	False Positives	44	16	5.3	0.1	6.8	4.2
	False Negatives	31.3	33.8	5.7	191	36.6	48.7

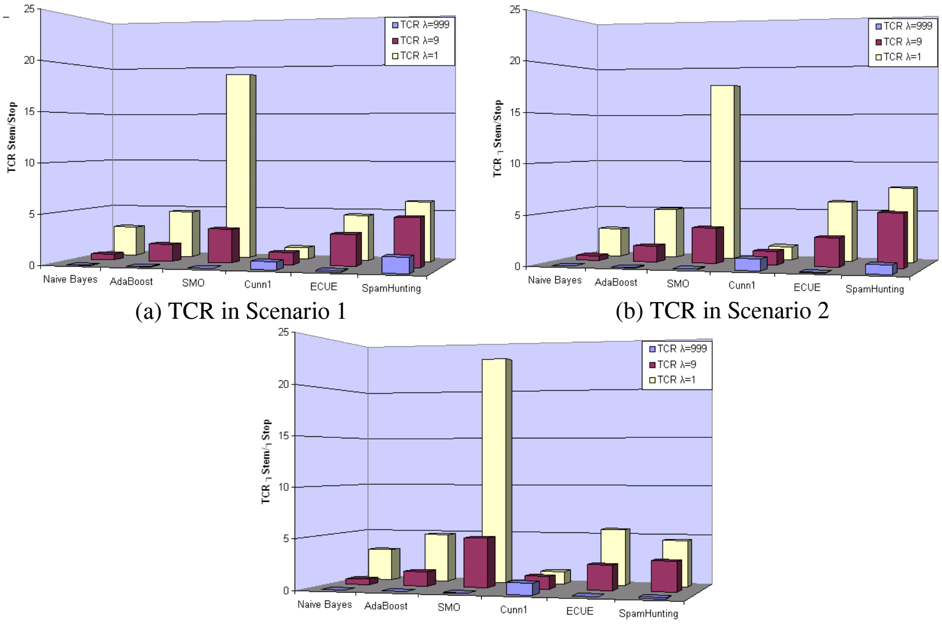
Table 3 shows a comparative study between the three proposed scenarios using recall and precision scores. Results on recall evaluation using classical ML models are better when no stopword removal and no stemming is used (Scenario 3). However, CBR/IBR approaches become better when only stopword removal is performed (Scenario 2) while the worst results are obtained when both stemming and stopword removal is applied (Scenario 1).

Table 3. Averaged precision and recall scores over 10 fold-cross validation

		Naïve Bayes	**Ada boost**	**SVM**	**Cunn Odds Rate**	**ECUE**	**Spam Hunting**
Scenario 1	Recall	0.842	0.836	0.978	0.150	0.793	0.837
	Precision	0.824	0.943	0.968	0.986	0.981	0.993
Scenario 2	Recall	0.853	0.854	0.977	0.266	0.857	0.871
	Precision	0.807	0.939	0.964	0.997	0.971	0.991
Scenario 3	Recall	0.869	0.858	0.976	0.198	0.846	0.795
	Precision	0.824	0.928	0.978	0.998	0.967	0.978

Analyzing in Table 3 precision scores gathered from experiments, we can realize that it is possible to obtain better results when stopword removal and stemming is applied (Scenario 1) except for SVM and *Cunn Odds Rate* models. These techniques work better without any preprocessing step (Scenario 3).

In order to compare the performance of the models taking into account the three predefined scenarios but with a cost-sensitive point of view, we calculate the TCR score in three different situations. TCR assumes than FP errors are λ times more costly than FN errors, where λ depends on the usage scenario (see [4] for more details). In the experiments carried out in this paper, the values for λ parameter were 1, 9 and 999.



(a) TCR in Scenario 1

(b) TCR in Scenario 2

(c) TCR in Scenario 3

Fig. 2. TCR score graphics varying λ (equal to 1, 9, 999) value in the three different scenarios

Figure 2 shows the results taking into account the TCR score and varying the λ parameter. The best performance on classical ML approaches are obtained when no stemming and no stopword removal are applied (Scenario 3) except for Adaboost, that increases its TCR score when only stopword are applied (Scenario 2). By other side, CBR/IBR approaches work better if stopword are applied to the corpus (Scenarios 1 and 2).

6 Conclusions

In this paper we have analyzed the impact of corpus preprocessing in the performance of anti-spam filtering software. For this task, we have briefly revised the most popular filtering techniques from the ML community as well as CBR and IBR previous successful implemented systems. Before defining the experiments to carry out, we have presented the benchmark corpora of e-mails and discussed several issues about message representation.

In order to carry out the experiments, we have considered three different scenarios and six standard scores to measure performance among the models. 10-fold stratified cross-validation was used in order to increase the confidence level of results obtained. From the analysis of these results we can infer valuable information about the preprocessing needed in order to construct accurate anti-spam filtering models.

Firstly, classical ML based models can get the best number of correctly classified messages by removing all preprocessing steps, but stopword removal and stemming is recommended if best accuracy is needed. Secondly, CBR/IBR models can obtain better performance by stopword removal although stemming can improve accuracy.

Moreover, applying stemming can significantly reduce the number of selected features belonging to the corpus. According to this, it will also decrease the time needed to compute IG for all features and the spam recall score. Nevertheless, as results show, if stemming is applied the models will obtain the smallest amount of correct classifications excluding the SpamHunting system.

If the main goal is the minimization of the FP errors among the models, the results obtained from experiments suggest that stemming should be used. This idea is backed up because legitimate messages are better classified when stemming is used (by the successful identification of semantic roots belonging to legitimate messages). This fact helps the models to better differentiate spam from legitimate e-mails. In addition, if the main goal is the improvement in correct classification rate (therefore diminishing the total number of errors), then stemming is not recommended.

It is important to highlight that depending on the model, different results are obtained when changing the preprocessing steps carried out over the whole corpus. So, these issues need to be kept in mind in both training the model comparing its accuracy with another anti-spam classifiers.

The main conclusion of this work is that the effort on stemming and removing stopwords does not pay in improvement of the current algorithms for spam detection. In addition, spam producers are very creative, and learning from a static corpus (preprocessed or not) seems to be a now a naive approach.

References

1. Vaknin, S.: The Economics of Spam. United Press International (2002)
2. Michelakis, E., Androutsopoulos, I., Paliouras, G., Sakkis, G., Stamatopoulos, P.: Filtron: A Learning-Based Anti-Spam Filter. Proc. of the First Conference on E-mail and Anti-Spam CEAS, (2004)
3. Wittel, G.L., Wu, S.F.: On Attacking Statistical Spam Filters. Proc. of the First Conference on E-mail and Anti-Spam CEAS, (2004)
4. Androutsopoulos, I., Paliouras, G., Michelakis, E.: Learning to Filter Unsolicited Commercial E-Mail. Technical Report 2004/2, NCSR "Demokritos" (2004)
5. Kelly, M.G., Hand, D.J., Adams, N.M.: The impact of changing populations on classifier performance. Proc. of the 5th International Conference on Knowledge Discovery and Data Mining, ACM Press, (1999) 367–371
6. Widmer, G., Kubat, M.: Learning in the presence of concept drift and hidden contexts. Machine Learning, Vol. 23 (1). (1996) 69–101
7. 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, (1998) 55–62, Technical Report WS-98-05 AAAI
8. 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
9. Vapnik, V.: The Nature of Statistical Learning Theory. 2nd Ed. Statistics for Engineering and Information Science, Springer, New York (1999)
10. 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
11. 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)
12. Fdez-Riverola, F., Méndez, J.R, Iglesias, E.L., Díaz, F.: Representación flexible de e-mails para la construcción de filtros anti-spam: un caso práctico. Proc. of the VI Jornadas de Transferencia Tecnológica de Inteligencia Artificial. *To appear*.
13. 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
14. Schapire, R.E., Singer, Y.: BoosTexter: a boosting-based system for text categorization. Machine Learning, Vol. 39 (2/3). (2000) 135–168
15. Isidore Rigoutsos and Tien Huynh. Chung-Kwei: a Pattern-discovery-based System for the Automatic Identification of Unsolicited E-mail Messages (SPAM). Proc. of the First Conference on E-mail and Anti-Spam CEAS, (2004)
16. Lenz, M., Auriol, E., Manago, M.: Diagnosis and Decision Support. Case-Based Reasoning Technology. Lecture Notes in Artificial Intelligence, Vol. 1400, (1998) 51–90
17. Salton, G., McGill, M.: Introduction to modern information retrieval, McGraw-Hill, (1983)
18. Baeza-Yates, R., Ribeiro-Neto, B.: Modern Information Retrieval. Addison Wesley, (1999)
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

Misuse Detection of Email Viruses base on SOM with k-medoids

Dong-Her Shih, Sheng-Fei Hsu, Hsiu-Sen Chiang, and Chun-Pin Chang

National Yunlin University of Science and Technology, Department of Information Management, 123, Section 3, University Road, Touliu, Yunlin, Taiwan
{shihdh,g9320803,g9023728,g9123728}@yuntech.edu.tw

Abstract. Email virus is an email that can infect other programs by modifying them to include a replication of it. When the infected emails are opened, the email virus spreads itself to others. We propose a novel approach to detect misuse emails by gathering and maintaining knowledge of the behavior of the malicious emails rather than anticipating attacks by unknown assailants. Any new activity of the email is compared to the malicious profile to detect a potential misuse. Comparison results show that our proposed methods outperformed than anti-virus software.

1 Introduction

In recent years, the number of Internet users worldwide has continued to rise dramatically as the Internet expands. Within this growth, serious problems such as unauthorized intrusions, denial of service attacks, and computer viruses have arisen. In particular, a computer virus (hereafter, virus) is able to cause damage to a large number of systems because of its ability to propagate. As a result, the power of such attacks can now have a serious impact on an information society. Analysis of reported virus incidents during the five-year period [1] provides interesting insights for anti-virus research, as it reflects a period of rapid uptake in the application of the Internet and the use of e-mail for business purposes.

Recently, a kind of virus that can infect executable files has appeared. This kind of virus spreads far more rapidly than before and can propagates by email, one means of information exchange among users. As shown in [2], the email vector again took an upward turn. Virus disasters and incidents have organizational ramifications beyond the money, resources, and effort required to recover from such incidents.

Our focus in this paper is primarily on detection of the misuse of malicious emails. Misuse detection systems offer a cost-effective compromise to establishing and assuring a certain degree of security in a system [3]. Our research presents a framework, an email virus filter that can detect malicious Windows attachments by integrating with an email server. Analyzing the characteristic of the email virus, to picked out differentiate one email virus from another. Our goal is to design and build a scanner that accurately detects email virus before they have been entered into a host.

© A. Gelbukh, R. Monroy. (Eds.)
Advances in Artificial Intelligence Theory
Research on Computing Science 16, 2005, pp. 139-148

The methods discussed in the paper are acting as a network email filter to catch malicious email virus before users receive them through their email.

2 Related Work

Detecting malicious executables is not a new problem in security. Protection from unknown viruses is, indisputably, the issue of the day in computer virology. Early methods used signatures to detect malicious programs. Current approaches are matching them to a set of known malicious programs. Experts were typically employed to analyze suspicious programs by hand. In an attempt to solve this problem, the anti-virus industry generates heuristic classifiers by hand [4]. Using their expertise, signatures were found that made a malicious executable example different from other malicious executables or benign programs. One example of this type of analysis was performed by Spafford [5]. He analyzed the Internet Worm and provided detailed notes on its spread over the Internet, the unique signatures in the worm's code, the method of the worm's attack, and a comprehensive description of system failure points. Malicious code is usually classified [6] into the following categories: Viruses, Worms, Trojan horses, Back doors, Spyware, Attack scripts, Java attack applets and Dangerous ActiveX. Combining two or more of these malicious code categories can lead to powerful attack tools. For example, a worm can contain a payload that installs a back door to allow remote access. When the worm replicates to a new system (via email or other means), the back door is installed on that system, thus providing an attacker with a quick and easy way to gain access to a large set of hosts. Once the back-door tool gains a large installed base, the attacker can use the compromised hosts to launch a coordinated attack, such as a distributed denial-of-service (DDoS) attack [7]. At IBM, Kephart and Arnold [8] developed a statistical method for automatically extracting malicious executable signatures. Their research was based on speech recognition algorithms and was shown to perform almost as good as a human expert at detecting known malicious executables. Their algorithm was eventually packaged with IBM's anti-virus software. Lo *et al.* [9] presented a method for filtering malicious code based on "tell-tale signs" for detecting malicious code. Similarly, filters for detecting properties of malicious executables have been proposed for UNIX systems [10] as well as semiautomatic methods for detecting malicious code [11]. As is well known, the main deficiency of commonly used anti-virus scanners is that these scanners are able to detect and delete only those malicious programs described in their anti-virus databases. The anti-virus community relies heavily on known byte-code signatures to detect malicious programs. The main disadvantage to this approach is that the scanner cannot detect a virus if the database doesn't contain its description and the user is unprotected from this new virus threatening during this time period.

Previous work done on detection can be divided into anomaly detection and misuse detection. Anomaly detection deals mainly with attack to the system from abuse. Misuse detection deals mainly with the attack to the system by an authorized user who is misusing his/her privileges. Misuse detection has generally been employed to complement the shortcomings of other prevention techniques [3]. Prior work on misuse detection has been mainly focused on using logs and user profiles. Profile-based detection systems audit the deviation of user activities from normal user profiles. In the past, a user's command history has been reviewed based on the percentage of commands used over a specific period of time. The logs are then scanned and mined [12] for interesting temporal and sequential patterns about a user's activity [13]. A successful misuse detection system must overcome many challenges. First of all, a user's profile may change over time. To handle dynamic profiles, learning algorithms are required to track user behavior and adapt to a dynamically changing concept. Chung et al. in [3] describe their misuse detection system, DEMIDS, for database applications. DEMIDS uses the access information of the user to the database tables, columns, and other structures to build a user profile to track the behavior of the user.

In the specific area of malicious email detection, however, we are unaware of published work directly relating to user query profile learning and abnormal query behavior detection. We implement a misuse detection warning by comparing user's email behavior to malicious email, learned through clustering, thus creating a new dimension to profile-based misuse detection.

Our method is different from the previous researches because we analyzed the entire features of malicious email instead of only boot-sector viruses or only Win32 binaries. Our technique is similar to data mining techniques that have already been applied to Intrusion Detection Systems by Lee *et al.* [14]. We applied a similar framework to the problem of detecting new malicious email viruses but included macros and script those Schultz *et al.* [15] and other researchers didn't provide.

3 Methodology

A Self-Organizing Map [16], or SOM, is a neural clustering technique. Having several units compete for the current object performs the SOM clustering. The unit whose weight vector is closest to the current object becomes the winning unit. The weight of the winning unit is adjusted as well as those of its neighbors. SOM assume that there is some topology among the input objects and the unit will take on this structure in space. The organization of these units is said to form a feature map. SOM is also capable of presenting the data points in one- or three-dimensional space. In order to find out the boundaries from results of SOM, we applied partitioning method. The most famous and commonly used partitioning methods are k-means and k-medoids, and their variation. The k-means algorithm is sensitive to outliers since an object with an extremely large value may substantially distort the distribution of data. Instead of taking the mean value of the objects in a cluster as a reference point, the medoid (representative object) can be used, which is the most centrally located object in a cluster. Thus, the partitioning method can still be performed based on the principle of minimizing the sum of the dissimilarities (distances) between each object and its corresponding reference point. This forms the basis of the k-medoids method. For example, PAM (partitioning around Medoids) [17], built in Splus, starts from an initial set of medoids and iteratively replaces one of the medoids by one of the non-medoids if it improves the total distance of the resulting clustering. PAM, use real object to represent the cluster, works effectively for small data sets, but does not scale well for large data sets. The k-medoids algorithm is more robust than k-means in the presence of outlier and noise because a medoid is less influenced by outliers or other extreme values than a mean. Therefore, we apply the k-medoids clustering to results of SOM. The procedure of the presented method is the training of the SOM and then applying the k-medoids clustering. The input vectors of the SOM are the benign email data and the input vectors of the k-medoids clustering are the prototype vectors of the SOM. The outline of the proposed method is shown in Figure 1.

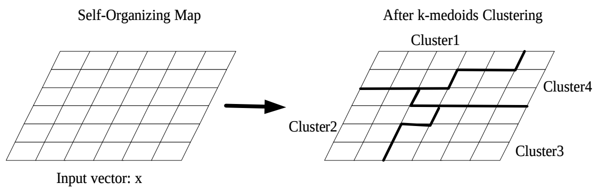


Fig.1. The outline of the proposed method

4 Experiment and Results

In this section we try to investigate some exploration features of email viruses. Every email virus is executed and tested in our lab. By observing and analyzing the behaviors of email viruses that we collected, we find some features that will help users and corporations to prevent from threat of email viruses. The purpose of our work was to explore the possibility of a standard technique to compute accurate detectors for new (unseen) malicious emails.

To evaluate the performance, we computed the false positive rate (FP rate) and the detection rate. The false positive rate is the number of benign emails that are mislabeled as malicious divided by the total number of benign emails. While the detection rate is computed as the number of malicious emails correctly classified divided by the total number of malicious emails tested. The overall accuracy of the algorithm is calculated as the number of emails the system classified correctly divided by the total number of emails tested.

4.1 Data set

We gathered a large set of emails from both public sources and our own UNIX email server. The sampling email viruses were downloaded from various FTP sites and were audited by commercial virus scanner. All these email viruses were discovered from 1999 to 2003. To standardize our data set, we used an updated Norton's virus scanner and labeled our emails as either malicious or benign emails. We statically extracted different features that represented different information contained within each email. These features were then used to generate detection models. The sample emails were separated into two classes: malicious and benign. Our data set consisted of a total of 2,203 programs split into 344 malicious emails and 1,859 benign emails. There were no duplicate emails in the data set and every email in the set is labeled as either malicious or benign by the commercial virus scanner. The type of gathered malicious email viruses were shown in Table 1, which consisted of IW (Internet worms), Trojans, Macros, Scripts, File-infectors, and so on. The contrasting benign data was also collected in order to perform relevance analysis. Using different features, we trained a set of classifiers to distinguish between benign and malicious emails. Note that the features extracted were static properties of the email and did not require execution.

Table 1. Samples of email viruses

Macro	MELISSA.A，GORUM.A
Executable File	ZAUSHKA.A-O，JERM.A，COBBES.A，MAGISTR.B，KAMIL.B，YOUGDOS.A
Trojan	PTWEAK.A ， GIFT.B ， HYBRIS.C ， SIRCAM.A ， TROODON.A ， XTC.A ， FEVER.A
Script	**JavaScript:** EXCEPTION.GEN，GERMINAL.A，EXITW.A，SEEKER.A6，ACTPA.A **VBScript:** REPAH.A，HARD.A，NEWLOVE.A，INFO.A，CHU.A，GOOFFY.A，NOONER.A ， ARIC.A ， KALAMAR.A ， CHICK.C ， CHICK.B ， CHICK.E ， ZIRKO.A ， LOVELETTER ， EDNAV.B ， HAPTIME.B ， HEATH.A ， HORTY.A ， VIERKA.B，GORUM.B，LIFELESS.A
Worm	NIMDA.A-O ， GOKAR.A ， ALIZ ， PETIK.C ， PET.TICK.Q ， BADTRANS.B ， GIZER.A ， BUGBEAR.A ， ENVIAR.BR ， ADIX.A ， UPDATR.A ， KLEZ.E ， KLEZ.H，LASTWORD.A，LOHACK.A，MERKUR.A，MYLIFE.E，PLAGE.A，PROLIN.A ， SOLVINA.B ， SHOHO.GEN ， DESOR.A ， PET.TICK.Q ， PETIK.E ， ZHANGPO.A，ZOHER.A，SOBIG.A，YAHA.E，BLEBLA.C，GAGGLE.C

Mail format descriptions involve many attributes and analytical characterization was performed. This procedure first removes irrelevant or weakly relevant attributes prior to performing generalization. Finally, from the email format, we extracted a set of features to compose a feature vector for each email as shown in Table 2.

Table 2. Extracted feature from email format

	Feature	Content
X_1	Mail content type	Text/plain/html，other
X_2	Mail size	Total size of email
X_3	MIME Format	Yes，No
X_4	Attachment file no	Number of attachments
X_5	Attachment size	Total size of attachments
X_6	Attachment file type	exe，doc，scr，pif……
X_7	Script language	VBScript，JavaScript…..
X_8	Subject	Re，Fw，Fwd，…
X_9	Carbon Copy	CC，BCC，…
X_{10}	Recipient	Single- Recipient，Multi-

4.2 Misuse detection with SOM and k-medoids

In order to detect unseen new email virus, we are going to incorporating the SOM network in misuse detection first. We build the network structure from training data and use the testing data to measure its performance. The training data and the testing data contain audit events. During training, the structure of SOM network is constructed based on malicious email data. During testing, we compute the accuracy of SOM network on the evidence of audit events of the testing data. We have used a sample of audit data contains normal activities and malicious activities.

First of all, we use Kohonen's Self-Organizing Map to organize benign email behavior into a two-dimensional map, according to emails' extracted features. Our input vectors consist of a set of malicious emails' features. The desired output is a two-dimensional map of N nodes (in this case, a 9×9 map of 81 nodes). The SOM algorithm has two parameters that change through iterations: the variance of the neighborhood function $\Lambda(n)$ (radial symmetric Gaussian function) and the learning rate $\eta(n)$ (n=1, 2 …). The adaptation laws for these parameters are presented as:

$$\eta(n)=0.9(1-n/1000),\ \ \Lambda(n)=\Lambda(n-1)(1-0.01n)$$

These parameters adaptation laws lead to a fast convergence of the algorithm without the lost of quality of its output. Using these laws together with the selective update of neurons weighs, there is a reduction of the algorithm complexity through iterations. The selective update consists of a threshold for the neighborhood function that allows only neurons above the threshold to be updated. Since the $\Lambda(n)$ decreases fast with time, so does the number of neurons to be updated.

Neural Networks are also sensitive to the number of neurons in their hidden layers. Too few neurons can lead to underfitting. Too many neurons can contribute to overfitting, in which all training points are well fit, but the fitting is uncertain about testing point. We adjust some parameters (neurons and attractive radius) through some experiments to make SOM have better performance. Using False positive rate and detection rate, to evaluation the performances of the SOM in various experiment designs status. We can calculate the detection rate of audit emails in testing data and gain the best performance, when use a 9×9 map of 81 nodes and the scale 5 of the attractive radius. The over accuracy rate of the different scale radiuses are presented in Table 3. Figure 3 shows the results in ROC curve of SOM with 4×4, 9×9 and 15×15 grids. We can observe that higher detection rare will incorporated a higher false positive rate in SOM. If we can obtain more training data, we may obtain a better result.

Table 3. The over accuracy rate(%) of the different radiuses in the same neurons

	Attractive radius		
Map	Scale 3	Scale 5	Scale 7
9×9	98.28%	98.28%	98.28%
4×4	98.64%	98.64%	98.28%
15×15	98.23%	98.37%	98.18%

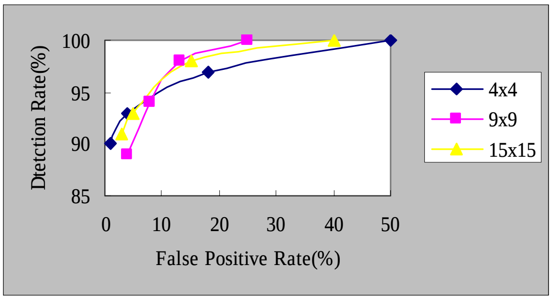


Fig. 2. The ROC curves of the SOMs

It is difficult to find clustering boundaries clearly in SOM. Therefore, we apply the k-medoids clustering after the training of the SOM. Assuming that there are only two clusters in the trained SOM map (malicious or benign). Applying the algorithm with 9×9 grids, Figure 3 shows the result of k-medoids clustering for the input vectors in SOM with testing data labeled clearly with "M" and "B".

Fig. 3. Result of SOM with k-medoids

Current most virus scanners technology use signature-based detector that detect new viruses [18]. The classic signature-based detection algorithm relies on signatures (unique telltale strings) of known malicious executables to generate detection models. Signature-based methods create a unique tag for each malicious program so that future examples of it can be correctly classified with a small error rate. These methods do not generalize well to detect new malicious binaries because they are created to give a false positive rate as close to zero as possible. Whenever a detection method generalizes to new instances, the tradeoff is for a higher false positive rate. Then, the accuracy of SOM with k-medoids clustering can be calculated and shown in Table 4 incorporated with anti-virus software. With a little false positive rate, our proposed method shows a high detection rate in the testing data.

Table 4. Comparison results (%)

Profile Type	Detection Rate (%)	F P Rate (%)	Overall Accuracy (%)
SOM(k-medoids)	88.08	1.88	95.53
Pcc2002	41.66	0	88.96
Pcc2003	86.90	0	97.52
Norton2002	30.95	0	86.93
Norton2003	82.14	0	96.62

One of the primary problems faced by the virus community is to devise methods for detecting new virus that have not yet been analyzed. Eight to ten viruses are created every day and most cannot be accurately detected until signatures have been generated for them [4]. During this time period, systems protected by signature-based algorithms are vulnerable to attacks. In order to find out the capability of the proposed method in detecting unseen email viruses, we found some new email viruses and tested our methods to see whether the proposed method can unearth them or not. These email viruses were found after May 2003. Therefore, they are not contained in our data set. Table 5 shows the testing result of anti-virus software and our proposed methods. Note that anti-virus software 2002 is not updated in 2003 and anti-virus software 2003 is not updated until May 2003. Since they are signature-based, we found that there are many email viruses that the anti-virus software 2002 and 2003 could not detect. However, all these email viruses were detected after we update our anti-virus software in October 2004. As a result, our proposed methods outperformed than some available anti-virus software in the detection of new unseen email viruses.

Table 5. Testing results of new email viruses (*"√"= detected*)

Virus Profile	SOM with k-medoids	SOM	Pcc2003	Norton2003	Pcc2002	Norton2002
NETSKY.C	√	-	-	-	-	-
NETSKY.D	√	-	-	-	-	-
MIMAIL.A	√	√		-	-	-
MYDOOM.A	√	√	-	-	-	-
PE_CIH.1003	√	√	√	√	√	√
WORM_YAHA.G	√	√	√	√	√	√
WORM_BAGLE.C	√	√	-	-	-	-
WORM_BAGLE.GEN-1	√	√	-	-	-	-
HTML_BAGLE.Q-1	√	-	-	-	-	-
WORM_BAGLE.J	√	√	-	-	-	-

5 Conclusion and Future Work

The contribution that we presented in this paper was a method for detecting different type of malicious emails included Macro and VBScript's attachments. We have presented a detection model that utilizes data mining methods to organize email virus in a domain to detect. Clearly the proposed method has generated clear clusters. The result of this system is very meaningful and can be easily incorporated with an email server to assist detection of malicious emails. Noted that the features extracted were static properties of the email and did not require executed. Furthermore, its evaluation of an attachment is based solely on the behavior of the email and not the contents of the attachment itself. That added the ability to detect both the set of known malicious emails and a set of previously unseen, but similar malicious emails. Viruses are consistently ranked as one of the most frequent security threats in organization and virus protection has become a major business. Therefore, understanding some aspects of protection are needed.

One of the most important areas of future work for this application is the development of more efficient algorithms. The current methods require a machine with a significant amount of memory to generate, and employ the classifiers. Another potential future work of proposed method is to make it into a stand-alone virus scanner and to port the algorithms to different operating systems. Finally, our future research will be investigating the scalability of the system, so that it can be incorporated with other detection models.

Acknowledgement: The author would like to give thanks to the National Science Council of Taiwan for grant NSC 92-2218-E-224-015- to part of this research.

References

1. Coulthard, A., Vuori, T.A.: "Computer viruses: a quantitative analysis", Logistics Information Management, vol.15, no.5/6, (2002) 400–409
2. Bridwell, L.: ICSA Labs' Eighth Annual Virus Prevalence Survey 2002 (2002). Available at http://www.icsalabs.com/2002avpsurvey/index.shtml
3. Chung, C. Y., Gertz, M., and Levitt, K.: A misuse detection system for database systems, In Third International IFIP TC-11 WG11.5 Working Conference on Integrity and Internal Control in Information Systems (1999), 159–178, Kluwer Academic Publishers.
4. Gryaznov, D.: Scanners of the Year 2000: Heuristics, Proceedings of the 5th International Virus Bulletin (1999)
5. Spafford, E. H.: The Internet worm program: an analysis, Tech. Report CSD–TR–823, Department of Computer Science, Purdue University (1988)
6. McGraw, G., Morrisett, G.: "Attacking malicious code: Report to the infosec research council", IEEE software, vol.17, no.5, (2000) 33–41
7. Michie, D., Spiegelhalter, J., Taylor, D. C. C.: Machine learning of rules and trees. In Machine Learning, Neural and Statistical Classification, Ellis Horwood (1994)
8. Kephart , J. O., Arnold, W. C.: Automatic Extraction of Computer Virus Signatures, 4th Virus Bulletin International Conference (1994) 178–184

9. Lo, R.W., Levitt, K.N., Olsson, R.A.: MCF: a Malicious Code Filter, Computers & Security, vol.14, no.6, (1995) 541–566
10. Kerchen, P., Lo, R., Crossley, J., Elkinbard, G., Ollson, R.:" Static Analysis Virus Detection Tools for UNIX Systems", Proceedings of the 13th National Computer Security Conference (1990) 350–365
11. Crawford, R., Kerchen, P., Levitt, K., Olsson, R., Archer, M., Casillas, M.: Automated Assistance for Detecting Malicious Code, Proceedings of the6th International Computer Virus and Security Conference (1993)
12. Ling, C.X., Gao, J., Zhamg, H., Qian, W., Zhang, H.: Improving encarta search engine performance by mining user logs, International Journal of Pattern Recognition and Artificial Intelligence (IJPRAI) 16 (2002) no. 8.
13. Sleznyov, A., Mazhelis, O.: Learning temporal patterns for anomaly intrusion detection, Symposium on Applied Computing (2000) 209–213
14. Lee, W., Stolfo, S., Mok, K.: A Data Mining Framework for Building Intrusion Detection Models, IEEE Symposium on Security and Privacy (1999)
15. Schultz, M. G., Eskin, E., Hershkop, S., Stolfo, S. J.: "MET: An Experimental System for Malicious Email Tracking", in Proceedings of the 2002 New Security Paradigms Workshop, NSPW-2002, Virginia Beach, VA: September 23rd - 26th (2002)
16. Kohonen, T.: The Self-organization maps, Proc. IEEE, vol.78, no.9, (1990) 1480–1481
17. Kaufman, L., Rousseeuw, P. J.: Finding Groups in Data: an Introduction to Cluster Analysis, John Wiley & Sons (1990)
18. White, Steve R., Swimmer, Morton, Pring, Edward J., Arnold, William C., Chess, David M., and Morar, John F.,: Anatomy of a Commercial-Grade Immune System, BM Research White Paper (1999). Available at: http://www.av.ibm.com/ScientificPapers/White/Anatomy/anatomy.html.

Image Processing and Human-Computer Interaction

A Statistical Learning-Based Method for Color Correction of Underwater Images

Luz A. Torres-Méndez and Gregory Dudek

Centre for Intelligent Machines, McGill University,
Montreal, Quebec, H3A 2A7, CA,
latorres,dudek@cim.mcgill.ca,
WWW home page: http://www.cim.mcgill.ca/~latorres/research.html

Abstract. This paper addresses the problem of color correction of underwater images using statistical priors. Underwater images present a challenge when trying to correct the blue-green monochrome shift to bring out the color visible under full spectrum illumination in a transparent medium. We propose a learning-based Markov Random Field (MRF) model based on training from examples. Training images are small patches of color depleted and color images. The most probable color assignment to each pixel in the given color depleted image is inferred by using a non-parametric sampling procedure. Experimental results on a variety of underwater scenes demonstrate the feasibility of our method.

1 Introduction

Image restoration in general, involves the correction of several types of degradation in an image. The restored image must be more suitable than the original image for a specific application. Traditionally, the most common sources of degradation are due to imperfections of the sensors, or in transmission. We are interested in restoring the color of underwater images. High quality image data is desirable for many underwater inspection and observation tasks. Particularly, vision systems for aquatic robots [2, 4, 7] must cope with a host of geometrical distortions: color distortions, dynamic lighting conditions and suspended particles (known as 'marine snow') that are due to inherent physical properties of the marine environment. All these distortions cause poor visibility and hinder computer vision tasks, e.g., those based on stereo triangulation or on structure from motion.

Underwater vision is plagued by poor visibility [10, 9] (even in the cleanest water). Additional factors are the ambient light, and frequency-dependent scattering and absorption, both between the camera and the environment, and also between the light source (the sun) and the local environment (i.e. this varies with both depth and local water conditions). The light undergoes scattering along the line of sight. The result is an image that is color depleted (typically appearing bluish), blurry and out of focus. In this paper, we focus on the specific problem of restoring/enhancing the color of underwater images. The term *color* refers to

© A. Gelbukh, R. Monroy. (Eds.)
Advances in Artificial Intelligence Theory
Research on Computing Science 16, 2005, pp. 151-160

the red, green and blue values (often called the color channels) for each pixel in an image.

Prominent blue color of clear ocean water, apart from sky reflection, is due to selective absorption by water molecules. The quality of the water determines its filtering properties. The greater the dissolved and suspended matter, the greener (or browner) the water becomes. The time of day and cloudiness of the sky also have a great effect on the nature of the light available. Another factor is depth, once at sufficient depth, no amount of filtration can effectively restore color loss. Due to the nature of underwater optics, red light diminishes when the depth increases, thus producing blue to grey like images. By 3m in depth there is almost no red light left from the sun. By 5m, orange light is gone, by 10m most yellow is also gone. By the time one reaches 25m only blue light remains [3]. Since many (if not all) of the above factors are constantly changing, we cannot really know all the effects of water.

Color recovery is not a simple linear transform since it depends on distance and it is also affected by quantization and even light source variations. We propose a learning based Markov Random Field model for color correction based on training from examples. This allows the system to adapt the algorithm to the current environmental conditions and also to the task requirements. As proposed in[5], our approach is based on learning the statistics from training image pairs. Specifically, our MRF model learns the relationships between each of the color training images with its corresponding color depleted image. Training images are small patches of regions of interest that capture the maximum of the intensity variations from the image to be restored. In the process it is important not to lose resolution or details which will create even a worse problem. To our knowledge this is the first formulation of using MRFs for the context of color correction of underwater images.

This paper is structured as follows. Section 2 briefly consider some of the related prior work. Section 3 describes our Markov Random Field model for color correction. Section 4 tests the proposed algorithm on two different scenarios with several types of experimental data each. Finally, in Section 5, we give some conclusions and future directions.

2 Related Work

There are numerous image retouching programs available that have easy-to-use, semi-automated image enhancement features. But since they are directed at land-based photography, these features do not always work with underwater images. Learning to manipulate the colors in underwater images with computer editing programs requires patience. Automated methods are essential, specially for real-time applications (such as aquatic inspection). Most prior work on image enhancement tend to approximate the lighting and color processes by idealized mathematical models. Such approaches are often elegant, but may not be well suited to the particular phenomena in any specific real environment. Color restoration is an ill-posed problem since there is not enough information in the poor colored image alone to determine the original image without ambiguity. In their work, Ahlen *et al.* [1] estimate a diffuse attenuation coefficient for three wavelengths using known reflectance values of a reference gray target that is present on all tested images. To calculate new intensity values they use Beer's Law, where the depth parameter is derived from images that are taken at different depths. Additional parameters needed are the image enhancements functions built into the camera. In general, their results are good, but the method's efficiency depends highly on the previously noted parameters. In [11] a method that eliminates the backscatter effect and improves the acquisition of underwater images with very good results is presented. Their method combines a mathematical formula with a physical filter normally used for land photography. Although the method does not perform color correction, the clarity achieved on the underwater images may allow for color correction.

3 Our MRF Approach for Color Correction

The solution of the color correction problem can be defined as the minimum of an energy function. The first idea on which our approach is based, is that an image can be modeled as a sample function of a stochastic process based on the Gibbs distribution, that is, as a Markov Random Field (MRF) [6]. We consider the color correction a task of assigning a color value to each pixel of the input image that best describes its surrounding structure using the training image pairs. The MRF model has the ability to capture the characteristics between the training sets and then used them to learn a marginal probability distribution that is to be used on the input images. This model uses multi-scale representations of the color corrected and color depleted (bluish) images to construct a probabilistic algorithm that improves the color of underwater images. The power of our technique is evident in that only a small set of training patches is required to color correct representative examples of color depleted underwater images, even when the image contains literally no color information. Each pair of the training set is composed by a color-corrected image patch with its corresponding color-depleted image patch. Statistical relationships are learned directly from the training data, without having to consider any lighting conditions of specific nature, location or environment type that would be inappropiate to a particular underwater scene.

3.1 The MRF Model

Denote the input color depleted image by $\mathrm{B} = \{b_i\}, i = 1, ..., N$, where $N \in \mathbf{Z}$ is the total number of pixels in the image and b_i is a triplet containing the RGB channels of pixel location i. We wish to estimate the color-corrected image $\mathrm{C} = \{c_i\}, i = 1, ..., N$, where c_i replaces the value of pixel b_i with a color value. The images in the training set are pairs of color depleted and color corrected regions of interest that capture the maximum of the intensity variations of the image to be restored. They are defined in a similar manner, let $\mathrm{B}_p^t = \{b_i^t\}$, $i = 1, ..., N_p^t$ be the color depleted training image p and $\mathrm{C}_p^t = \{c_i^t\}$, $i = 1, ..., N_p^t$, the corresponding color corrected image. $N_p^t \in \mathbf{Z}$ represents the total number of pixels in the training image p.

The color correction problem can be posed as a labeling problem. A labeling is specified in terms of a set of *sites* and a set of *labels*. Sites often represent image pixels or regions in the Euclidean space. Let $\mathcal{S}$ index a discrete set of N sites $\mathcal{S} = \{s_1, s_2, ..., s_N\}$, and $\mathcal{L}$ be the set of corresponding labels $\mathrm{L} = \{l_1, l_2, ..., l_N\}$, where each l_i takes a color value. The inter-relationship between sites define the *neighborhood system* $\mathcal{N} = \{N_s \mid \forall s \in \mathcal{S}\}$, meaning any collection of subsets of $\mathcal{S}$ for which 1)$s \notin N_s$, and 2)$s \in N_r \iff r \in N_s$. N_s is the set of *neighbors* of s and the pair $\{\mathcal{S}, \mathcal{N}\}$ is a graph in the usual way. Each site s_i is associated with a random variable F_i. Formally, let $\mathrm{F} = \{F_1, ..., F_N\}$ be a random field defined on $\mathcal{S}$, in which a random variable F_i takes a value f_i in $\mathcal{L}$. A realization $f = f_1, ..., f_N$, is called a *configuration* of F, corresponding to a realization of the field. The random variables F defined on $\mathcal{S}$ are related to one another via the neighborhood system $\mathcal{N}$.

F is said to be an MRF on $\mathcal{S}$ with respect to $\mathcal{N}$ if and only if the following two conditions are satisfied [8]:

$P(f) > 0$ (positivity), and

$P(f_i \mid f_{\mathcal{S}-\{i\}}) = P(f_i \mid f_{N_i}) \quad$ (Markovianity).

where $\mathcal{S} - \{i\}$ is the set difference, $f_{\mathcal{S}-\{i\}}$ denotes the set of labels at the sites in $\mathcal{S} - \{i\}$ and $f_{\mathcal{N}_i} = \{f'_i \mid i' \in \mathcal{N}_i\}$ stands for the set of labels at the sites neighboring i. The Markovianity condition depicts the local characteristics of F.

The choice of N together with the conditional probability distribution of $P(f_i \mid f_{\mathcal{S}-\{i\}})$, provides a powerful mechanism for modeling spatial continuity and other scene features. On one hand, we choose to model a neighborhood N_i as a square mask of size $n \times n$ centered at pixel location i. On the other hand, calculating the conditional probabilities in an explicit form to infer the exact MAP in MRF models is intractable. We cannot efficiently represent or determine all the possible combinations between pixels with its associated neighborhoods. Therefore, we avoid the usual computational expense of sampling from a probability distribution (Gibbs sampling, for example) and color correct a pixel value b_i with neighboorhood $\mathcal{N}_i$ by using a non-parametric sampling strategy that is easy to implement, generates good results and is fast to execute.

To compute the MAP estimate for a color value, one first need to construct an approximation to the conditional probability distribution $P(f_i \mid f_{N_i})$ and then sample from it. For each new color depleted value $c_i \in \mathrm{C}$ to estimate, the samples $(\mathcal{A})$, which correspond to the set of small image patches in the training set, are queried and the distribution of C_i is constructed as a histogram of all possible values that occurred in the samples. $\mathcal{A}$ is a subset of the real infinite set of all color images, denoted by $\mathcal{N}_{real}$.

Based on our MRF model, we assume that the color-depleted value c_i depends only of its immediate neighbors, i.e. of N_i. If we define a set

$$\Gamma(c_i) = \{N^\star \subset \mathcal{N}_{real} : \| N_i - N^\star \| = 0\} \tag{1}$$

containing all occurrences of N_i in $\mathcal{N}_{real}$, then the conditional probability distribution of c_i can be estimated with a histogram of all center color values in $\Gamma(c_i)$. Unfortunately, we are only given $\mathcal{A}$, i.e., a finite sample from $\mathcal{N}_{real}$, which means there might not be any neigbhorhood containing exactly the same characteristics in intensity and range as N_i in $\mathcal{A}$. Thus, we must use a heuristic which let us find a plausible $\Gamma'(c_i) \approx \Gamma(c_i)$ to sample from.

In the non-parametric approach, a color value c_p with neighborhood N_p, is synthesized by first selecting the most similar neighborhood (N_{best}) to N_p, i.e., the closest match to the region being filled in,

$$N_{best} = \underset{A_q \in \mathcal{A}}{\textbf{argmin}} \ \| N_p - A_q \|, \tag{2}$$

Second, the k neighborhoods A_q in $\mathcal{A}$ that are similar (up to a given threshold ϵ) to this N_{best} are included in $\Gamma'(c_p)$, as follows

$$\| N_p - A_q \| < (1 + \epsilon) \| N_p - N_{best} \| \tag{3}$$

The similarity measure $\| \ . \ \|$ between two generic neighborhoods N_a and N_b is defined as the weighted sum of squared differences (WSSD) over the two neighborhoods. The "weighted" part refers to applying a 2-D Gaussian kernel to each neighborhood, such that those pixels near the center are given more weight than those at the edge of the window. We can now construct a histogram from the color values c_p in the center of each neighborhood in $\Gamma'(c_p)$, and randomly sample from it. c_q is then used to specify c_p. For each successive augmented voxel this approximates the maximum *a posteriori* estimate.

Measuring the dissimilarity between image neighborhoods is crucial for obtaining quality results, especially when there is a prominent color (blue or green) as in underwater images. Color information can be specified, created and visualized by different color spaces (see [12] for more information about color spaces). For example, the *RGB* color space, can be visualized as a cube with red, green and blue axes. Color distance is a metric of proximity between colors (e.g. Euclidean distance) measured in a color space. However, color distance does not necessarily correlate with *perceived* color similarity. Different applications have different needs which can be handled better using different color spaces. For our needs it is important to be able to measure differences between colors in a way that matches perceptual similarity as good as possible. This task is simplified by the use of *perceptually uniform* color spaces. A color space is perceptually uniform if a small change of a color will produce the same change in perception anywhere in the color space. Neither RGB, HLS or CIE XYZ is perceptually uniform. We use the CIE *Lab* space, which was designed such that the equal distances in the color space represent equal perceived differences in appearance.

(a) (b)

Fig. 1. (a) The ground truth (color) image. (b) The simulated color depleted image (this is the test image to be color corrected by our algorithm).

4 Experimental results

We test the proposed approach in two different scenarios. In the first scenario, we use color underwater images available on the web [1] as our ground truth data. These images were taken with a professional camera. The second scenario, involves the acquisition of underwater video by our aquatic robot. Sections 4.1 and 4.2 describe these scenarios with the experimental results.

4.1 Scenario 1

In order to simulate the effects of water, an attenuation filter were applied to each of the color underwater image. Figure 1a shows the ground truth (color) image and Figure 1b, the simulated (color depleted) image after applying the attenuation filter. The images in the training set correspond to small image regions extracted from the ground truth image and the color depleted image (see Figure 2). These images correspond to regions of interest in terms of the variations in pixel color values, thus the intention is that they capture the intrinsic statistical dependencies between the color depleted and ground truth pixel values. The size of the neighborhoods in all experiments were 5×5 pixels, and the number of possible candidates k, was fixed to be 10. Figure 3a shows the training image

[1] http://www.pbase.com/imagine (used with the kindly permission of Ellen Muller.)

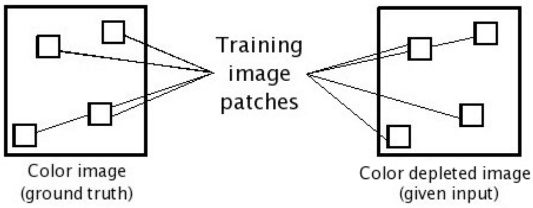
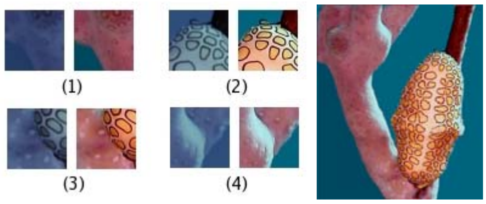


Fig. 2. Diagram showing how the training image pairs are acquired for the Scenario 1.

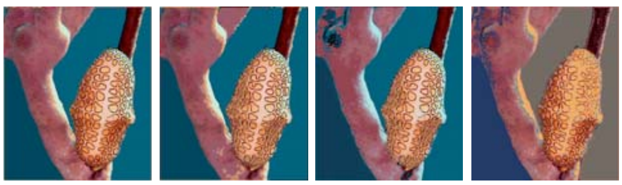


(a) (b)

Fig. 3. (a) The training image patches. (b) The color corrected image.

patches from where our algorithm learns the compatibility functions and Figure 3b shows the resulted image after running our learning-based method. The color-corrected image *looks* good, the discontinuities and edges are preserved since our method assign colors pixel by pixel, thus avoiding over-smoothing. Also, there are no sudden changes in color which are typically both unrealistic and perceptually unappealing. To evaluate the performance of our algorithm, we compute the mean absolute residual (MAR) error between the ground truth and the color corrected images. For this case, the MAR error is 4.4. Note that while our objective is perceptual similarity, this is difficult to evaluate and we use this objective measure to obtain quantitative performance data. For comparison purposes, we calculate the MAR error between the input (color depleted) image and the ground truth image, this is 28.85.

Using the same input image (Figure 3b), we now show how the final result varies depending on the training data. In Figure 4, 4 examples when using different training pairs are shown. For example, Figure 4a shows a color-corrected image when using training pairs (1) and (3) (see Figure 3a). The MAR errors are 5.43, 6.7, 8.54 and 25.8, respectively. It can be seen that the resulting images are limited to the statistical dependencies captured by the training pairs.



(a) (b) (c) (d)

Fig. 4. Color correction results using different training sets. The input image is shown in Figure 1b. The training pairs (labeled) are shown in Figure 3a. Results using training pair (a) (1), (2) and (3); (b) (1) and (2); (c) , and (d) (1).

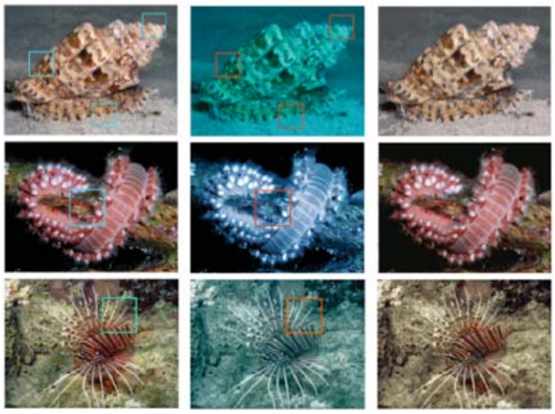


Fig. 5. The training pairs are indicated by the squares in the original and input images.

Three more examples of underwater scenes are shown in Figure 5. Each row shows from left to right, the ground truth color image, the input bluish image and the color corrected image after running our algorithm. The training image regions are shown by squares in the corresponding color and bluish images. In general the results looks very good. For the last two examples, the size of the image patches in the training set is very small and enough to capture all the statistical dependencies between bluish and color information, as a result, the number of total comparisons in our algorithm is reduced and speed is achieved. The average computation time for an input image of 300×400 pixels with small number of training pairs (4 or less) of size 50×50 is 40 seconds on generic PC's.

4.2 Scenario 2

The following scenario is of special interest as it can be applied to color correct underwater images in real-time with no user intervention. Our application is specifically for aquatic robot inspection. As our aquatic robot [7] swims through the ocean, it takes video images. In order to be able to correct the color of the images, training data from the environment that the robot is currently seeing needs to be gathered. How can better images be acquired? As light is absorbed selectively by water, not only does it get darker as you go deeper, but there is a marked shift in the light source color. In addition, there are non-uniformities in the source amplitude. Therefore, the aquatic robot needs to bring its own source of white light on it. However, due to power consumption, the light cannot be left turned on. Therefore, only at certain time intervals, the robot stops, turns its light on and take an image. These images are certainly much better, in terms of color and clarity, than the previous ones, and they can be used to train our algorithm to color correct neighboring frames (under the assumption that neighboring frames are similar). Figure 6 shows this scenario, here frame t_3 represents the image pair to be used to train our model for color correction.

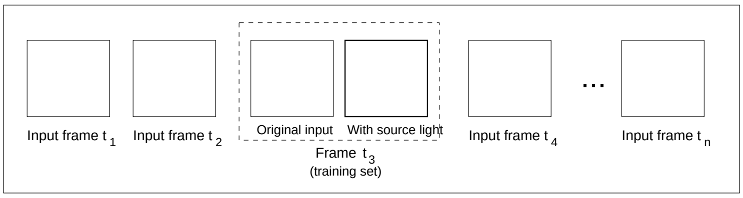


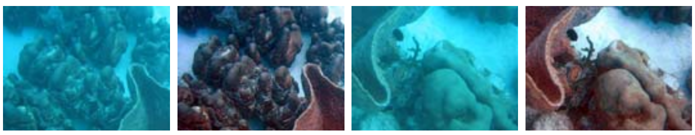
Fig. 6. The scenario 2.

Now we show an example. Figures 7a,b show the training image pair captured at time t. The robot moves around and then at time $t + \delta$ takes an image (Figure 7c), which is input to our algorithm. The resulting color-corrected image is shown in Figure 7d. Since we do not have ground truth data for this scenario, we cannot measure the performance of our algorithm, however it can be seen that the resulting image looks visually good.

Additional results on images, using the same training pair of previous example, are shown next. Figures 8a,c show the color depleted frames and Figures 8b,d, the color corrected images. Our algorithm performs very well in both examples. Note that the size of the training image pair is considerably bigger than those considered in previous section. The computation time is increased for the searching of best matching candidates in the training data. We have implemented a $kd-$tree structure on the training data, thus significantly decreasing computational burden, taking on average one minute for an image of 400×300 pixels. Details on this implementation are skip due to space limitations.

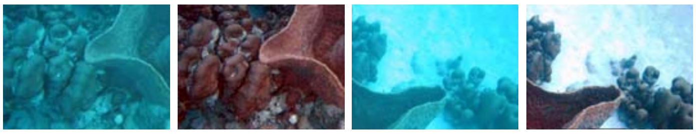
5 Concluding remarks

Color restoration and image enhancement are ubiquitous problems. In particular, underwater images contain distortions that arise from multiple factors making them difficult to correct using simple methods. In this paper, we show how to formulate color recovery based on using statistical learning constraints. This approach's novelty lies in using a pair of images to constrain the reconstruction. There are some factors that influence the quality of the results, such as the ade-



(a) (b) (c) (d)

Fig. 7. (a)-(b) The training image pair captured at frame t. (c) Image taken at frame $t + \delta$ and input to our algorithm. (d) The color corrected image.



(a) (b) (c) (d)

Fig. 8. (a)-(b) The training image pair captured at frame t. (c) Image taken at frame $t + \delta$ and input to our algorithm. (d) The color corrected image.

quate amount of reliable information as an input and the statistical consistency of the images in the training set. In some cases, ambiguities on local information is obtained, this is due to the fact that both the color corrected and the color depleted classifiers look only at a small image patches. Therefore, propagating information between regions can be used to resolve the ambiguity. More specifically, the marginal probability of the MRF model can be calculated by using belief propagation (BP) [13]. Results in this direction will be reported elsewhere.

References

1. J. Åhlén and D. Sundgren. Bottom reflectance influence on a color correction algorithm for underwater images. In *Proc. SCIA*, pages 922–926, 2003.
2. T. Boult. DOVE: Dolphin omni-directional video equipment. In *Proc.Int. Conf. Robotics and Automation*, pages 214–220, 2000.
3. J. Dera. *Marine Physics*. Elsevier, 1992.
4. G.L. Foresti. Visual inspection of sea bottom structures by an autonomous underwater vehicle. *IEEE Trans. Syst. Man and Cyber., Part B*, 31:691–795, 2001.
5. W.T. Freeman, E.C. Pasztor, and O.T. Carmichael. Learning low-level vision. *International Journal of Computer Vision*, 20(1):25–47, 2000.
6. S. Geman and D. Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Trans. on PAMI*, 6:721–741, 1984.
7. C. Georgiades, A. German, A. Hogue, H. Liu, C. Prahacs, A. Ripsman, R. Sim, L. A. Torres-Méndez, P. Zhang, M. Buehler, G. Dudek, M. Jenkin, and E. Milios. AQUA: an aquatic walking robot. In *Proc. of IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, volume 3, pages 3525–3531, Sendai, Japan, 2004.
8. J.M. Hammersley and P. Clifford. Markov field on finite graphs and lattices. In *Unpublished*, 1971.
9. S. Harsdorf, R. Reuter, and S. Tonebon. Contrast-enhanced optical imaging of submersible targets. In *Proc. SPIE*, volume 3821, pages 378–383, 1999.
10. J.S. Jaffe. Computer modeling and the design of optimal underwater imaging systems. *IEEE J. Oceanic Engineering*, 15:101–111, 1990.
11. Y.Y. Schechner and N. Karpel. Clear underwater vision. In *Proc. of Intt. Conference of Vision and Pattern Recognition*, volume 1, pages 536–543, 2004.
12. G. Wyszecki and W.S. Stiles. *Color Science: Concepts and Methods, Quantitative Data and Formulae*. Wiley, NewYork, 1982.
13. J. Yedidia, W. Freeman, and Y. Weiss. Understanding belief propagation and its generalizations. Technical report, Mitsubishi Elect. Research Labs., Inc., 2001.

Recognition of Shorthand Writing using Neural Networks

[1]Diana M. Vázquez, [1]Karla L. Segovia and [2]Roberto A. Vázquez
[1]Unidad Profesional Interdisciplinaria de Ingeniería y Tecnología Avanzadas –IPN
Departamento de Biónica
Av. Instituto Politécnico Nacional No. 2580
Ciudad de México, 07340, México.
Contact: dvazqueze@ipn.mx, ksegovia@ipn.mx

[2] Centro de Investigación en Computación – IPN
Av. Juan de Dios Batíz, esquina con Miguel Otón de Mendizábal
Ciudad de México, 07738, México.
Contact: ravem@ipn.mx

Abstract. Shorthand is any system of rapid handwriting which can be used to transcribe the spoken word as fast as people speak. In this paper is described a simple but efficient method for the recognition of shorthand signs. This method is based on a new technique for obtaining invariant descriptions from shorthand signs. The technique proposed is invariant to scaling, translations and deformations. In this technique, signs are divided into sub-regions, and those sub-regions are then described in terms of their geometric moments of order zero. Finally, the invariant description already obtained is normalized in order to train a neural network. The proposal is tested using a bank of 1100 signs written by 10 different people.

1 Introduction

Character recognition [6, 7] is an interesting task for many researches. Nowadays, OCR (Optical Character Recognition) systems recognize without difficulty printed characters, but handwriting is still a difficult task due to different ways of writing the same character. Due to OCR can not recognize handwriting, new tools knowing as ICR (Intelligent Character Recognition) were developed. An ICR [8] can recognize handwriting under affine transformations as rotation, translation, deformations and scale change, and also can recognize printed characters.

Shorthand is any system of rapid handwriting which can be used to transcribe a spoken word. Shorthand systems use special signs to represent phonemes, words and phrases. There are many different shorthand systems currently in use. The most popular ones include: Pitman shorthand method and Gregg shorthand methods. In Mexico the Pitman method was adapted to Spanish by Luis E. Maumejan [5] in 1903 and has 20 basic signs, see Table 1. Actually, the Pitman method adapted to Spanish is used in Mexico.

The use of shorthand methods combined with technology is a powerful tool, useful for every one who wants to write as fast as he speaks, and then having the possibility

© A. Gelbukh, R. Monroy. (Eds.)
Advances in Artificial Intelligence Theory
Research on Computing Science 16, 2005, pp. 161-170

of recovering what he wrote, converted into printed characters. However, when the signs are noisy and with a lack of uniformity, classification is a very difficult problem, because decision regions are hard to define in an optimal way.

First thing that has to be clear is that an OCR can not be used for recognizing shorthand signs due to OCR can not recognize handwriting signs. On the other hand, an ICR can recognize handwriting signs, but the recognition of shorthand signs is a difficult task; this because Pitman's system has basically two signs and the orientation and thickness of these signs are associated with only one sound. In other words, an ICR is not useful for recognizing shorthand signs due to this tool is invariant under rotations and the same sign rotated in different degrees has several associated sounds, see Table 1.

In this paper recognition of shorthand signs is proposed, the efforts are aimed to develop a simple but efficient technique for extracting an invariant descriptor of these sings. This technique is invariant to scaling, translations under *x* axis and deformations of the signs. First, a sign is obtained and divided into sub-regions, and then the geometric moment of order zero of each sub-region is computed. After that, the invariant description obtained is normalized and transformed into its binary version with the propose of train a neural network for its further recognition. Finally, the identity of the sign is determined by using the neural network already trained. The technique proposed is tested using 100 images of each sign wrote by 10 different people.

The rest of the paper is organized as follows. In section 2, the classification tools used in this research are described. In section 3, the proposal is described in detail. In section 4, numerical examples to better follow the functioning of the proposal are given. In section 5, experiment results with signs and words written by 10 different people are provided, while in section 6, conclusions and directions for further research are given.

2 Neural networks

Since the rebirth of neural networks, several models inspired in neurobiological process have emerged in the last years. Such models are often dedicated and incorporate some existing clustering or classification algorithm. Advantages of neural networks are adaptability, robustness, and easy of implementation. In this paper two neural networks are investigated and compared on their classification capabilities.

2.1 Self-Organizing Maps

Kohonen's Self-Organizing Map (SOM) [1] is one of the most popular artificial neural network algorithms. Kohonen, who presents an unsupervised learning network as an explanation of the existence of ordered maps in the brain, orders the output units such that not only the weight vectors to a winning unit am affected, but those to its neighbors as well, see figure 1(a). SOM learn to classify input vectors according to how they are grouped in the input space. They differ from competitive layers in that neighboring neurons in the self-organizing map learn to recognize neighboring sections of the input space. Thus, self-organizing maps learn both the distribution (as do competitive layers) and topology of the input vectors they are trained on.

The basic idea of a SOM is to map the data patterns onto an *n*-dimensional grid of neurons. That grid forms what is known as the output space, as opposed to the input space where the data patterns are. This mapping tries to preserve topological relations, i.e., patterns that are close in the input space will be mapped to neurons that are close in the output space, and vice versa.

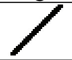
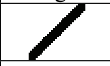
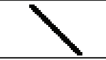
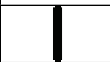
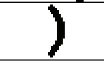
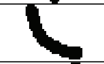
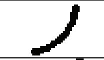
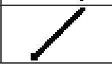
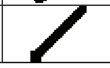
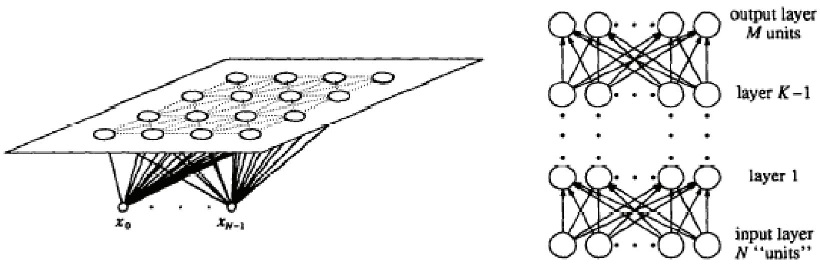
Sign	Sound	Sign	Sound	Sign	Sound	Sign	Sound
	(re)		(rre)		(ne)		(ñe)
	(que, ke)		(gue)		(pe)		(be)
	(te)		(de)		(me)		(es, ex, ez)
	(fe)		(ve)		(er)		(el)
	(le)		(ye, lle)		(che)		(je, ge)

Table 1. Signs used in Pitman's method. Sign (che) is equal to sign (re) but (che) is identified by the direction in which it is written (right-up to left-down); the same to (je, ge) and (rre).

2.2 Feed-Forward network

The general topology of what we call a K-layer' feed-forward network is depicted in figure 1(b). The weights and biases of each unit j in layer 1 describe a hyperplane in N-dimensional space. This hyperplane divides the input patterns in two classes, indicated by the activation value. In a feed-forward network, a unit in layer 2 represents a convex region in hyperspace, enclosed by the hyperplanes determined by the units of layer 1. When three layers are used, any arbitrary shape in N-dimensional space can be enclosed. For detail see [2].



(a) (b)

Figure 1. (a) Typical Kohonen network. (b) General topology of a K-layer feed-forward network.

3 The proposal

One important problem to solve in character recognition is the recognition of shorthand signs even if they are noisy or with lack of uniformity. Signs, written by people, often present distortions or small spurious, see Table 2. The technique proposed in this paper can recognize shorthand signs even if those signs present deformations.

The technique is based on dividing a sign in sub-regions. An invariant description of a given sign is computed using the information of each sub-region.

To compute the invariant description of each sign, we get an image of the sign and then proceed as follows:

```
1. Apply a standard threshold [2] to get a binary image.
2. Separate  the sign form its background
3. Obtain the invariant description.
```

Let **I** an image of length $m \times n$, $\mathbf{x} \in \mathbb{N}^m$ and $\mathbf{y} \in \mathbb{N}^n$ vectors used for isolating a sign are given by:

$$x_l = \sum_{k=0}^{n} i_{lk} \quad , \quad y_l = \sum_{k=0}^{m} i_{kl} \qquad \textbf{(1)}$$

xmin = l where the component $x_l \neq 0$ and l is the minimum, xmax = l where the component $x_l \neq 0$ and l is the maximum, ymin = l where the component $y_l \neq 0$ and l is the minimum, ymax = l where the component $y_l \neq 0$ and l is the maximum.

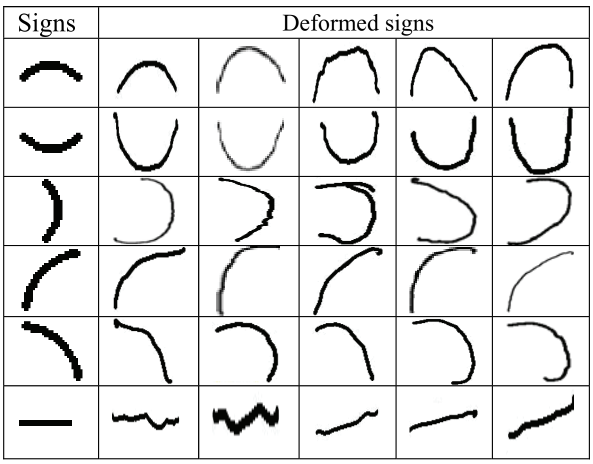


Table 2. Some distorted versions of shorthand signs.

In order to separate the sign from its background, the next steps are applied:

```
1. Compute vectors x, y using equation 1
2. For x and y compute xmin, xmax, ymin and ymax.
```

Finally, the sign is centered in a rectangle with vertexes (xmin,ymin) and (xmax,ymax).

3.1 Invariant descriptions of shorthand signs

After a sign is enclosed by the rectangle, the region obtained is divided into 25 sub-regions, see figure 2, and then an invariant description of the sign is obtained.

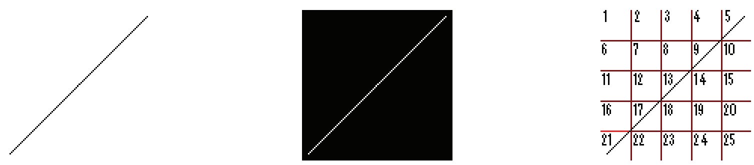
Invariant feature $\mathbf{r} \in \mathbb{N}^{25}$ is obtained computing the geometric moment of order zero [4] over each sub-region as follow:

$$r_i = \sum_x \sum_y x^0 y^0 \mathbf{R}_i(x, y) \qquad \textbf{(2)}$$

where $\mathbf{R}_i$ is a sub-region of the enclosed sign.

Finally, the feature vector is normalized and transformed into its binary feature vector version. This transformation is given by:

$$\overline{r}_\text{i} = \begin{cases} 1, & \text{if} \quad r_\text{i} \geq \dfrac{\max(\mathbf{r})}{3} \\ 0 & \text{otherwise} \end{cases} \qquad \textbf{(3)}$$



(a) (b) (c)

Figure 2. (a) Shorthand sign. (b) Shorthand sign enclosed by a rectangle. (c) Region divided into 25 sub-regions.

Thickness of the sign is defined by a threshold. If the density of $\mathbf{R}_i$ is less than the threshold the sign is thin, other case is thick.

In order to determined the associated sound to the sign, the image is divided in five regions: $\text{S}_\text{A}, \text{S}_\text{E}, \text{S}_\text{I}, \text{S}_\text{O}, \text{S}_\text{U}$. The position of the sign in the image (before of being isolated) determine its associated sound, see figure 3. If the center of mass of sign$\left(\bar{x},\bar{y}\right) \in \mathrm{S_A}$, the associated sound is (a) and so on. Just for remember center of mass is given by:

$$\bar{x} = \frac{m_{10}}{m_{00}}, \ \bar{y} = \frac{m_{01}}{m_{00}} \tag{4}$$

where $m_{10} = \sum_x \sum_y xf(x, y)$ and $m_{01} = \sum_x \sum_y yf(x, y)$.

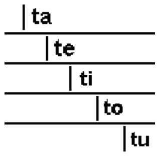


Figure. 3. Sound associated to each region.

4 Numerical examples

To better understanding of the proposal, let us look the next numerical examples. Suppose we want to obtain the invariant vectors from sign and its deformed versions showing in table 2.

First, an invariant vector using equation 2 is obtained. Next, the invariant vector already obtained is normalized and transform into its binary vector using equation 3.

Sign	Invariant vector (using eq. 2)	Normalizad vector (using eq. 3)
	r = [0 10 19 12 0 1 9 0 7 1 7 0 0 0 8 6 0 0 0 6 7 0 0 0 7]	$\overline{\mathrm{r}}$ = [01110 01010 10001 10001 10001]
	r = [0 11 21 20 0 0 9 0 4 8 5 3 0 0 5 10 0 0 0 7 16 0 0 0 7]	$\overline{\mathrm{r}}$ = [01110 01001 00000 10001 10001]
	r = [0 26 29 23 0 11 1 0 5 14 10 0 0 0 9 11 0 0 0 10 12 0 0 0 0]	$\overline{\mathrm{r}}$ = [01110 10001 10001 10001 10000]
	r = [0 7 21 21 11 7 15 0 0 13 14 0 0 0 10 10 0 0 0 8 9 0 0 0 4]	$\overline{\mathrm{r}}$ = [01111 11001 10001 10001 10000]
	r = [6 16 11 0 0 11 0 7 7 0 11 0 0 14 2 11 0 0 0 10 9 0 0 0 13]	$\overline{\mathrm{r}}$ = [11100 10110 10010 10001 10001]

As you can appreciate, although the signs appear with small spurious and deformations, the normalized vectors obtained are very close.

5 Experimental results

In this section, the proposal is tested with 100 images of each signs written by 10 different people. A SOM and feed-forward neural network were trained using the feature vector of each sign obtained with the proposal. In a second set of experiments, a feed-forward neural network was trained using the well-known first four Hu descriptors, invariant to translations, rotations and scaling, computed over the signs. In addition, commercial OCR's and ICR's were tested in the recognition of shorthand sings. Finally, 20 words written by 5 different people were use to test the performance of the technique, example of this words are given in Table 5. This experiment consists on translated some words written using signs into Spanish.

5.1 Training of the networks

Only the eleven thin signs whose images are shown in table 1 were used to train the networks. To each sign the feature vector was obtained as in section 3.

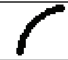
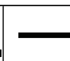
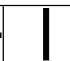
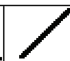
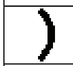
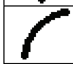
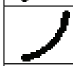
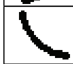
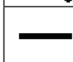
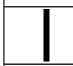
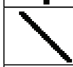
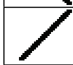
Sign												other
	8%	10%	0%	10%	3%	3%	0%	4%	0%	0%	0%	62 %
	0%	96%	1%	0%	0%	1%	0%	1%	0%	0%	0%	1%
	15%	2%	54%	0%	4%	8%	12%	0%	5%	0%	0%	0%
	0%	0%	0%	95%	0%	0%	0%	0%	0%	0%	0%	5%
	34%	10%	18%	0%	28%	0%	9%	0%	0%	0%	0%	1%
	0%	39%	0%	0%	0%	61%	0%	0%	0%	0%	0%	0%
	0%	0%	29%	0%	1%	1%	69%	0%	0%	0%	0%	0%
	0%	1%	0%	0%	1%	0%	0%	87%	5%	6%	0%	0%
	0%	0%	1%	0%	0%	0%	2%	0%	90%	1%	5%	2%
	0%	0%	0%	0%	0%	0%	0%	0%	0%	97%	1%	2%
	0%	0%	0%	0%	0%	0%	0%	0%	1%	0%	97%	2%
% of Recognition		**71%**										

Table 3. Percentage of classification using a feed-forward network.

Feed-forward neural network. It was trained using two layer; input layer with 25 neurons and output layer with four neurons. Training was performed using hardlim function and 50 epochs.

Self-organizing map. A 2-dimensional SOM network of length 25x11 was used. Initial weights were initializing to 0.5. Kohonen learning rate was set to 0.01 and the conscience learning rate was set to 0.001. Training was performed using Euclidian distance and 500 epochs.

5.2 Results

Using the feed-forward neural network combined with the proposal the performance obtained was of 71 %, Table 3 summarizes the classification results.

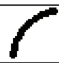
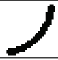
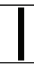
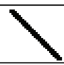
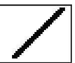
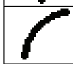
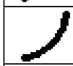
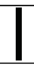
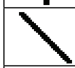
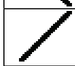
Sign											
	85%	0%	5%	5%	0%	0%	1%	4%	0%	0%	0%
	0%	83%	3%	0%	1%	11%	0%	1%	1%	0%	0%
	0%	0%	84%	0%	0%	0%	11%	0%	5%	0%	0%
	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%	0%
	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%	0%
	0%	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%
	1%	0%	0%	0%	0%	0%	99%	0%	0%	0%	0%
	0%	0%	0%	0%	1%	0%	0%	98%	0%	1%	0%
	0%	0%	0%	0%	1%	0%	3%	0%	92%	4%	0%
	0%	0%	0%	0%	0%	0%	2%	0%	0%	98%	0%
	0%	0%	0%	0%	1%	0%	0%	0%	0%	0%	99%
% of Recognition		**95%**									

Table 4. Percentage of classification using SOM.

In general, deformed instances of the signs were incorrectly classified, due to distortions and similarity between the feature vector obtained and the feature vector of each sign. In some cases the sign was not classified into any of the existing classes.

The performance of the proposal combined with SOM was of 95%, Table 4 summarizes the classification results. Majority of the deformed instances of the signs were correctly classified. Bad classification of the instances is due to deformation of the signs. These deformation causes that the feature vector was more similar to other class than its own class, i.e. the signs were classified in the classes of their most similar signs already learned.

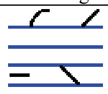
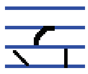
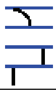
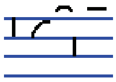
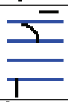
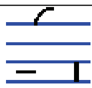
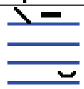
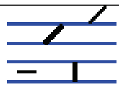
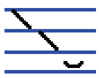
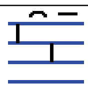
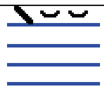
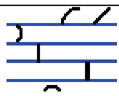
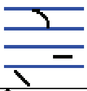
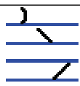
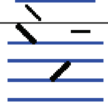
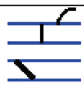
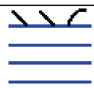
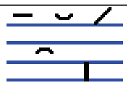
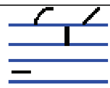
Spanish word	Pitman signs	% of recognition	Spanish word	Pitman signs	% of recognition
Colabora		100%	Pollito		100%
Tuerto		100%	Telematica		80%
Tuerca		80%	Colado		100%
Pagano		100%	Corredora		100%
Pepino		80%	Tematica		100%
Banana		100%	Estimuladora		80%
Puerco		100%	Aspero		100%
Barrica		100%	Botella		100%
Cotorro		100%	Papaya		100%
Caminadora		60%	Coladera		100%

Table 5. Percentage of translation from pitman signs to Spanish word.

Performance drastically decreases when Hu descriptors were used. This is due to those signs are the same sign but in different rotation and Hu descriptors are invariant to rotation therefore, the description obtained from some signs, for example (re, que, te and pe) or (fe, le, ne, me, er, el, es), is the same or the descriptions obtained are very close.

Three OCR were tested: ABBYY FineReader 7.0 Professional Edition, Microsoft Office Document Imaging, Iread Forms. Neither of them could recognize shorthand signs and handwriting. Soft writing 4.1 is an ICR which can recognize handwriting but in the recognition of shorthand signs had a low accuracy, signs (re, que, te and pe) were classified into the same class.

In last experiment 90% of the words were correctly recovered and translated. Incorrect translations were due to some signs incorrectly classified; Table 5 summarizes the classification results.

6 Conclusion and ongoing research

A new technique for obtaining invariant descriptors of shorthand signs has been proposed. The proposal is invariant to scaling, translation and deformations. The proposal has been tested with 100 images of each sign written by 10 different people.

Proposal was tested with two neural networks. The performance obtained using a feed-forward network was of 71 %, while using a self-organizing map the performance obtained increase to 95 %.

In addition, a comparison among the proposal, Hu descriptors, and others commercial OCR's was performed. Hu descriptors, OCR and ICR tools in this kind of signs were useless. This is due to those tools are invariant to rotation and the descriptions obtained from some signs were the same.

The results obtained (95%) support the effectiveness of the proposal and the advantage is clear against Hu descriptors, OCR and ICR tools. Translation of words written using pitman signs was also performed with an accuracy of 90%.

Nowadays, we are testing the proposal using associative memories in order to generalize the method for the recognition of handwriting.

References

1. T. Kohonen. Self-organized formation of topologically correct feature maps. *Biological Cybernetics,* 43:59-69, 1982.
2. F. Rosenblatt. Principles of Neurodynamics: Perceptrons and the theory of brain mechanisms. Spartan, Washington, DC, 1962.
3. N. Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on SMC*, 9(1):62-66. 1979.
4. M. K. Hu. Visual pattern recognition by moment invariants. *IRE Transactions on Information Theory*, 8:179-187. 1962.
5. Luis E. Maumejan. Manual de Fonografía. Escuela de taquigrafía, 1903.
6. P. Patrick van der Smagt. A comparative study of neural networks algorithms applied to optical character recognition. *Proc. of International conference on industrial and engineering applications of artificial intelligence and expert systems*, 2:1037-1044, 1990.
7. D. Mehr and S. Richfield. Neural net applications to optical character recognition. *IEEE First International Conference on Neural Networks*, 1987.
8. Ning Sun et al. Intelligent recognition of character using associative matching technique. *Proc. of Pacific Rim International conference on Artificial Intelligence*, 1:546-551, 1990.

Data Compression

Data Compression Using a Dictionary of Patterns

Ángel Kuri[1] and José Galaviz[2]

[1] Department of Computing, ITAM.
akuri@itam.mx
[2] Department of Mathematics, Facultad de Ciencias, UNAM.
jgc@fciencias.unam.mx

Abstract. Most modern lossless data compression techniques used today, are based in dictionaries. If some string of data being compressed matches a portion previously seen, then such string is included in the dictionary and its reference is included every time it occurs. A possible generalization of this scheme is to consider not only strings made of consecutive symbols, but more general patterns with gaps between its symbols. In this paper we introduce an off-line method based on this generalization. We address the main problems involved in such approach and provide a good approximation to its solution.
Categories and Subject Descriptors: E.4 [**Coding and Information Theory**]–*data compaction and compression*; I.2.8 [**Artificial Intelligence**]:Problem Solving, Control Methods, and Search–*heuristic methods*.

General Terms: Approximation algorithms, heuristics

Additional Key Words and Phrases: Combinatorial optimization, NP-hardness

1 Introduction

Most of the successful modern lossless compression methods used today are based in dictionaries, such as the Lempel-Ziv family [10]. In these approaches, strings of symbols that occur frequently in the data being compressed (which we call the *sample*) are included in a dictionary. Every occurrence of some of these strings in the sample is further replaced by a reference to its location in the dictionary. If strings in the dictionary are large and frequent enough, this scheme achieves sample compression.

These methods only consider strings of consecutive symbols. In this paper we will introduce a generalization by considering "strings" with gaps, whose symbols are no necessarily consecutive. This generalization will be called *meta-symbol* or pattern in the rest of paper. A similar concept is used in [2] for approximate string matching and is analogous to the concept of *motif* commonly used in computational biology [5].

© A. Gelbukh, R. Monroy. (Eds.)
Advances in Artificial Intelligence Theory
Research on Computing Science 16, 2005, pp. 173-182

In this paper a *meta-symbol* is a sequence of symbols in some alphabet, probably interspersed with *don't-care* symbols or gaps of arbitrary length. The symbol used to specify a gap is not in the alphabet and is called the *gap symbol*. Those symbols in a meta-symbol that are in the alphabet are called *solid symbols* or *defined symbols*. Hence, every string of one or more symbols in the alphabet is a meta-symbol, and every string that begins and ends with symbols in the alphabet, and has an arbitrary sequence of gap symbols and alphabet symbols in between, is also a meta-symbol. We can establish the formal definition of meta-symbol as follows:

Definition 1. Given a finite alphabet of symbols Σ, and a special symbol $\gamma \notin \Sigma$, called *gap symbol*, a meta-symbol p in Σ^+ is any string generated by the regular expression:

$$\Sigma \cup [\Sigma(\Sigma \cup \{\gamma\})^\star \Sigma]$$

There are several features associated with every meta-symbol:

- The *order* of the meta-symbol, denoted $o(p)$, is the number of solid or defined symbols in meta-symbol p.
- The meta-symbol *size*, denoted $|p|$, is the total length of meta-symbol considering both, the solid and gap symbols.

A meta-symbol can be specified in several ways. In this paper our representation is made up by two vectors that describe, respectively, the *contents*, and the *structure* of the meta-symbol. For example, if we use the underscore as the gap symbol, the following is an order 5 and size 10 meta-symbol:

$p_1 =$ `EL_M_E___A`.

that occurs in the string (see figure 1): $S_1 =$ `GTELKMIELAMAEPGLARTELVMWEQYYA`. Such meta-symbol can be specified by:

- The contents vector $\boldsymbol{c}(p)$ of the defined symbols in the meta-symbol p in the order they appear. In our example $\boldsymbol{c}(p_1) = [E, L, M, E, A]$. Obviously, the size (number of entries) of this vector is: $|\boldsymbol{c}(p)| = o(p)$
- The structure vector $\boldsymbol{d}$ of the distances between every consecutive symbol in $\boldsymbol{c}$. In our example $\boldsymbol{d}(p_1) = [1, 2, 2, 4]$. The size of this vector is: $|\boldsymbol{d}(p)| = o(p) - 1$.

We are interested in the search of meta-symbols that occur frequently in longer strings of symbols in Σ. We will call *sample* to any string in Σ^+ where the meta-symbol search takes place. Hence we will require additional features in order to characterize a meta-symbol relatively to the sample. Hence, given a meta-symbol p that occurs in a sample S we define:

- The meta-symbol *occurrence list*, denoted $L(p)$, as the increasing ordered list of absolute positions where p occurs in S (the first symbol in a sample is at position zero). In our previous example $L(p_1) = \{2, 7, 19\}$.

– The *meta-symbol frequency*, denoted $f(p)$, as the number of times meta-symbol p occurs in the sample S. In our example: $f(p_1) = 3$.
– The *rough coverage*, denoted $\mathrm{Cov}(p)$, as the formal number of symbols in S, generated (or covered) by all the occurrences of meta-symbol in the sample, that is the product of frequency times the meta-symbol order. In the example $\mathrm{Cov}(p_1) = 5 \times 3 = 15$.
– The *effective coverage*, denoted $\mathrm{Cov}_{\mathrm{eff}}(p)$, as the actual number of symbols in the sample covered by all the occurrences of meta-symbol. This is the rough coverage defined above, minus the number of symbols in the sample covered more than once by different meta-symbol occurrences. In our example, since the letter E in position 7 is covered twice by different occurrences of p_1, we have: $\mathrm{Cov}_{\mathrm{eff}}(p_1) = 14$.

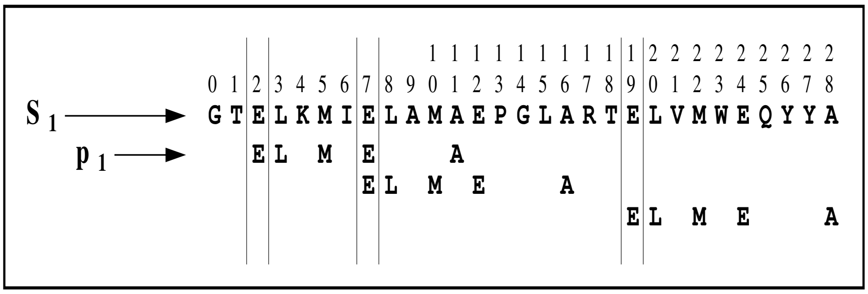


Fig. 1. The meta-symbol p_1 occurs three times in sample S_1.

2 Meta-Symbol-based Data Compression

Our goal is to achieve data compression building a dictionary of frequent meta-symbols found in the sample being compressed. Then every occurrence of a frequent meta-symbol in the sample is replaced by a reference to its position in the meta-symbol dictionary. Since such reference will be shorter than the meta-symbol itself, data compression is achieved.

In order to apply this procedure we need to take several steps:

1. Identify a set P of frequent meta-symbols in a given sample S.
2. Determine the subset $Q \subseteq P$ that maximizes the compression ratio if it is adopted as the dictionary.
3. Determine the best way for meta-symbol representation and encoding, and the best way for reference encoding. The compressed sample will include the, perhaps compressed, dictionary and the compressed representation of the original sample.

In the first step it is needed to find frequent meta-symbols in the sample. The kind of meta-symbols in definition 1 is very general, since the order and gap run-length is not bounded. The ony restriction is imposed by our concept of *matching*: we use rigid rater than flexible meta-symbols (where the gap run-length can vary in different meta-symbol occurrences). This is a very general, and well known, *pattern discovery* problem [9]. Unfortunately the reported algorithms for the discovery of patterns of the kind we are interested in, have exponential (on the sample size) time complexity [9]. However, if some prerequisites of order and frequency are established, there exist more efficient algorithms. In [6] is introduced the *Teiresias* algorithm, that performs pattern discovery given a minimum frequency and order. The patterns found are called *maximal*. A pattern p is maximal if for every other pattern q, such that p occurs inside q, there exist occurrences of p, where q does not occur. We found this approach very useful and, in order to approximate the first step, we use our own implementation of *Teiresias* algorithm.

The second step is a hard combinatorial optimization problem, in [4] we have shown that it is in the class of NP-complete problems, and here we propose the use of a heuristic in order to approximate the solution. Such approximation will be further optimized by the use of local optimizers (hill climbers).

In the third step, the decisions about the encoding mechanism for the sample re-expression and its dictionary, are guided (as those in the second step) by the the Rissanen minimum description length (MDL) principle [7, 1]. The goal is the simultaneous minimization of data required for the description of the sample being compressed and the model that describes the compression. In our approach we delegate the encoding decision to the compression algorithm, based on the compressed size of the whole set (dictionary + sample re-expression) obtained using different alternatives.

3 Selecting Good Subset of Meta-Symbols

The second step of the aforementioned process is the selection of a good subset of the whole set of frequent meta-symbols found in the sample. The “goodness” of a subset of meta-symbols is given by the compression ratio obtained if such subset of meta-symbols is the dictionary.

Obviously the best subset also must cover a considerably large amount of symbols contained in the sample, since every meta-symbol in the subset must have a good coverage. Also it must have a low amount of data symbols covered several times. That is, the best subset must have efficient meta-symbols: a large amount of covered symbols but a small amount of symbols covered by another meta-symbols or by different occurrences of the meta-symbol itself.

In order to evaluate such characteristic in every proposed meta-symbol we require another concept which we call *exclusive coverage*. Given an order set of meta-symbols Q (a list), the exclusive coverage of some meta-symbol p, denoted $\mathrm{Cov}_{\mathrm{exc}}(p, Q)$, is the total number of symbols in the sample, covered by all the occurrences of p, but not already covered by any other meta-symbol in Q.

Let S be a sample. We will denote by $|S|$ the number of symbols in such sample (its original size). Let P be a set of frequent meta-symbols found in S, and $Q \subseteq P$ a subset of frequent meta-symbols. Let $q' \notin Q$ be the meta-symbol made of all the symbols in S not already covered by any meta-symbol in Q, therefore the set $D(S,Q) = Q \cup \{q'\}$ is a possible dictionary to express S.

The word bit length needed to refer to any meta-symbol in D is:

$$r = r(Q) = \lceil \log_2 |D(Q)| \rceil = \lceil \log_2 (|Q| + 1) \rceil$$

If $f(q)$ denotes the frequency of meta-symbol $q \in Q$ in the sample S, then the expression of S in terms of dictionary references is:

$$E(Q) = \sum_{q \in D} r\, f(q) = \sum_{q \in Q} r\, f(q) + r = r \left[\sum_{q \in Q} f(q) + 1 \right]$$

Now we need to determine the dictionary size in bits. For every meta-symbol $q \in D(Q)$ we need to specify its $o(q)$ defined symbols and its $o(q)-1$ gap lengths. Let g be the bit length needed to express the gap runs, and let b be the bit length needed to express every single symbol in D. Therefore the dictionary size in bits is:

$$V(S,Q) = \sum_{q \in D} [b\, o(q) + g\,(o(q) - 1)] + M(S,Q)$$

where M denotes the overhead implicit by the inclusion of a model required to decode the single symbols in the dictionary. If the original symbols are encoded in the dictionary using Huffman encoding, for example, then b will be approximated by the average bit length of Huffman codewords and M is the amount of bits used to store the frequency table required to decode the Huffman codes. In a plain dictionary $M = 0$ and $b = (8\ bits)$.

We will denote with $T(S,Q)$ the size of the compressed sample using the subset of meta-symbols Q, hence:

$$T(S,Q) = V(S,Q) + E(Q)$$

Now we can define the compression ratio:

Definition 2. The *compression ratio* obtained by using the subset of meta-symbols Q is:

$$G(S,Q) = 1 - \frac{T(S,Q)}{|S|} \tag{1}$$

Denoting by p_i the i-th meta-symbol contained in a set of meta-symbols, the coverage of a subset of meta-symbols Q is:

$$\mathrm{Cov}(Q) = \sum_{p_i \in Q} \mathrm{Cov}_{\mathrm{exc}}(p_i, Q \setminus p_i) \tag{2}$$

We can now state our problem as follows:

Definition 3. OptimalMeta-SymbolSubset problem.
Given:

- A data sample S with $|S|$ symbols.
- A set $P = \{p_1, \ldots, p_n\}$ of frequent meta-symbols of S with frequencies $f(p_i) = f_i$ and sizes $t(p_i) = t_i$

Find a subset $Q \subset P$ that maximizes $G(Q)$ subject to the restriction:

$$\mathrm{Cov}(Q) \leq |S| \quad (3)$$

Hence we must find a subset of P. But there are $2^{|P|}$ of such subsets. This is a huge search space even for small values of $|P|$. In fact this is an NP-complete problem as we have shown in [4]. Similar problems have been proven NP-complete in [8, 3]. However in [8] the dictionary size is not considered, and the patterns are strings of consecutive symbols. In [3] also the dictionary size is ignored and coverage of patterns is used as the only criterion to determine the best subset.

Theorem 1. OptimalPatternSubset *Problem is NP-complete.*

Since ops is NP-complete we need heuristics in order to obtain an approximate solution for large samples. In what follows we will address a proposed heuristic.

4 Heuristics for Subset Selection

4.1 A first proposal.

For the approximate solution of ops problem we propose a greedy algorithm. Given a set P of frequent meta-symbols found in sample, in each step of our algorithm the meta-symbol with the greatest exclusive coverage is selected and removed from P. This process continues until all the symbols in the sample are covered or the meta-symbols that remain in P do not cover new symbols in the sample. All the meta-symbols selected in each step are stored in a set B, that is the proposed dictionary.

1. Let $B \leftarrow \emptyset$
2. Set the current coverage $Cv \leftarrow \emptyset$.
3. Select the meta-symbol $p \in P$ with highest exclusive coverage $\mathrm{Cov}_{\mathrm{exc}}(p, B)$.
4. Add p to B.
5. Remove p from P
6. Use the coverage of p to update the current coverage Cv.
7. Go to step 3 until $|Cv|$ equals the sample size or $P = \emptyset$ or $\mathrm{Cov}_{\mathrm{exc}}(q, B) = 0$ for every $q \in P$.

With the strategy described we will obtain the subset of meta-symbols B with highest coverage. But a good coverage does not guarantee the best compression ratio. It may occur that meta-symbols with large size and poor frequency or conversely are included in B. Since we want the inclusion of some meta-symbol in the dictionary to be well amortized by its use in the sample, this is not desirable.

4.2 Optimization by local search.

For the improvement of the heuristic described above, we will use hill climbing. Two different hill climbers will be defined:

MSHC *Minimum Step Hill Climber.* Searching the better binary string in the Hamming distance neighborhood of radius 1. Given a binary string that encodes a subset, we perform a search for the best string between those that differ from the given one in only one bit.

MRHC *Minimum Replacement Hill Climber.* Searching the better binary string in the Hamming distance neighborhood of radius 2. Given a binary string, we perform a search for the best string between those that exchanges the position of every bit with value 1 with the position of those bits with value 0.

To find a better subset than the one given by the coverage-based heuristic we sequentially run both hill climbers: the string obtained by the heuristic is passed to MSHC, and the output of this climber is thereafter passed to MRHC. Every hill climber is executed iteratively until no further improvement is obtained or a given number of evaluations is reached.

5 Experimental results

In order to test the effectiveness of the heuristic methods above, and the whole compression algorithm, some experiments were done. We generated a set of random samples with different sizes. Such samples were built using a short meta-symbol alphabet. Every sample was compressed using the method described in this paper, which we call *Pattern-based Data Compression* (PBDC), and other well known methods: Huffman codes (HUF), adaptive Huffman (AHUF), arithmetical encoding (ARIT), Lempel-Ziv-Welch (LZW), and gzip (GZIP), which is a variant of LZ77. Since the samples were built with a pre-defined set of meta-symbols, then we are able to calculate, *a priori*, the compression achieved if such set is defined as the dictionary of meta-symbols.

For every sample size included in the test, we generate as many samples as are needed in order to obtain a confidence interval for the mean of compression ratio with a 95% certainty and a radius below the 5% of mean magnitude. The results obtained are shown in table 1. The leftmost column contains the sample size in bytes, the column "Sam" specifies the number of random samples that were needed, the column labeled "±" contains the radius around the mean value that determines the confidence interval at 95%, "StdDev" stands for the standard deviation. The rightmost column in the table contains the theoretical compression ratio, considering the built-in meta-symbols as the dictionary.

Probably the reader will notice an apparent contradictory result in the first row of the table, since the compression ratio obtained by PBDC is greater than the one predicted by theory in the last column. This phenomenon may occur in short samples, where the accidental combination of dictionary meta-symbols can produce even better meta-symbols with significantly high probability.

Size	Sam	PBDC	±	StdDev	AHUF	HUF	ARIT	LZW	GZIP	Known
128	12	0.340	0.0057	0.0100	0.118	0.070	0.254	0.150	0.001	0.297
256	40	0.506	0.0262	0.0844	0.104	0.080	0.342	0.260	0.202	0.586
384	63	0.601	0.0300	0.1216	0.181	0.163	0.426	0.336	0.311	0.680
512	66	0.680	0.0337	0.1397	0.319	0.314	0.513	0.420	0.430	0.729
1024	67	0.739	0.0357	0.1489	0.228	0.225	0.569	0.434	0.535	0.799
2048	68	0.764	0.0379	0.1596	0.241	0.241	0.679	0.500	0.704	0.836

Table 1. Comparison of PBDC with other compression methods (sample size in bytes).

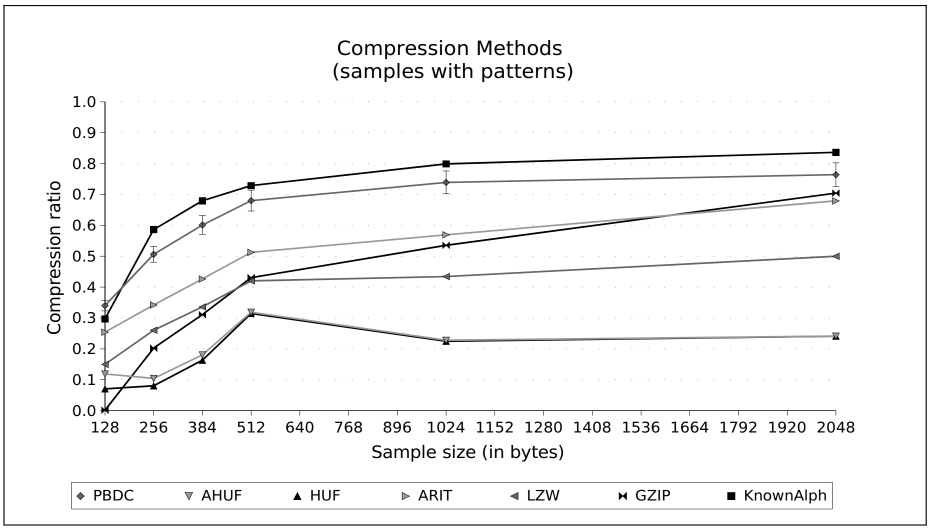


Fig. 2. Comparison of average compression ratios for several algorithms on samples built with meta-symbols.

The data in table 1 are also shown in figure 2, where it is more clear that the PBDC method outperforms all other methods. It is also clear that the compression ratio achieved by PBDC is close to the theoretical one.

6 Summary and Further Work

We have defined a very general compression method based on a dictionary of frequent meta-symbols. The meta-symbols in such dictionary are called *meta-symbols*, since we can think that the sample was generated by a source that produces such meta-symbols instead of the single symbols in the original alphabet. Therefore, working under the hypothesis that the sample is composed by meta-symbols, we try to find its meta-symbolic alphabet or dictionary, made up by those meta-symbols that minimize the expression of the sample together with the model that describes its composition. Since the search of such meta-alphabet is a hard problem, we propose coverage-based heuristic to approximate its solution.

In order to refine the solution proposed by this heuristic, two different methods of hill climbing where introduced: MSHC and MRHC. These hill climbers are executed on the solution proposed by the heuristic described.

Our results show that the approximation proposed by the heuristic and further refined by the local optimization performed by hill climbers, provides a very good solution for the problem addressed and outperforms several usual data compression algorithms in samples composed by meta-symbols.

Further work is required in two different ways: to test the method with samples not necessarily composed by meta-symbols and to optimize the performance of the meta-symbol discovery algorithm used.

References

[1] Barron, A., J. Rissanen and B. Yu, "The Minimum Description Length Principle in Coding and Modeling", *IEEE Transactions on Information Theory*, 1998, pp. 2743–2760.

[2] Burkhardt, Stefan and Juha Kärkkäinen, "Better Filtering with Gapped q-Grams", *Proceedings of 12th Annual Symposium on Combinatorial Pattern Matching CPM 2001*, Amihood Amir and Gad M. Landau (editors), Lecture Notes in Computer Science, No. 2089, 2001, pp. 73-85.

[3] Klein, Shmuel T., "Improving Static Compression Schemes by Alphabet Extension", *Proceedings of 11th Annual Symposium Combinatorial Pattern Matching CPM 2000*, Raffaele Giancarlo and David Sankoff (editors), Lecture Notes in Computer Science, No. 1848, 2000, pp. 210-221.

[4] Kuri, A. and J. Galaviz, "Pattern-based Data Compression", in *MICAI 2004: Advances in Artificial Intelligence*, Raul Monroy (Editor), Springer Verlag, Lecture Notes in Artificial Intelligence 2972, 2004, pp. 1–10.

[5] Parida, L. *Algorithmic Techniques in Computational Genomics*, Doctoral Dissertation, Courant Institute of Mathematical Sciences, University of New York, 1998.

[6] Rigoutsos, I. and A. Floratos, "Combinatorial Pattern Discovery in biological sequences: The Teiresias Algorithm", *Bioinformatics*, Vol. 14, No. 1, 1998, pp. 55–67.

[7] Rissanen, J., "Modeling by Shortest Data Description", *Automatica*, Vol. 14, 1978, pp. 465–471.

[8] Storer, James y Thomas Szymanski, "Data Compression via Textual Substitution", JACM, Vol. 29, No. 4, october 1982, pp. 928-951.

[9] Vilo, Jaak, *Pattern Discovery from Biosequences*, PhD Thesis, Technical Report A-2002-3, Department of Computer Science, University of Helsinki, 2002.

[10] Ziv, Jacob and Abraham Lempel, "A Universal Algorithm for Sequential Data Compression", *IEEE Transactions on Information Theory*, Vol. 23, No. 3, 1977, pp. 337-343.

MSIM: A Pattern Based Lossless Data Compressor

[1]Angel Kuri-Morales, Oscar Herrera-Alcántara

[1]Instituto Tecnológico Autónomo de México.
Río Hondo No.1, Tizapán San Angel, C.P. 01000, MÉXICO.
akuri@itam.mx

Abstract. We present a lossless data compression method based on the identification of unbounded patterns within a message. The patterns are called *metasymbols* and consist of groups non-necessarily neighboring symbols. Metasymbols are selected under the criterion that the original message can be re-expressed in the most compact way using an efficient coding scheme. We have shown that the problem of discovering metasymbols is NP-Hard and as an approach to the solution we developed an algorithm called MSIM. The performance of MSIM was tested on a set of messages that contains known metasymbols. Then we use a reference compressor (REF) which yields a compression function K to evaluate the theoretical limit of compression and then compare the performance of MSIM relative to REF. MSIM has a statistically significant very similar performance to REF and represents not only a compressor but also an excellent tool to discover unknown patterns in arbitrary messages.

1 Introduction

During the last decades several methods for lossless data compression have been proposed such as RLE[1], Huffman Coding[2], Arithmetic Coding[3], LZ[4], BWT[5], and PPM[6]. We present a method for lossless data compression where an original message *m* is re-expressed in the most compact way by identifying patterns. Patterns are called *metasymbols* and groups non-necessarily symbols in *m*.

Compressing with metasymbols has two main motivations:

1. To achieve maximum compression for *m*.
2. To identify high order structures in arbitrary sets of data.

The problem of identifying the metasymbols that allow us to re-express *m* in the most compact way is NP-Hard[7]. Considering this we appeal to soft computing techniques and particularly to a Genetic Algorithm called MSIM that identifies the already mentioned metasymbols and achieves maximum compression. In section 2 we describe MSIM, in section 3 we discuss its efficiency to discover metasymbols, in section 4 we present some experimental results. Finally, in section 5 we offer our conclusions. The performance of MSIM as a compressor and as a tool for metasymbolic discovery is supported statistically analyzing a set of 10,500 messages. Experimental results are satisfactory and show that MSIM does not depends of the content of the messages and is able to identify structural patterns. We show that MSIM performance is similar to the theoretical compressor REF which has *a priori* knowledge of the metasymbols. From the analysis we confirm that, if there are metasymbols in

© A. Gelbukh, R. Monroy. (Eds.)
Advances in Artificial Intelligence Theory
Research on Computing Science 16, 2005, pp. 183-192

the message, MSIM will discover them with statistical certainty. Future applications include protein sequences analysis and time series classification based on their plausible compressibility.

We define compression as

$$\text{Compression} = \frac{\text{length of } m}{\text{length of compressed message}} \tag{1}$$

2 Compression with metasymbols

We model a message *m* as an one-dimensional array of symbols; metasymbols are groups of these symbols and have four basic properties:

1. Order $|M_i|$. The number of symbols that the i-th metasymbol includes.
2. Frequency $|f_i|$. The number of repetitions of the i-th metasymbol in *m*.
3. Gaps. The spaces between the symbols of a metasymbol.
4. Positions. The indices of the first symbol of all the instances of a metasymbol.

As an example consider an hypothetical message m = "$A_0a_1D_2b_3E_4c_5F_6d_7e_8B_9f_{10}g_{11}C_{12}h_{13}i_{14}j_{15}k_{16}D_{17}l_{18}E_{19}m_{20}F_{21}n_{22}a_{23}o_{24}A_{25}p_{26}A_{27}q_{28}r_{29}s_{30}t_{31}u_{32}v_{33}B_{34}w_{35}B_{36}C_{37}x_{38}C_{39}D_{40}y_{41}E_{42}D_{43}F_{44}E_{45}z_{46}F_{47}$" with $|m| = 48$ and where the positions of the symbols are denoted by subindices. To ease visualization we include a two-dimensional matrix for *m* as we show in figure 1.

	0	1	2	3	4	5	6	7
0	A	a	D	b	E	c	F	d
8	e	B	f	g	C	h	i	j
16	k	D	l	E	m	F	n	a
24	o	A	p	A	q	r	s	t
32	u	v	B	w	B	C	x	C
40	D	y	E	D	F	E	z	F

Fig. 1. A bi-dimensional representation of a message

We can appreciate a first metasymbol ("ABC") at positions 0, 25 and 27 with gaps of size 9 and 3 respectively. In what follows we will use the representation $M_1=A_9B_3C$: 0, 25, 2 which is read: There is an "A" at position 0, a "B" 9 positions away from "A" and a "C" 3 positions away "B"; the metasymbol is repeated at position 25 and 2 away from 25 (25+2=27). A second metasymbol is $M_2=D_2E_2F$: 2, 15, 23, 3. The symbols that do not form repeated patterns are lumped in a pseudo-metasymbol we call the *filler*. One possible efficient encoding depends on the fact that the filler is easily determined as the remaining space once the other metasymbols are known. Hence

$$M_{filler}=\text{abcdefghijklmnaopqrstuvwxyz}$$

Metasymbolic search consists not only in the identification of repeated patterns but also in the selection of those which imply the minimum number of bits to re-express *m*. In the last example we assume, for simplicity, that metasymbols M_1, M_2 and M_{filler} comply with this criterion but, as we pointed out above, this problem is NP-hard. Therefore, we have developed an efficient Genetic Algorithm to tackle it.

We describe the proposed encoding scheme:

Let λ be the number of bits of each symbol (in the example λ=8)

Let μ be the number of metasymbols (in the example μ=3)

We use nibble[1] 'a' to indicate the value of λ in binary[2]. In the example a=1000_2.

We use nibble 'b' to indicate the number of bits required to code the value of μ in binary. In the example b=0010_2 and μ is coded as $\mu=11_2$.

Likewise, the maximum gap for M_1 is 9, and we need $g_1=\log_2\lceil 9+1\rceil$=3 bits to encode this value in binary. The other gaps for M_1 will be encoded with g_1 bits.

We use nibble 'c_1' to indicate the number of bits required by the g_1 value. In the example $c_1=0011_2$.

The maximum position for M_1 is 25 and we need $p_1=\log_2\lceil 25+1\rceil$=5 bits to encode this value in binary. The positions of the other instances of M_1 will be encoded with p_1 bits.

We use nibble 'd_1' to indicate the number of bits required by the p_1 value. In the example $c_1=0101_2$

$|M_1|$=3 and we need 3λ bits to encode its content.

The maximum gap for M_2 is 2, and we need $g_2=\log_2\lceil 2+1\rceil$=3 bits to encode this value in binary. The other gaps for M_2 will be encoded with g_2 bits.

We use nibble 'c_2' to indicate the number of bits required by the g_2 value. In the example $c_2=0011_2$.

The maximum position for M_2 is 23 and we need $p_2=\log_2\lceil 23+1\rceil$=5 to encode this value in binary. The positions of the other instances of M_2 will be encoded with p_2 bits.

We use nibble 'd_1' to indicate the number of bits required by the p_2 value. In the example $c_2=0101_2$.

$|M_2|$ = 3 and we need 3λ bits to encode this content.

Also, $|M_{filler}|$=30 and is simply an enumeration of values, Therefore, we need 30λ bits to encode it.

The number of bits for *m* expressed as metasymbols is given by

$$K = a + b + \mu + \sum_{i=1}^{M-1} (|M_i - 1| g_i + c_i + d_i + |M_i| \lambda + f_i p_i) + \\ + \lambda (|m| - \sum |M_i| f_i) \quad \mathbf{(2)}$$

[1] By "nibble" we mean 4 consecutive bits

[2] We simply put "binary" when referring to weighted binary

2.1 Metasymbolic search

The GA developed for metasymbolic Search (MSIM) is based on the Vasconcelos's Genetic Algorithm[8] which codes individuals as binary strings and has three operators: selection, mutation and crossover. VGA's mutation assumes a binary coding of the individuals, mutation replaces a bit's value by its complement with probability P_m. VGA Crossover requires two individuals and interchanges their genetic material with probability Pc. Selection is deterministic and elitist.
In the case of MSIM, the individuals are permutations of the array of indices of *m* in the interval [0, |*m*|-1] and each index is coded in binary requiring of $\lceil \log_2 |m| \rceil$ bits. A metasymbol's span is gotten from the indices of an individual by looking for an ascending order and reading from left to right. This representation warrants that each symbol belongs exclusively to one metasymbol as shown in Figure 2.

[0 9 12] [2 4 6] [1 3 5 7 8 10 11 13 14 15 16 18 20 22 23 24 26 28 29 30 31 32 33
35 38 41 46] [40 42 44] [43 45 47] [25 34 37] [27 36 39] [17 19 21]

M_1 $A_0B_9C_{12}$, $A_{25}\,B_{34}\,C_{37}$, $A_{27}\,B_{36}\,C_{39}$
M_2 $D_2\,E_4\,F_6$, $D_{17}\,E_{19}\,F_{21}$, $D_{40}\,E_{42}\,F_{44}$, $D_{43}\,E_{45}\,F_{47}$
M_{Filler} $a_1b_3c_5d_7e_8f_{10}g_{11}h_{13}i_{14}j_{15}k_{16}l_{18}m_{20}n_{22}a_{23}o_{24}p_{26}q_{28}r_{29}s_{30}t_{31}u_{32}v_{33}w_{35}x_{38}y_{41}z_{46}$

Fig. 2. A hypothetical individual of MSIM

A= 9 8 4 | **5** 6 7 | 1 3 **2** 10
B = 8 7 1 | **2** 3 10 | 9 **5** 4 6
A'= 9 8 4 | **2** 3 10 | 1 6 **5** 7
B'= 8 10 1 | **5** 6 7 | 9 **2** 4 3

Fig. 3. Crossover operator for MSIM

MSIM does not work on the binary genotype as VGA does; it has special operators that work on the phenotype (indices represented in binary) and maintains the individuals in the feasible region. MSIM has 3 operators: mutation, crossover and catastrophe. Mutation interchanges two indices of an individual and crossover is an operator that acts on two individuals (A, B) and uses PMX[9] to interchange the indices from one individual to the other. It ensures that there are no repeated indices in the first individual and that the same indices are missing in the other. Fig. 3 Illustrates PMX.
2.1.1 Catastrophe. An inconvenience of K is that it considers the bits used by the re-expressed message if the metasymbols are known; to know the metasymbols, on the other hand, we need to minimize K. This leads to a vicious circle whose way out is described next. We propose an on-line method for metasymbol searching. Note that the first 3 terms a, b and μ of ec. 2 may be neglected (they represent a few bits) and that the fourth term (the summation) represents the number of bits used by all metasymbols. Finally, the fifth term represents the number of bits required by the filler. From these considerations we define a discriminant d_i that corresponds to the number of bits required by the i-th metasymbol. Hence the fourth term can be expressed as

$$\sum_{i=1}^{M-1} d_i = \sum_{i=1}^{M-1} (|M_i - 1| g_i + c_i + d_i + |M_i| \lambda + f_i p_i) \quad (3)$$

where

$$d_i = |M_i - 1| g_i + c_i + d_i + |M_i| \lambda + f_i p_i \quad (4)$$

The main advantage of d_i is that it does not depend of the other metasymbols and, consequently, we can search metasymbols iteratively by comparing their d_i values and selecting the one with minimum value. The catastrophe operator is in charge of this task: it encodes the metasymbols as individuals of MSIM and inserts them as potential individuals into the population. The algorithm for the catastrophe algorithm is as follows:

Let i = 1
Mark all the symbols of *m* as free nodes, FreeNodes = |*m*| and LEN = 1
Do
  D= Number of different symbols of *m*
  For j=1 **to** j=D
    s_{j0} = j-th different symbol of *m*
    Mark as used all instances of s_{j0}
    Do
      Identify the symbol s_{jLEN} with the highest frequency (f_{jLEN}) nearest to any s_{j0} (to g_{LEN} positions) and mark all the instances of s_{jLEN} as free nodes. If there is more than one symbol with the same frequency, select s_{jLEN} randomly.
      Free the symbols s_{jk} with $0 \leq k \leq$ LEN. If s_{jLEN} is not at g_{LEN} positions of any instance of sj_0 then make LEN = LEN+1
      Evaluate the compression function and do d_{jLEN} = K
    While $f_{jLEN} > 1$
  next j
  Select the metasymbol $M_j = s_{j0}s_{j1}s_{j2} ...s_{jLEN}$ such that it has minimum d_{jLEN}
  Mark as used nodes the symbols used by the M_i metasymbol in all its instances
  Update the number of free nodes of *m*
  i=i+1
While FreeNodes>0 and $d_{jLEN} > 1.0$
Free nodes are grouped in the filler
The number of metasymbols is M = i+1

The fitness function for the individuals of MSIM is K and the individuals evolve to the best solution with MSIM's three operators.

3 Experiments

A set of experiments was performed to:

1. Measure the efficiency of MSIM as a compressor.
2. Set a confidence level for the expected lower bound of compression achieved with MSIM.
3. Measure the efficiency of MSIM as a tool to discover the metasymbols that allow us to re-express a message in the most compact way.
4. Measure the efficiency of MSIM as tool for discovering metasymbols in spite of the content of the messages.

To this effect we built a set messages that contain metasymbols. The algorithm to generate the set is:

1. Select the length of the message $|m|$ with a value in the interval [a, b], Set i=0
2. **While** there are non-used indices in *m* and for a maximum number of iterations
3. i=i+1
4. Generate a random value for $|M_i|$ in the interval $[1, |m|]$
5. Propose $|M_i|$-1 gaps for the symbols of the metasymbol that is being generated
6. Generate a random value for f_i in the interval $[1, |m|]$
7. For j=1 to f_i
   8. Generate a position p_i for a instance of the current metasymbol
   9. Try to put the instance of the metasymbol at p_i whenever it does not overlap with another metasymbol.
   10. If the number of tries is exceeded do f_i= number of successful outcomes and go to step 2

   next j

Wend

11. If there remain free indices in *m*, group the symbols in the filler
12. i=i+1
13. M expresses the final number of metasymbols in *m*.
14. Store in a database the properties of all the M metasymbols.

Note that this algorithm simply determines the structure of the metasymbols but does not assign them contents; contents are determined later in the process.

We define a compressor called REF that represents our theoretical limit of compression. It takes the information stored in the database and, from ec. 2, determines the number of bits (K) in the corresponding compressed message. Now, we can measure the efficiency of MSIM relative to REF as long as MSIM yields the compression of REF. If MSIM reaches a performance similar to the hypothetical one of REF we rest assured that:

MSIM is able to reach the "theoretical" limit of compression (goal 1).
MSIM is able to identify the metasymbols hidden in *m* (goal 2).

We compared the compression achieved by MSIM against the compression of other 4 compressors: LZ77, LZW, Arithmetic and PPM.

Our third goal reflects our interest on the search for structural patterns which are independent of the content of the message. Hence, we generate messages with the same metasymbolic structure but different contents. First, we store the properties of the sets of metasymbols (order, positions and gaps) then, for a given set of metasymbols, we explore 5 variations of the same message by filling the metasymbols with contents determined from the experimental PDF from a) Spanish text file, b) English text file, c) Audio compressed file, d) Image compressed file and e) Uniformly distributed data. Because the REF compressor is fully based on the structure of the metasymbols and is impervious to its contents it was used to measure the performance of MSIM as a metasymbolic identification tool.

We want to estimate the worst case compression value achieved for each of the compressors and for each of the 5 distributions. We need to determine the values of the mean (μ) and the standard deviation (σ) of unknown distributions of the compression rates for the messages (call them distributions "A"). We will appeal to the Central Limit Theorem [10] by generating the corresponding distributions where each mean is the average of n objects from distributions A (call them distributions "B"). If X represents an object from one distribution A for a given type of file (Spanish, for example) then

$$\overline{X}_i = \frac{\sum_{i=1}^{n} X_i}{n} \quad (5)$$

represents an object of the corresponding distribution B.

The mean $\mu_{\overline{X}}$ for any of the B distributions is given by:

$$\mu_{\overline{X}} = \frac{\sum_{i=1}^{Z} \overline{X}_i}{Z} \quad (6)$$

Where Z is the number of objects of the B distributions.

The standard deviation $\sigma_{\overline{X}}$ for distribution B is given by:

$$\sigma_{\overline{X}} = \sqrt{\frac{\sum_{i=1}^{Z} (\overline{X}_i - \mu_{\overline{X}})^2}{Z}} \quad (7)$$

We generate as many messages (objects of A distributions) as were necessary until the next two conditions were fulfilled:

1. For each distribution A we require that there are at least 16.6% of Z samples in each sextil in the associated distribution B.
2. The value of the means of both distributions A and B are similar enough

$$\frac{(\mu - \sigma_{\overline{X}})}{\mu_{\overline{X}}} > 0.95$$

In our experiments we needed 2,100 for each of the different distributions: i.e. we analyzed a total of 10,500 messages.

Then we may estimate the values of the mean μ and the standard deviation σ for the A distributions since:

$$\mu = \mu_{\overline{X}} \quad \mathbf{(8)}$$

$$\sigma = \sqrt{n}\sigma_{\overline{X}} \quad \mathbf{(9)}$$

4 Results

As we see in table 1, MSIM provides consistent values of compression (see μ and σ) for the 5 explored types of file and, as expected, regardless of the contents of the metasymbols.

Table 1. Values for the parameters μ and σ of the distributions A for different compressors

	Message	English	JPG	MP3	Spanish	Uniform	Average
Compressor							
REF	μ	1.92	1.92	1.92	1.92	1.92	1.92
	σ	0.7	0.7	0.7	0.7	0.7	0.7
MSIM	μ	1.58	1.66	1.66	1.58	1.66	1.62
	σ	0.35	0.4	0.4	0.35	0.4	0.35
PPM	μ	1.66	1.21	1.21	1.71	1.21	1.4
	σ	0.9	0.75	0.75	0.9	0.75	0.8
LZW	μ	1.33	1.05	1.05	1.35	1.05	1.16
	σ	0.7	0.75	0.8	0.7	0.75	0.7
LZ77	μ	1	0.94	0.94	0.99	0.94	0.96
	σ	0.35	0.35	0.35	0.35	0.35	0.35
ARIT	μ	1.55	1.25	1.25	1.57	1.25	1.37
	σ	0.6	0.35	0.35	0.6	0.35	0.45

Once we have estimated the values of the parameters for the five distributions A, we estimate the value of the compression for the worst case X_{worst} of the different compressors appealing to the Chebyshev's Theorem

$$P(\mu_X - k\sigma \leq X \leq \mu_X + k\,\sigma) \;\geq 1 - \frac{1}{k^2} \quad \mathbf{(10)}$$

where *k* is the number of standard deviations. Assuming that the distributions A are symmetric we find, from ec. 10 that

$$P(\mu_X - k\sigma \leq X) \geq 1 - \frac{1}{2k^2} \quad \mathbf{(11)}$$

and then is possible to estimate the expected compression value X_{worst} for all distributions. It immediately follows that $P(\mu - k\sigma \leq X) \geq 0.7 \rightarrow k = 1.3$. Table 2 shows the experimental results.

Table 2. Values of the worst case compression rates

Compressor	Message	English	JPG	MP3	Spanish	Uniform	Average
REF	Xworst	1.01	1.01	1.01	1.01	1.01	1.01
MSIM	Xworst	1.12	1.14	1.14	1.12	1.14	1.13
PPM	Xworst	0.49	0.23	0.23	0.54	0.23	0.34
LZW	Xworst	0.42	0.07	0.01	0.44	0.07	0.20
LZ77	Xworst	0.54	0.48	0.48	0.53	0.48	0.50
Arithmetic	Xworst	0.77	0.79	0.79	0.79	0.79	0.78

From table 2 we know that in 70% of the cases MSIM achieves better performance than the other compressors and that its behavior is not a function of the contents of the messages.

Finally, we analyze the likelihood between the distributions B of REF and the distributions B of MSIM. Remember that REF does not depend of the metasymbol contents so its five B distributions are identical. In table 3 we can appreciate how MSIM distributions display similar behavior as REF distributions. A direct conclusion is that MSIM also does not depend of the contents. A goodness test of fit was used to test this hypothesis. We can see, from table 3, that the χ^2 values for MSIM are small enough. In contrast see, for example, the cases of PPM ($\chi^2 > 22$), JPG, MP3 and Uniform. They reveal strong dependency between their performance and the content of the messages. Other compressors (LZW, LZ77, Arithmetic, Huffman, etc.) behave similarly but we are unable to show the corresponding numbers for lack of space.

Table 3. χ^2 values for 35 distributions B

Compressor	Message	Sextil1	Sextil2	Sextil3	Sextil4	Sextil5	Sextil6	χ^2
REF	English	13	10	12	16	23	10	0.00
	JPG	13	10	12	16	23	10	0.00
	MP3	13	10	12	16	23	10	0.00
	Spanish	13	10	12	16	23	10	0.00
	Uniform	13	10	12	16	23	10	0.00
MSIM	English	12	9	16	19	14	14	7.19
	JPG	10	12	11	20	18	13	4.16
	MP3	10	11	12	20	18	13	3.78
	Spanish	12	8	18	18	14	14	8.85
	Uniform	10	14	12	15	19	14	4.65
PPM	English	12	12	16	16	15	13	5.49

	JPG	11	20	18	14	9	12	22.48
	MP3	10	21	18	14	9	12	24.96
	Spanish	13	9	16	19	14	13	6.42
	Uniform	11	20	18	14	9	12	22.48

5 Conclusions

The performance of MSIM has been compared with other compressors by compressing large sets of files that contain the same structure but different contents. Statistical validation allows us to obtain lower bounds on the compression ratio of the acknowledged "best" compressors and MSIM stands out in all cases. The application of MSIM, however, relies on relatively large execution times and, until we are able to identify a faster algorithm, is more suitable for archival purposes, where the emphasis is on optimal compression and multiple accesses rather than on real time applications.

On the other hand, the principle of minimum description length, which underlies MSIM, translates in an unsupervised pattern recognition method where deep similarities between apparently dissimilar sets of data are likely to be found. At present this characteristic is being applied to the discovery of functional relationships in biological genomic sequences.

References

1. Nelson, M., Gailly, J. L., *The Data Compression Book*, Second Edition, M&T Books Redwood City, CA (1995).
2. Huffman, D. A., "A method for the construction of minimum-redundancy codes", *Proc. Inst. Radio Eng.* 40, 9 (.), 1098–1101, Sept 1952.
3. Witten, I. H., R. Neal, and J. G. Cleary. 1987. "Arithmetic coding for data compression", *Communications of the ACM* 30(6): 520-540.
4. Ziv, J., and Lempel, A., "A Universal Algorithm for Sequential Data Compression", *IEEE Trans. on Inf. Theory* IT-23, 3 (May 1977), 337-343.
5. Burrows, M., and Wheeler, D. J., "A block-sorting lossless data compression algorithm", *Digital Syst. Res. Ctr.*, Palo Alto, CA, Tech. Rep. SRC 124, May 1994.
6. Cleary, J. G. and Witten, I. H., "Data compression using adaptive coding and partial string matching", *IEEE Transactions on Communications*, Vol. 32, No. 4, 396-402, April 1984.
7. Kuri, A. and Galaviz, J., "Pattern-based data compression", *Lecture Notes in Artificial Intelligence* LNAI 2972, 2004, pp. 1-10.
8. Kuri, A., "Solution of Simultaneous Non-Linear Equations using Genetic Algorithms", *WSEAS Transactions on Systems*, Issue 1, Vol. 2, 2003, pp. 44-51.
9. Goldberg, D., *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison Wesley Publishing, 1989.
10. Feller, W., An Introduction to Probability Theory and Its Applications, pp. 233-234, John Wiley, 2nd. Edition, 1968.

Practical Estimation of Kolmogorov Complexity using Highly Efficient Compression Algorithms

Angel Kuri-Morales[1], Oscar Herrera[2], José Galaviz[3], Martha Ortiz-Posadas[4]

[1]Departamento de Computación. Instituto Tecnológico Autónomo de México
[2]Centro de Investigación en Computación. IPN
[3]Departamento de Matemáticas, Facultad de Ciencias. UNAM
[4]Laboratorio de Informática Médica. Departamento de Ingeniería Eléctrica.
Universidad Autónoma Metropolitana Iztapalapa.
akuri@itam.mx
jose@pateame.fciencias.unam.mx heoscar@yahoo.com posa@xanum.uam.mx

Abstract. In this paper we describe a heuristic approach to the problem of calculating the algorithmic information of message *m* by estimating Kolmogorov complexity. This is achieved by defining a basic element of information which we call a metasymbol. We show that when *m* is expressed as an ensemble (which we call M) of metasymbols it complies with the minimum message length principle. Therefore, M is the shortest way of encoding *m* and approaches Kolmogorov's bound. We discuss approaches to the problem of finding the best set of metasymbols using AI techniques.

1 Introduction

Information, in the classical theory, is defined as a function of the probability of appearance of a symbol S_i at one end of a communication channel. If the S_i are statistically independent, the information contained in a message is equal to the sum of the information of every one of its symbols, which is intuitively satisfactory. However, in spite of its simplicity, we must notice that we ignore the *meaning* of a message and focus *only* in communicating it between a *transmitter* (tx) and a *receiver* (rx) under the assumption that the universe of possible messages is known both to tx and rx. This notion of information is a measure of one's freedom when selecting a message. Given the choice of sending the full contents of this paper or the phrase "Let's eat" the information concerned is, precisely, one bit.

Another intuitive way to look at the problem is to consider that the information in a finite string is the number of bits of the shortest self-contained program that computes the string and terminates. Thus, a long sequence of *n* 0's (for *n*, say, equal to 10,000) contains little information because a program of about $\log_2(n)$ bits outputs it. Furthermore, it can be shown that all reasonable choices of programming languages lead to quantification of the amount of absolute information in individual objects that is invariant up to an additive constant. We call this quantity the *Kolmogorov complexity* (K-complexity) of the object. If an object contains regularities, then it has a shorter description than itself. We call such object *compressible*. A consequence of

© A. Gelbukh, R. Monroy. (Eds.)
Advances in Artificial Intelligence Theory
Research on Computing Science 16, 2005, pp. 193-202

this is that a truly random sequence cannot be expressed in a shorter string than the enumeration of its constituents and, conversely, that a string compressed to its utmost is, for all practical purposes, random and that its length is a strict measure of its K-complexity. This second fact leads us to conclude that an algorithm achieving highly efficient compression will allow us to measure the K-complexity of an object; a fact which has been overlooked in the past.

The crux of the matter lies in defining "highly efficient compression" (HEC). In what follows we discuss a representation which provides an operative definition of HEC. It relies on the discovery of the so-called metasymbols and their proper selection. We have already proven that both problems are NP-complete and, therefore, that their practical solution must depend on heuristics. We discuss a set of algorithms which approach theoretical bounds and, therefore, offer a practical tool for the calculation of K-complexity. In part 2 we discuss the representation, in part 3 we discuss the algorithms, in part 4 we offer our conclusions.

2 Representation

Our method relies on the analysis of *m* in order to identify a set of the underlying patterns (which we call *metasymbols*). In what follows we describe one possible representation. In order to do this we must clearly define a metasymbol and the associated concepts.

2.1 Definitions

Metasymbol (Ms). It is a collection of symbols which appears embedded in *m*. Its symbols are not necessarily contiguous (they may show gaps). Its size (number of symbols) is, in general, different from the sizes of other Ms. In general, we expect that Ms_i appears more than once in *m*. The *i*-th copy of a Ms in *m* is called its *i*-th *instance*.

Position of Ms_i. It is defined by the i-th offset (the number of symbols intervening between the first symbol of Ms_i and the first symbol of Ms_{i+1}). By convention the offset of the Ms_1 is relative to the first symbol found in *m*. The position of all the symbols in Ms_i is completely specified by these offsets. Since the structure of the Ms is fixed, when we specify the offset of Ms we implicitly specify the position of all of its symbols.

Gap. It is the number of unspecified symbols between two contiguous symbols in a Ms. There may be S symbols (not necessarily different) in Ms*i*; hence, there are S-1 gaps in it. By the *structure* of a Ms we mean the enumeration of its gaps.

Content of a metasymbol. It is the enumeration of the values of all the symbols in the Ms. In general, *m* is not fully covered by all instances of the different Ms, i.e. there are symbols which are not accounted for by a collection of Ms. The enumeration of those symbols not covered in *m* is called the *filler*.

2.2 The Method

In general, the steps to express *m* as a collection of Ms [1] are: 1) Find an adequate set of Ms, 2) Express *m* as a sequence of Ms, 3) Describe the position, structure and contents of each of the Ms, 4) Describe the contents of the filler.

In what follows we illustrate the representation method with a hypothetical *m* arranged in a 16X8 matrix (see figure 1). In *m* we may identify 5 metasymbols. The first metasymbol (Ms_1) consists of 12 symbols (see figure 2), and it appears 3 times. The positions of its first symbol in the first, second and third instances are (1,1); (7,3) and (9,4). To identify the position of the next instances of the Ms_1 refer to figure 1.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	
M	S	M	Q	E	T	Q	V	Q	H	F	R	K	D	R	Q	1
A	R	R	A	E	R	A	Q	A	K	D	N	R	F	Q	A	2
H	Q	Q	T	I	O	M	R	D	M	E	K	O	V	K	K	3
Q	Y	K	D	H	H	D	Q	M	K	E	S	E	T	H	V	4
D	N	I	K	K	D	D	K	H	A	I	E	E	D	Q	M	5
D	K	D	N	R	R	S	Y	K	S	H	I	I	R	H	L	6
Q	M	H	T	Q	O	H	Q	Q	Y	M	O	H	Q	D	Q	7
A	K	K	E	K	T	H	G	A	I	K	G	A	L	K	A	8

Fig. 1. Message arranged in a 16X8 matrix

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	
M				E			V						D			1
				E						D	N					2
				I					M							3
	Y			H									T			4
																5
																6
																7
																8

Fig. 2. First instance of Ms_1

Ms_2 consists of 8 symbols and it appears 2 times. Figure 3 shows its first instance with initial position at (1,3). The second instance appears at (15,4).

Ms_3 consists of 3 symbols and it appears 9 times. Its first instance is shown in figure 4. Observe that its initial position is (15,1). For instances 2 to 9 their initial positions are (2,2), (3,2), (6,2), (13,2), (8,3), (5,6), (6,6) and (14,6) respectively (refer to figure 1).

Ms_4 consists of 2 symbols and it appears 7 times. In figure 4 is shown its first instance and its position at (4,1). For instances 2 to 7 its positions are (7,1), (9,1), (16,1), (1,7), (9,7) and (16,7) respectively (refer to figure 1).

Ms_5 consists of 3 symbols and it appears 2 times. Figure 4 shows its first instance with positions at (15,6) and (3,7).

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	
																1
																2
H								D								3
Q											S					4
								H								5
D											I				L	6
																7
																8

Fig. 3. First instance of Ms_2

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	
			Q											R		1
			A											Q		2
											K					3
																4
																5
														H		6
							Q									7
							G									8

Fig. 4. First instances of Ms_3, Ms_4 and Ms_5

The symbols of all instances of Ms_i, i=1, …, 5 account for 99 of the 128 symbols in *m* (77.34%). The rest of the symbols follow no pattern and make up the filler.

For clarity, we assign the Greek letters α, β, γ, δ and ε to Ms_1, ..., Ms_5 respectively; then $m = \alpha\,\delta\,\delta\,\gamma\,\delta\,\gamma\,\gamma\,\gamma\,\gamma\,\beta\,\alpha\,\gamma\,\alpha\,\beta\,\gamma\,\gamma\,\gamma\,\varepsilon\,\delta\,\varepsilon\,\delta\,\delta$. However, without an explicit definition of the position, structure and contents of the various Ms the metasymbolic expression of the message is meaningless. Therefore, we proceed as follows.

1) We describe the positions of the instances of the metasymbols. In the message this corresponds to the sequence: 0 2 2 1 5 0 1 0 2 6 3 5 0 16 5 21 0 7 0 1 1 5 6. Every number in the sequence represents the distance between the *i*-th and (*i+1*)-th Ms. In *m*, Ms_1 starts on the position of the first symbol (1,1); then the first instance of Ms_4 is two symbols away from Ms_1. The second instance of Ms_4 appears 2 symbols away from its first instance. Likewise, the first instance of Ms_3 is 5 symbols away from the second instance of Ms_4, and so on.

2) We define the structure of all the Ms as a sequence of gaps, one for every Ms. Since the symbols are unknown, code '0' is reserved to indicate the end of the structure of a Ms. The structure of Ms_i is, thus, a collection of gaps plus a '**0**'. Furthermore, since it is possible to have a gap of size 0, every gap is encoded as its actual size +1. As an example, the structure of Ms_1 is specified by the sequence: 4 3 6 7 6 1 9 5 8 3 9 **0**. The gap between S_1 and S_2 is 3 (4-1), the gap between symbols S_2 and S_3 is 2 (3-1), and so on.

3) We proceed to define the contents of every metasymbol. The contents of the Ms in *m* are defined as follows. Ms_1: {MEVDEDNIMYHT}; Ms_2: {HDQSHDIL}; Ms_3:{RQK}; Ms_4:{QA}; Ms_5 :{HQG}.

At this point we have defined all necessary elements: order, position, structure and contents, for every Ms in *m*. However, as stated, there are symbols which are not accounted for by the collection of Ms defined. However, the exact positions of all undefined symbols in *m* are, at this point, known.

4) We enumerate the contents of undefined localities to complete the cover. The following sequence is the filler of *m*:{SMTHFRKAQKFTHDIKKAEDKSKMEKHAK}.

Under this representation *m*'s K-complexity (K) is given by

$$K = \mu(1 + \sum N_i) + \omega \sum N_i + (\gamma + \lambda)\sum S_i + L - \lambda \sum S_i N_i$$

Where N_i denotes the number of instances of Ms_i; M = number of different Ms; $\mu = \lceil \log_2(M) \rceil$; $\omega = \lceil \log_2[\text{max. offset } Ms_i] \rceil$; $\gamma = \lceil \log_2[\text{max. gap } Ms_i] \rceil$; $\lambda = \lceil \log_2 |Ms_i| \rceil$; $S_i = \lceil \log_2(\text{max. symbols } Ms_i) \rceil$ and $L = |m|$ in bits.

3 Algorithms

In the preceding discussion we assumed that the Ms in *m* are known. As pointed out in the introduction, such is not the case in general. In fact, the problem of finding the Ms is NP-hard. In what follows we discuss two approaches to the solution. In 3.1 we focus on an approach where Ms are determined on-line; in 3.2 we describe an alternative where these are determined off-line. We have conducted statistical tests which allow us to affirm that both methods approach the Ms set which leads to maximum compression.

3.1 On-Line Search

MSIM is a tool for searching metasymbols based on Vasconcelos' Genetic Algorithm (VGA)[2]. The VGA uses a population *P* of *N* individuals, a temporary population of 2*N* individuals, deterministic selection (from the best *N* individuals) and has two main operators: mutation and crossover. Assuming a binary coding of the individuals, mutation replaces a bit's value by its complement with probability *Pm*. Crossover requires two individuals and interchanges their genetic material with probability *Pc*.

The off-line algorithm (MSIM) models *m* as an array of symbols indexed from 0 to |*m*|-1. The individuals of MSIM are permutations of all the |*m*| indices of the message which lie in the interval [0, |*m*|-1]. Each index is coded in traditional weigthed binary and requires $\lceil \log_2 L \rceil$ bits. We delimit the metasymbols of an individual by reading the indices from left to right and looking for an ascending order.

As an example, in figure 5 we illustrate a possible individual for the hyptothetical *m* "xyazbcdxyez".

0 1 3	2 4 5 6 9	7 8 10
x y z	a b c d e	x y z
Ms1	Ms2	Ms3

Fig. 5. An individual of VGA in MSIM

The relation between the first occurrence of a metasymbol and its repeated instances is obtained by comparing the contents of the metasymbols as well as their gaps. In the example of figure 1, Ms 3 is a repeated instance of Ms 1 because they match their contents as well as their gaps. The other symbols make up the filler. The positions of all the metasymbols can be determined from the positions of the first symbol of each metasymbol. In genetic algorithm's terminology, the measure of fitness for MSIM is *K* which is to be minimized.

VGA's mutation and crossover operations are not directly applicable in manipulating the individuals of MSIM because they act on the genotype (binary encoding) which may lead to invalid individuals (out of the feasible region), i.e., individuals with repeated indices or with indices out of the interval [0, |*m*|-1]. For this reason we redefined the mutation and crossover operators to render them appropriate.

The mutation operator consists of a permutation of two indices whereas the crossover corresponds to PMX [3]. Both of them retain the resulting individuals in the feasible region. Figure 6 illustrates this two operators.

Mutation

A = 0 1 3 2 4 5 6 9 7 8 10

A'= 0 1 3 9 4 5 6 2 7 8 10

Crossover

A= 9 8 4 | 5 6 7 | 1 3 2 10
B= 8 7 1 | 2 3 10 | 9 5 4 6

A' = 9 8 4 | 2 3 10 | 1 6 5 7
B' = 8 10 1 | 5 6 7 | 9 2 4 3

Fig. 6. Mutation and crossover operator for MSIM

MSIM's representation is slightly different to the one described in section 2.2; it encodes the position between the i-th and the j-th symbol corresponding to the next repeated instance of the i-th metasymbol. Here we do not need to explicitly express the sequences of the metasymbols since we can group the positions of the α's, β's, γ's, δ's and ε's, as shown in figure 7.

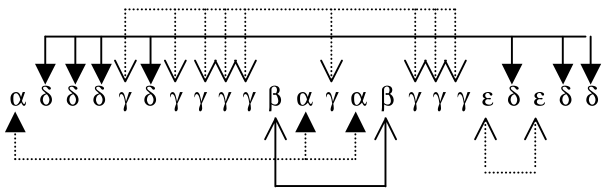


Fig. 7. Alternative coding of m

Hence, instead of the sequence of metasymbols we can indicate their positions as follows:

α: 0, 11, 2; β: 10, 4; γ: 4, 2, 1, 1, 1, 3, 3, 1, 1; δ: 1, 1, 1, 2, 14, 2,1, 1; ε: 18, 2

The number of bits of the re-expressed *m* can be calculated by grouping the bits required for each metasymbol, thus

$$K' = \sum \left[\psi_i N_i + (\gamma + \lambda) S_i\right] + L - \lambda \sum S_i M_i$$

where ψ_i is the number of bits required to encode the positions of all the instances of the i-th Ms. The term of the first summation represents the number of bits required to encode the i-th metasymbol and is denoted by A_i, then

$$A_i = \psi N_i + (\gamma + \lambda) S_i$$

The number of bits used by the symbols of the i-th Ms in *m* is B_i and is given by

$$B_i = |M_i| f_i$$

Notice that, although we have derived two different expressions (K and K') for Kolmogorov complexity we have shown that, in practical applications, both are equivalent [4]. In fact, there is an arbitrary number of possible representations and they are all computationally equivalent. The latter is interesting in terms of MSIM's implementation.

We now define the discriminant d_i as

$$d_i = \frac{A_i}{B_i}$$

Any value of d_i smaller than 1 implies that we may achieve *m*'s compression and it does not depend on the rest of the Ms. Therefore, we can iteratively search for metasymbols whenever we are able to find an associated d_i smaller than 1. Of course, we are looking for Ms such that they minimize d_i. The Ms in the set are encoded as individuals of MSIM and introduced to the population via a special operator called the *catastrophe operator*. This operator corresponds, roughly, to forcing a large number of mutations on the individuals, hence its name. It relies on the iterative search for Ms with minimum d_i value and it will never submit individuals in which of *K'* is larger than $|m|$.

3.2 Off-Line Search

In the previous section we have introduced a method that attempts to find an optimal subset of patterns as the patterns are found. This on-line approach has the advantage that the time used to determine a good subset of patterns, is amortized by the time consumed in the pattern discovery procedure. But it has the disadvantage that some good patterns can be disregarded. If, at some stage of the algorithm, the frequency and *order* (number of defined symbols) of two patterns are alike then one of these will be selected by the algorithm, while the other is dropped even if its potentiality could emerge later in the process.

An alternative approach is to generate a set of frequent patterns without any evaluation of its goodness, and then proceed to determine the best subset of patterns to build the *a posteriori* dictionary. Evidently this approach is more expensive than the previous one, but the risk that higher performance patterns will be ignored is reduced.

The process implied in the implementation of this approach is divided in two phases:

1. The search for frequent patterns. The result obtained at the end of this phase is a set *P* of frequent patterns.
2. The search for a subset $B \subseteq P$ (*B* is assumed as the dictionary or *set of metasymbols*) such that the amount of bits required for the re-expression of the *m* together with the dictionary is minimum.

Given the kind of patterns we are looking for in *m*, the problem related to phase 1 is very complex. All the algorithms reported have an exponential complexity on $|m|$ [5]. However if some restrictions are imposed on the frequency and order of the patterns, there exist algorithms with better performance [6].

Concerning phase 2, we have proved that the search for an optimum subset of patterns if it is adopted as the dictionary, is an NP-complete problem [7], therefore heuristics are needed in order to approximate such subset.

The optimality of some pattern is given by its frequency and its order: the higher the order and frequency, the better is the pattern. But there is a trade-off between these two quantities. In one end we have that single symbols are patterns with highest frequency and minimum order; on the other we have that the pattern encompassing *m* has maximum order and minimum frequency. Hence, we need to find patterns which maximize both quantities simultaneously. Therefore, we require a measure for pattern quality including frequency and order.

The cost to include some pattern in a dictionary is better amortized if such pattern appears frequently in *m* and there is a large amount of symbols covered by all these appearances. Therefore, the product of pattern frequency and pattern order, (which we call the pattern *coverage*), can roughly provide us with an estimate the pattern contribution. However, it may happen that some symbols covered in one occurrence of a pattern overlap with those covered in another. The amount of symbols covered at least one time for a pattern is called the *effective coverage*, which is a more precise measure for the pattern's quality. The concept of effective coverage, can be generalized in the following way: given a set of patterns *B*, and a pattern *p*, the *exclusive coverage* of *p* is the number of symbols in the sample covered at least one time by all the occurrences of *p*, and not already covered by some other pattern in *B*. This is the final criterion we use to evaluate the patterns.

The second phase of the procedure sketched above is, therefore, accomplished by a heuristic based on the concept of exclusive coverage (which is why we have called it the *coverage-based heuristic*). The output of this algorithm may be considered as a proposed subset that could be further optimized by using some local optimization procedure. This is attempted by two local optimizers which are, actually, two different kinds of hill climbers that we will describe below.

The coverage-based heuristic proceeds as follows:

- **Input:** A set *P* of frequent patterns found in a sample *S*.
- **Output:** A subset $D \subseteq P$, assumed to be an approximation to the metasymbol dictionary.

1. Let $D = \emptyset$.
2. While the total coverage of patterns in *D* is less than $|m|$ and *P* has patterns with exclusive coverage, respect to *D*, greater than zero.
   2.1. Chose the pattern *p* in *P* with the highest exclusive coverage respect to *D*.
   2.2. Add *p* to *D*, and remove *p* from *P*.
3. Return *D* as the dictionary.

The resulting dictionary can now be optimized by local search procedures. We have defined two of these mechanisms which are applied consecutively.

- Minimum Step Hill Climber (MSHC). Given the code of the subset of patterns (a binary vector of size $|P|$, where 1 in the *i*-th position means that the *i*-th pattern of *P* is included in the subset), the hill climber proceed to flip every bit in the vector. Every change is evaluated and, if the size obtained with the change improves the previous size, the changed code is the current subset.
- Minimum Replacement Hill Climber (MRHC). MRHC proceeds to exchange systematically, the position of every bit with value 1 with every other position whose value is 0.

Both of these climbers can be executed until no improvement is found or a fixed number of evaluations have been performed.

4 Conclusions

We have shown that the best metasymbolic re-expression and further post-encoding of message *m* yields the most compact representation of *m*. This is true provided that we are able to find an adequate set of metasymbols in *m*. Furthermore, since this process is lossless, the original information in *m* remains unaltered. The most economic representation of *m* is possible only when enough information is provided in order for the decoder to infer the structure, position and contents of the metasymbols. A formalization of these facts may be found in the *Invariance Theorem* which embodies (most of) the theoretical foundation of K-complexity. We now state it without proof (but see [8]).

There is a universal partial recursive function (prf) φ_0 *for the class of prf's to compute x given y. Formally this says that* $C_{\varphi_0}(x|y) \leq C_{\varphi}(x|y) + c_{\varphi}$ *for all prf's* φ *and all x and y, where* c_{φ} *is a constant depending on* φ *but not on x or y.*

For the purposes of this paper this theorem says that we will be able to decode message x from its metasymbolic representation y and that φ_0 is the prf (embodying our method of representation) allowing us to do so. Usually K-complexity is defined in terms of a Universal Turing Machine and its calculation is, in practice, very difficult at best. Our representation and the related algorithms are a practical approach which simply consists of finding those sets which minimize c_{φ} and, therefore, correspond to the smallest possible constant involved.

As discussed above, the computational cost of achieving such goal is justifiable as long as high order compression methods (other than ours) are not found.

References

1. Kuri A, "Pattern based lossless data compression", WSEAS Transactions on Communications, Issue 1, Vol. 3, 2004, pp. 22-29.
2. Kuri A, "Solution of Simultaneous Non-Linear Equations using Genetic Algorithms", *WSEAS Transactions on Systems*, Issue 1, Vol. 2, 2003, pp. 44-51.
3. Goldberg, D., (1989), *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley Publishing Company.
4. Kuri A and Herrera O, "Efficient lossless data compression for nonergodic sources using advanced search operators and genetic algorithms",*Advances in Artificial Intelligence, Computing Science and Computer Engineering*, Nazuno J, Gelbukh A, Yañez C, et. al. (editors), ISBN: 970-36-0194-4, ISSN: 1665-9899, Vol. 10, 2004, pp. 243-251.
5. Vilo, J., *Pattern Discovery from Biosequences*, PhD Thesis, Technical Report A-2002-3, Department of Computer Science, University of Helsinki, 2002.
6. Rigoutsos, I. and A. Floratos, "Combinatorial Pattern Discovery in biological sequences: The Teiresias Algorithm", *Bioinformatics*, Vol. 14, No. 1, 1998, pp. 55-67.
7. Kuri, A. and J. Galaviz, "Pattern-based Data Compression", in *MICAI 2004: Advances in Artificial Intelligence*, Springer Verlag, Lecture Notes in Artificial Intelligence 2972, 2004, pp. 1-10.
8. Li M and Vitányi P, *An introduction to Kolmogorov complexity and its applications*, Springer Verlag, New York, 2nd Ed, 1997.

Robotics

Steering Control of an Ackerman Mobile Robot Using Fuzzy Logic

E. Alarcón-Avila, S.A. Martínez-Cuevas, A. Rangel-Huerta

Facultad de Ciencias de la Computación, Postgrado BUAP
14 Sur y Av. San Claudio; Edificio 135, C.U. San Manuel,
Puebla, Puebla, México. C.P. 72570
arangel@cs.buap.mx, henry0227@yahoo.com

Abstract. In this study, we implement a rule based on fuzzy logic algorithms, on a micro controller PIC16F84, to control the steering of an Ackerman mobile robot. The robot has three frontal infrared sensors that send a set of signals through a block of rules to determine the steering velocity. The task of the robot is to follow a moving object of reference. The implementation is realized in Language C, to make the structure of the programs easier.

1 Introduction

In this work we implement a modern control algorithm based on fuzzy logic rules to control the steering motion of an Ackerman mobile robot that we made[1]. The robot has three front-placed infrared sensors that send information to the robot to generate the direction of the navigation in an unfamiliar environment.

This investigation is divided into two parts: First, the design, construction and proofs of our mobile robot of type Ackerman, based on a micro controller PIC 16F84, the mechanical chassis, the electronic design and the electronic cards, including the infrared sensors and the batteries. We make some proofs using a preliminary software downloaded into the micro controller[2-4]. These proofs have the main task of determining if the steering system, wheel traction and sensing have perform correctly.

Secondly, we have developed fuzzy control software for controlling the steering of the mobile robot. The task of the robot consist in following a moving object, that is, the sensor actives the steering direction to determine the wheels velocity necessary to follow the object without losing it.

2 The Mechanical Structure of the Robot

The mechanical structure of the mobile robot consists of a plastic base with four wheels. The pair on the back is motorized, generating the traction of the motion, and the pair in the front has the steering power. Thus, we have a wheeled mobile robot of

© A. Gelbukh, R. Monroy. (Eds.)
Advances in Artificial Intelligence Theory
Research on Computing Science 16, 2005, pp. 205-212

Ackerman configuration with three frontal infrared sensors and a board micro controller (Figure 1).

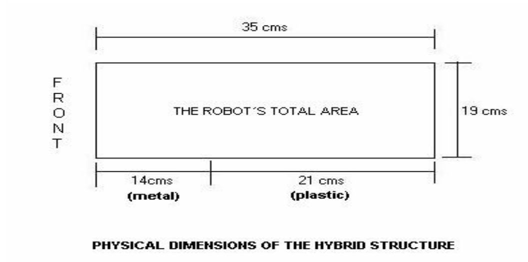


Figure 1. The space distribution of the chassis.

The robot is constructed plastic and metal components with rigid mechanisms of articulation. In this context we can propose a model for the mobility of the robot in terms of the analysis of a number of free movement unions.
A mobility definition depends on the number of independent parameters necessary to specify the position of each of the mechanism links or degrees of freedom (Figure 2).

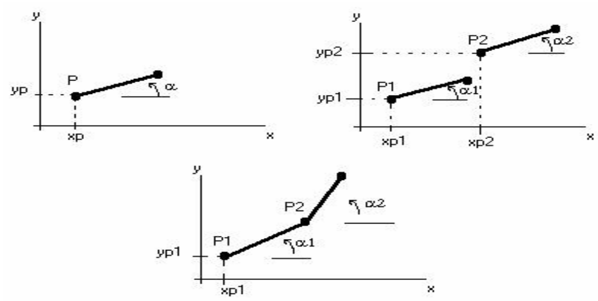


Figure 2. Degrees of Freedom of the links.

Through an analysis of the Figure 2, in Ref.[5] the equation for the mobility of our mobile robot is:

$$M = 3(n) - 2f1 - f2 \qquad (1)$$

M = mobility or degree of freedom, n = total number of arms, f1 = links with one degree of freedom, f2 = links with two degrees of freedom.
In our case we have,

$$M = 3(2) - 2(1) - 0 = 6 - 2 = 4. \qquad (2)$$

However, the corrected Eq.(1) used to determine the degree of freedom of the specific steering mechanism is,

$$M = 3(n - 1) - 2f1 - f2, \qquad (3)$$

because the mechanism is in contact with the surface.

The articulations are as follows: The power traction is an articulation of torsion, which realizes a torque stress between the input and output links. The output axis is parallel to the input ones (Figure 3).

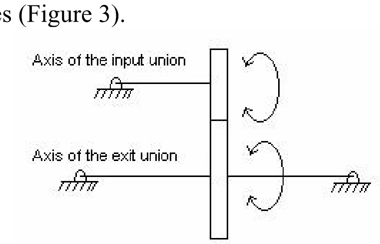


Figure 3. Relationship between input and output axis.

In our case, each rotational link has one degree of freedom (pure rotation) between the connecting links. Then we have n=2 arms, f1=1 links of a degree of freedom, f2=0, and M=1.
A revolution articulation is used for the steering mechanism. The rotational axis is parallel to the link input axis and orthogonal link output axe, i.e. the output link rotate around the input link (Figure 4).

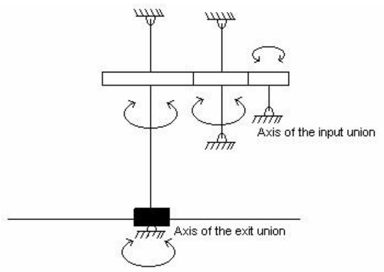


Figure 4. Revolution link has one degree of freedom.

The analysis of the steering components using the above description gives us n=4 arms, f1=4 links with one degree of freedom, f2=0 links with two degrees of freedom, M= 3(n-1)-2f1-f2=3(4-1)-2(4)-0 =1. Hence this articulation has one degree of freedom.
Based on the above description of the mechanical components of the mobile robot, we have constructed a prototype which is represented in Figure 5. An electronic card, based on the PIC-16F84[4], is used to realize the control the drive motors and infrared sensors. Because the algorithms are in Language C, an interface is required to download code to the control card.

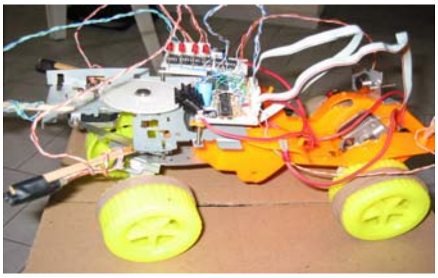


Figure 5. Mobile Robot of Ackerman configuration based on the PIC-PIC16F84.

3 Fuzzy Control of the Robot's Steering

Fuzzy logic has a variety of applications for solving engineering control problems. Here is used as a means of controlling the steering. Because the robot functions we can obtain smooth direction changes. If the steering angle is small, a robot makes a smooth direction change. If angle is large, the direction change is rapid and smooth with a deceleration as it attains the target direction.
The linguistic variables that have been proposed to control the steering of the mobile robot are shown in Table 1.

Input variables			Output variables	
Right Sensor	**Left Sensor**	**Frontal Sensor**	**Steering velocity**	**Position**
All left	All right	All right	Fast right	Right
Medium left	Medium right	Medium right	Slow right	Center
Aligned	Aligned	Aligned	Zero	Left
Medium right	Medium left	Medium left	Slow left	
All right	All left	All left	Fast left	

Table 1. Set elements of the linguistic variables.

We have three input variables and one output variable, which feedbacks into the process control as shown in Figure 6.

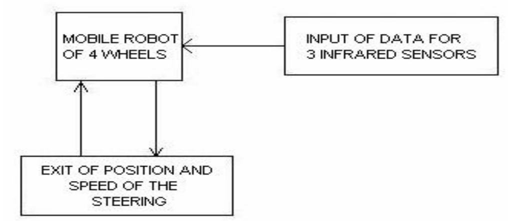


Figure 6 Block diagrams of interaction for the linguistic variables.

The ranks of the linguistic variables establish some limits present in the physical system, like maximum torsion angle and the response time of the electronic signals, as shown in Table 2. The sensor variables admit only two values because its electronic circuits do not accept analogical values.

Variable	Range		Type
Right Sensor	0	1	Integer
Left sensor	0	1	Integer
Frontal Sensor	0	1	Integer
Position	-45^o	45^o	Integer
Velocity	50mSec	100mSec	Integer

Table 2. Ranks of the linguistic variables.

The fuzzy functions are established under the desired constraints. There are four basic sets of fuzzy functions [2]: Type Z, type PI, type LAMBDA, type S, (Figure 7).

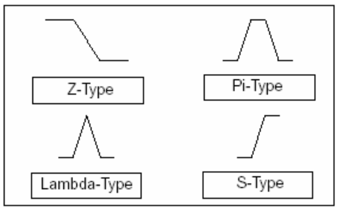


Figure 7. Four types of fuzzy functions.

In our case we utilize a *lambda* fuzzy function. This function involves the values for its "defuzzification" which must be define its degree of membership.

The fuzzy sets for the membership functions and their fuzzy accessible values are given for the same variables, as described above. We have defined the followed membership functions:

RIGHT_ SENSOR = {all_left, medium_left, aligned, medium_right, all_right}
LEFT_SENSOR = {all_right, medium_right, aligned, medium_left, all_left}
FRONT_SENSOR = {all_right, medium_right, aligned, medium_left, all_left}
VEL_POSITION = {fast_right, slow_right, zero, slow_left, fast_left}
POSITION = {left, center, right}

The membership function of fuzzyfication corresponds to VEL_POSITION. The input data sensor processed in the rules block, which provides feedback for the variable POSITION (Figure 8).

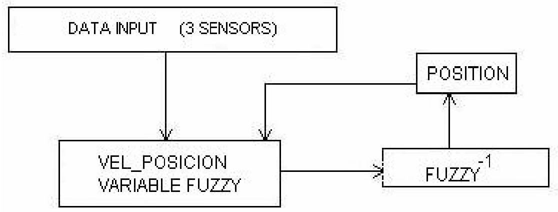


Figure 8. Flow of information of the fuzzy control as it goes through the electronic system.

4 Fuzzy Control Algorithms Programming

To implement the fuzzy control scheme, as described in the last section, we need to generate some code, that is understandable by the control electronic card, or specifically for the micro-controller.

The fuzzyfication of the set of values of the variable VEL_POSITION generate *lambda* type functions, (Figure 9).

Figure 9. Fuzzy variable for torsional velocity.

The intersections of the *lambda* functions, gives us some of the membership degrees. The method used to calculate the membership degrees is the *center of maximum* (COM). Three types of membership are defined associated to each of the linguistic variables. The first type, corresponds to the maximum degree of membership fixed at 100 percent. The second one is the membership degree that which establishes its value in agreement with the corresponding *lambda* function. The third type is associated with the maximum argument that establishes the non fuzzy self value, Table 3.

Maximum argument (M)	Fast_ right	Slow_ right	Zero	Slow_ left	Fast_ left
Membership degree (MD)	0.7	0.7	0.7	0.7	0.7
Maximum MD	1	1	1	1	1

Table 3. Degree of membership of the linguistic variable of velocity.

4.1 Center of the Maximum (CoM)

For the defuzzification, we use the method of the CoM, first of all, we need to define the value *C* = CRISP as the value of defuzzing [2-3]:

$$C = \frac{\sum i[(I)(Max(M))(\arg(\max(M)))]}{\sum i[I]} \quad (4)$$

being,

I = membership degree, *Max(M)* = maximum degree of membership, *Arg(Max(M))* = Argument of the maximum degree of membership.

As an example of defuzzification, lets consider that the fuzzy value is SLOW_LEFT, to calculate the corresponding crisp value. The arguments of the maximum degree of membership are taken as: Arg(Max(SLOW_LEFT)) = 20, Arg(Max(FAST_LEFT)) = 37, Arg(Max(ZERO)) = 0. The degree of membership: I(SLOW_LEFT) = 0.7, I(FAST_LEFT) = 0.7, I(ZERO) = 0.7. Maximum degree of membership: Max(SLOW_LEFT) = 1, Max(FAST_LEFT) = 1, Max(ZERO) = 1.
Then the maximum of the center result is: C = 19.

5 Results

5.1 Object Follower Fuzzy Steering control

After downloading the fuzzy control code into the control card of the robot, we obtain a soft regulated steering velocity. The activity of the sensors is generated by a moving object of reference. The steering of the robot is oriented through the current active sensor. When the object movement activates another sensor, the steering direction changes into the new defined direction with sudden start follow a soft velocity brake. When two sensors are active simultaneously, the intermediate current velocity is preserved as a steering position of following or returning to the former position. The magnitude of the steering velocity depends directly on the separation of the sensors. The fuzzy control for the steering of our mobile robot constitutes a powerful tool for effectively following the reference object.

5.2 Code in Language C

The code of the program is written in Language C to be compatible with the compiler PICC of HITECH Software [4]. This compiler was chosen mainly due to the resulting code structure. The flow chart showing code is presented in Figure 10.

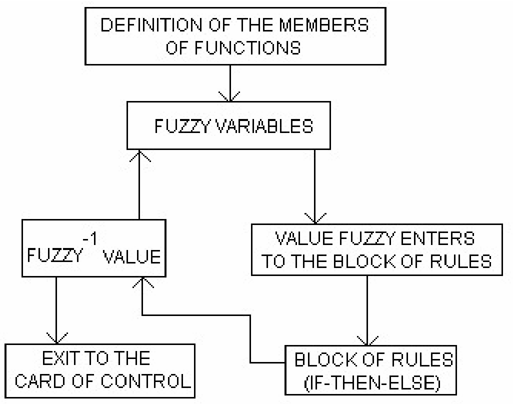


Figure 10. Fuzzy control flow chart.

6 Conclusions

It is possible to implement fuzzy control algorithms in a micro controller in the context of mobile robotics. We have used fuzzy control code to control the steering directions of a mobile robot of Ackerman configuration. We obtained good performance on the steering direction changes of the robot following a reference object, regulated by the sensor activity. The main complication in using fuzzy control algorithms in a micro controller, is that it uses a much of the program memory in the defuzzification process.

References

1. J. M. Angulo Usategui, S. R: Yesa, I. Angulo Martínez, Microbótica, Thomson Ed. (1999).
2. J.-S.R. Jang, C.-T. Sun, E. Mizutani, Neuro-Fuzzy Soft Computing, Matlab Series. 1997.
3. Robert Schreiber, Air Flow Control Using Fuzzy Logic. Microchip Technology Inc. Application Notes, 1997.
4. HI- TECH Software, PIC microcontrollers have a powerful FREE ANSI C compiler. http://www.htsoft.com
5. J. Edward Shigley, J. J. Uicker. Teoría de Máquinas y Mecanismos. Ed. Mc Graw Hill, Primera Edición, 1988.

Evolving Robot Behavior for Centralized Action Selection

Fernando M. Montes-González[1], and José Santos Reyes[2]

[1]Facultad de Física e Inteligencia Artificial, Universidad Veracruzana
Sebastián Camacho No. 5, Centro, Xalapa, Ver., México
fmontes@uv.mx
[2]Departamento de Computación, Facultad de Informática,
Universidade da Coruña, Campus de Elviña, 15071 A Coruña
santos@udc.es

Abstract. An effective central model of action selection for solving a foraging task has been presented in past papers. The main task is decomposed in behavioral modules that use multiple sensor information fused together in the form of a unified world perception. In turn, the urgency to-be-executed of each behavior is calculated from its internal status and the unified perception of the world. In this model, the behavior with the highest salience is allowed to be expressed through motor commands. However, for selection to occur we assume that behavioral modules have already been learnt; as a consequence it is necessary to have designed these modules at an earlier stage. In this paper, we propose the use of genetic algorithms to nearly optimize behaviors related to travel the arena where the robot is to be set. Furthermore, we propose that by sharing the same topology as the evolved behaviors, backpropagation can be used to train the locating cylinders behavior.

1 Introduction

The Action Selection problem is a recurrent topic in robotics. This problem finds its roots in ethology where it is termed the "behavior switching" or the "decision making" problem. A developing interest in action selection has grown in researchers looking for modeling animal behavior in robots. Therefore, either a set of tasks or a single task has been chosen for building and programming these animal-robots (*animats*). Mostly, these tasks are concerned with the foraging behavior and social behaviors such as flocking and prey-catching among others [1]. Once the main task is set for the robot to be solved, a mechanism for selection is needed. Thus, a variety of action selection mechanisms can be used, these models range from arrow-box diagrams, and Gedanke experiments [2], to complex equations for replicating animal behavior.

Whatever method is chosen, the main task has to be decomposed in algorithmic versions of behaviors that fused together solve the desired task. Next, the action selection mechanism, and the algorithmic behaviors have to be implemented using any available robotic platform. Thus, the design of the robot task using the *animat* approach meets specific needs that the roboticist has to fulfill if the task is to be re-

© A. Gelbukh, R. Monroy. (Eds.)
Advances in Artificial Intelligence Theory
Research on Computing Science 16, 2005, pp. 213-222

solved. Frequently, roboticists focus more on designing either the action selection mechanism or the behaviors that compose the main task. In this paper, we are trying to focus on both the design of the action selection mechanism and the selected behaviors. Thus, we intend to use an evolutionary approach to fine tuning some of the behaviors that the action selection mechanism has to switch. Besides, we also plan to use backpropagation to train another behavior. Finally, a model of Central Action Selection with Sensor Fusion (CASSF) is used to switch between these behaviors.

The development of the experiments below described requires some necessary background on Genetic Algorithms, which is explained in section 2. Later on, in section 3 we explain how we design the behaviors: *cylinder-seek*, *cylinder-pickup*, *wall-seek*, *wall-follow* and *wall-deposit*. Then, these behaviors will be used in conjunction to solve the foraging task set for the Khepera robot. The selection of the behaviors is made using the CASSF model, which is explained in Section 4. Next, section 5 presents the results of integrating the evolved behaviors, and the neural-net trained behavior with some algorithmic behaviors. Finally, in section 6 we provide a general discussion highlighting the importance of these experiments.

2 Evolving Robot Behavior

Several evolutionary methods have been considered by the robotics community for producing either the controllers or the behaviors required for robots to perform their assigned tasks, and survive in their environments. All of them may be taken as variations of a general process whereby each generation of individuals from a population is evaluated, and these individuals procreate according to their fitness. Later on, the population undergoes mutation processes and a new population is somehow selected from the old one to continue with the process. These methods include Genetic Algorithms (GAs), Evolutionary Strategies, Genetic and Evolutionary Programming and Co-Evolution, although in this work we have used GAs [3].

The use of evolutionary techniques for the development of robot control systems commonly relies on the use of neural networks [4]. Therefore, a population of robot controllers is encoded into genotypic representations; then the selection, crossover, and mutation operators are applied to the population in order to produce a new offspring. The decoding of every genotype into individual phenotypes (neural controllers) allows the evaluation of the new offspring by having each individual to live for a limited period of time. This process is repeated until a satisfactory evaluation level is obtained (Figure 1).

In order to find a solution within a search space the genetic operators are applied to move across a convoluted landscape (Figure 2). This landscape is the result of measuring the fitness of all the individuals of the population. In the evolutionary robotics methodology measuring the fitness corresponds to how well the robot performs when evolving a particular behavioral module. Next, the robot is allowed to operate in the environment for an adequate number of steps or lifetime. The fitness calculation implies robot time-consumption; therefore, in the majority of the cases the use of a robot simulator is preferred. In order to minimize the “reality gap” between the simulator and the actuators, the simulator has to introduce different noise-levels in the sensors and the actuators. As a result an optimal final transference of the controllers from the simulator to the real robots can be obtained [4,5].

Usually the specialized representation of the robot controller to-be-tested is coded into a chromosome, where each locus (or position) takes a finite possible value (allele). This representation corresponds to the genotype from where the phenotype is derived. Often the genotype directly codes into the phenotype; however, sometimes an elaborated translation is needed. Initially, a population of random controllers is spawned, their fitness evaluated, and then the GAs' operators are applied. The *selection* operator chooses to breed, based on the individual fitness, the best individuals in the next offspring by means of operators such as the crossover and the mutation. The selection is made in a probabilistic way; thus, the use of the weighted *roulette-wheel* is a common method of selection. In this method, each slot of the roulette corresponds to an individual in the population, and the size of this slot is proportional to its fitness.

Another way of selecting individuals is the *tournament selection* where new individuals are generated from the result of local competitions. In the competitions the fittest individuals who beat the less fitted; the former are inserted back in the new progeny whether using or not the crossover operator. The preservation of the best individuals, from a previous generation, into the next offspring is called *elitism*. Selection in this way is used to guarantee that the best solution found so far is not lost. The *crossover* is a probabilistic operator, which takes two individual encodings and swaps their contents around a random point along the chromosome (one point crossover). *Mutation* occurs with a probabilistic flip of the bits in the chromosome encoding of the individuals of the new offspring (in general, with a random change of the alleles of the genes). Often, mutating the individual contents of a new offspring causes inferior individuals; though, better individuals will occasionally emerge.

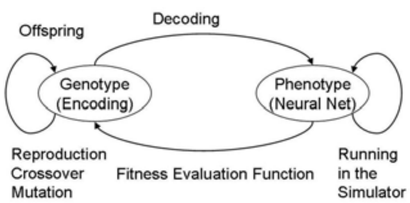


Fig. 1. The new offspring is generated from the genotype of previous robot controllers (adapted from [6])

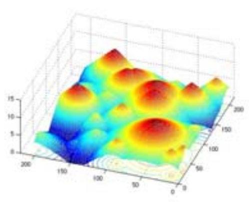


Fig. 2. A fitness landscape generated using a DF1 test problem generator [7]

3 The Design of the Behaviors

In the last decade the use of a commercial robotic platform and the use of a robot simulator have been a popular choice for researchers trying to model robot behavior. One example is the Khepera robot [8], which has been commonly used in evolutionary robotics. The Khepera is a small robot, which has a diameter-long of about 70 mm; the two DC motors control the displacement of the robot on its wheels. This robot has been equipped with a ring of eight infrared sensors all around the body of the Khepera. Despite many simulators of the Khepera been freeware over the Internet, the use of a commercial simulator has several advantages over freeware software.

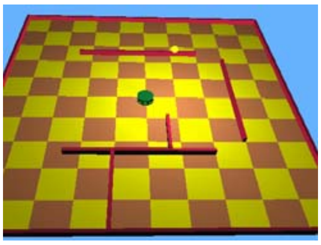


Fig. 3. The *wall-seek* and *wall-follow* behaviors were evolved using the Webots Simulator

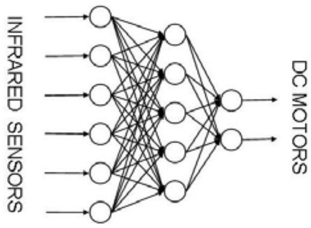


Fig. 4. The behaviors *wall-seek*, *wall-follow* and *cylinder-seek* share the same Neural Network topology

One of the main advantages of using commercial software for robot control and simulation is a complete support of turret attachments. Webots is a 3D robot simulator [9] that widely supports the Khepera robot amongst other platforms. In this work we employed the Webots simulator for developing the follow walls and avoid obstacles behavioral modules (Figure 3). The controller for these behaviors is a fully connected feedforward multilayer-perceptron neural network with no recurrent connections (Figure 4). The topology for the neural network is as follows: 6 neurons in the input layer, 5 neurons in the hidden layer, and 2 in the output layer. The sigmoid transfer function is used at the hidden and the output neurons. The infrared output from the Khepera (0 to 1023) is made binary by using a collision threshold $th_c = 750$. This binary input is fed into the neural network; in turn, the binary output of the neural network is scaled to ± 20 values for the DC motors.

The genetic algorithm nearly optimizes the weights of the neural network, and the use of a direct encoding sets the genotype as a vector **c** of 40 elements. Having a single vector representing every individual of the population, the size of the population is equivalent to all the number of neural networks (to be evaluated) in one generation. Random values are generated for the weights $\mathbf{w}_i$, $-1 < \mathbf{w}_i < 1$, of the $n=100$ neural controllers of the initial population G_0. *Elitism* is used to facilitate the copy of the two best individuals into the next offspring. Four random parents are chosen for $(n/2)-1$ local competitions for the *tournament* selection. The two winners of one local tournament are breed using a random crossover point with a probability of 0.5. On the other hand, the new offspring is affected with a mutation probability of 0.01. Individuals are allowed to run for about 22 seconds in the Webots simulator. In order to let the individuals to start from different locations, and orientations, a supervisor node was set in Webots. Communication of the supervisor with the neural controller, over TCP/IP, facilitated the motion of the Khepera in the fast-speed mode. The world used for running the evolution of both controllers is shown in Figure 3. A world such as this favors avoiding obstacles while traveling close to walls.

For the avoiding obstacles behavior (*wall-seek*), the employed fitness formula (adapted from [10]) for each individual was

$$f_{c1} = \sum^{3000} abs(ls) * (1 - \sqrt{ds}) * (1 - \max\_ir) \quad (1)$$

where *ls* is the linear speed in both wheels, *ds* is the differential speed on both wheels (a measurement of the angular speed), and *max_ir* is the highest infrared sensor value. The use of a formula fitness like this rewards those fastest individuals who travel on a straight line while avoiding obstacles. The plot of the fitness and the average fitness for this behavior, over 11 generations, is shown in Figure 5. Next, the fitness formula employed for the behavior *wall-follow* (adapted from [11]) was as follows

$$f_{c2} = f_{c1} * (tgh)^2 \quad (2)$$

In this formula the tendency to remain close to walls (*tgh*), or thigmotaxis, is calculated as the fraction of the total time an individual is close to a wall. Therefore, a fitness formula such as this selects the individuals that travel near the wall with an avoidance behavior. Figure 6 shows the plot of fitness and the average fitness for the *wall-follow* behavior over 18 generations.

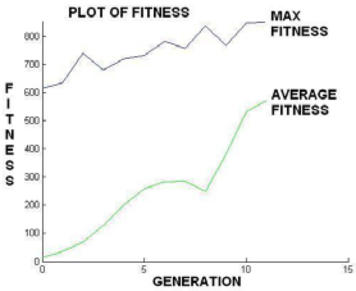


Fig. 5. The evolution of the *wall-seek* behavior is reached after 11 generations

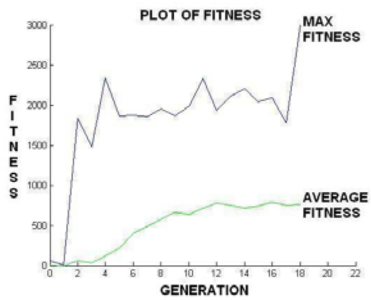


Fig. 6. An appropriate *wall-follow* behavior is obtained after 18 generations

The development of the *cylinder-seek* behavior employed a similar architecture for the neural network than the previous evolved behaviors. However, a set of fifteen patterns was used to train the neural network with backpropagation. These patterns correspond to the main situations the robot finds when the behavior has to be activated. Instead of using a test set of patterns, the trained network was tested on the simulator to assert an adequate generalization. Basically, on this behavior the Khepera, when running around the squared arena, avoids obstacles until is driven close to a cylinder, and then the robot is stopped. Generalization is obtained after 1,184 generations when the total-error drops below 0.02 (Figure 7).

Finally, the *cylinder-pickup* and the *wall-deposit* behaviors were programmed as algorithmic routines with a fixed number of iterations for clearing the space for lowering the arm, opening the claw, and moving upwards the arm. These two actions always have to be carried out in the same sequence, but in a reversed order, to either grip an object or to release the same object. Due to the sequential nature of these behaviors we preferred the use of algorithmic versions of them, rather than trying to shape those behaviors using supervised learning. Once all the mentioned behaviors were appropriately designed, and evolved, they were implemented on the real Khepera (Figure 8). Due to the simulator properties of introducing noise in the simulated sensors and actuators, and the topology of the neural network acting as a tolerant classifier to noisy inputs; the transference of the evolved behaviors from the simulator to the real robot was made in a straightforward manner. On the other hand, the algorithmic routines for handling the gripper required none modification.

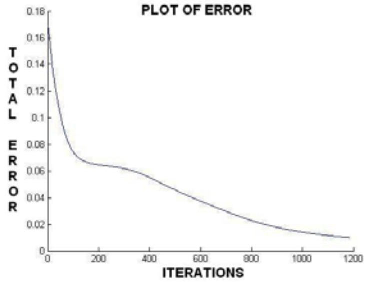


Fig. 7. The Neural Network adequately classifies the 15 training patterns after 1,184 iterations with a total-error below 0.02

Fig. 8. The Khepera robot set in the middle of a squared arena with simulated food (wooden-cylinders)

4 The CASSF Action Selection Mechanism

A centralized model of Action Selection with sensor fusion (CASSF) has already been used for allowing a winning competing behavior to be expressed at the time [1, 12, 13]. A main control loop updates at every step of the simulation sensor readings and motor commands. The different sensors readings from the Khepera, the infrared, the odometry, and the optical barrier in the gripper; all they 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 to be executed. However, not all the variables are equally relevant for a particular behavior. For instance, the searching of a place for releasing a cylinder requires the presence of an object in the gripper. Additionally, behaviors 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 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 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 the foraging behavior, the perceptual variables wall_detector(e_w), gripper_sensor(e_g), cylinder_detector(e_c), and corner_detector(e_r); they form the context vector, which is constructed as follows ($\mathbf{e} = [e_w, e_g, e_c, e_r]$, $e_w, e_g, e_c, e_r \in \{1,0\}$). Five different behaviors return a current busy-status (c_i) indicating that ongoing activities should not be interrupted. Next, the current busy-status vector is formed as described next, $\mathbf{c} = [c_s, c_p, c_w, c_f, c_d]$, $c_s, c_p, c_w, c_f, c_d \in \{1,0\}$, for *cylinder-seek*, *cylinder-pickup*, *wall-seek*, *wall-follow*, and *wall-deposit* 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:

$$\mathbf{s}_i = \mathbf{c}_i w_b + \mathbf{e}_i \mathbf{w}_i^e \quad \textit{for} \tag{3}$$

$$\begin{array}{llrrrrl}
\text{cylinder}-\text{seek} & \mathbf{w}_s^e = [& 0.0, & -0.15 & -0.15, & 0.0 &], \\
\text{cylinder}-\text{pickup} & \mathbf{w}_p^e = [& 0.0, & -0.15, & 0.15, & 0.0 &], \\
\text{wall}-\text{seek} & \mathbf{w}_w^e = [& -0.15, & 0.15, & 0.0, & 0.0 &], \\
\text{wall}-\text{follow} & \mathbf{w}_f^e = [& 0.15, & 0.15, & 0.0, & 0.0 &], \\
\text{wall}-\text{deposit} & \mathbf{w}_d^e = [& 0.15, & 0.15, & 0.0, & 0.15 &]
\end{array}$$

Behaviors are chosen accordingly to the task that is to be resolved. For instance, the simulated foraging task resembles the food-retrieval of a wall-following rat when afraid to facing a novel environment. The robot basal ganglia proposal of Prescott et al. [14] address more neurophysiological issues related to this work. Whereas for cleaning an arena full of cans, and depositing them close to walls, is solved using a complete evolutionary approach in Nolfi [15]. For our experiments we have decomposed the foraging task in algorithmic behaviors that can be assembled together to show this behavior. Finally, we assume that behaviors were previously learnt, and they can be considered as selected chunks of memory.

The centralized model can be thought as a particular instance of a neural network (Figure 9), and evolutionary learning could be used to set its weights instead of the delta rule. 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 to win the competition. In order to contribute on the calculation of the salience, a winning behavior sends a busy signal to the computing output layer of the neural network. A behavior is deselected when its salience is below the strongest salience.

5 Experiments and Results

The foraging task was set in a squared-box for running the experiments. A standard RS232 interface was used to connect the robot to the computer host. The simulated 'food' was made of four wrapped foil-paper wooden-cylinders, which were set in the middle of the arena. Foil paper was used to facilitate the use of the resistivity sensor in the claw of the gripper. The definition of the behaviors, as described earlier, is as follows: *cylinder-seek* is used to travel the arena searching for food while avoiding obstacles, *cylinder-pickup* clears the space for grasping the cylinder, *wall-seek* is employed to locate the nearest wall and avoiding obstacles, *follow-wall* travels next to a proximate wall, and *cylinder-deposit* lowers and opens an occupied gripper.

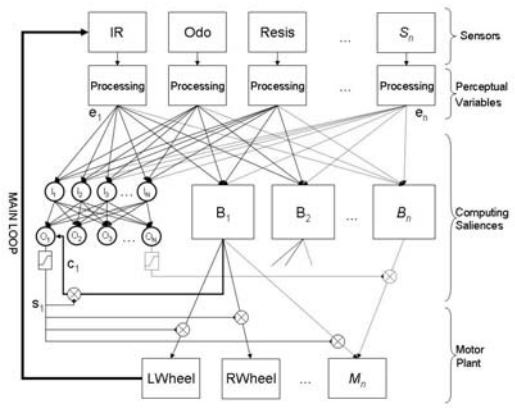


Fig. 9. Perceptual variables (e_i) form the input for the Salience Neural Network. The output selection of the highest salience (s_i) is gated to the motors of the Khepera. Notice the busy-status signal (c_1) from behavior B1 to the output neuron

Commonly, the foraging task is formed by four grasping-depositing bouts of foiled cylinders; thus, one typical grip of a cylinder is shown in Figure 10. Additionally, the next ethogram (Figure 11), and some related statistics (Table 1) resume this behavior. The time resolution of the ethogram is reported in seconds, and seconds and milliseconds for the statistics analysis. Neither of the behaviors was selected before cylinder-seek. However, it took only 0.01 seconds for cylinder-seek to be selected. Next, cylinder-pickup was selected at 3.25 seconds of the total elapsed time (42.64 seconds). Wall-seek was selected at 4.71 seconds, followed by wall-follow at 6.87 seconds, wall-deposit was selected the last after approximately 7 seconds of having initiated the search of a can. The remaining three bouts were repeated in a similar fashion with different search periods. The behavior selected the most was can-seek with 5 times, all the rest with 4 times.

6 Conclusion

In this paper we have shown that non-homogenous behaviors such as wall-seek and wall-follow were the result of using genetic algorithms, and cylinder-seek was obtained using backpropagation. The three behaviors share the same neural network topology, but with different weights, for having the neural network to behave in a different manner. The use of the same circuit with different weights may evidence flexibility in the use of shared circuits for motor control (plasticity). The central model of action selection coupled well with the design of the behaviors, which were developed using Neural Networks, GAs, and backpropagation, together with algorithmic routines designed by hand.

The calculation of the salience in CASSF drives the selection of non-homogenous behaviors for the resolution of a foraging task. Additionally, the use of a busy-status signal accounts for boosting the salience when the behavior has to be maintained, and the disconnection of the same signal favors the interruption of any of the behaviors during the simulation. The occurrence of regular bouts of grasping-depositing cans offers a proof of the correct integration of the non-homogenous behaviors with the model of central selection. Furthermore, due to its detached-from-the-behaviors design, we consider CASSF a robust model of action selection. Besides, adaptability and extensibility is offered, at relatively low cost, by allowing independent modification of both the selection module and the behaviors.

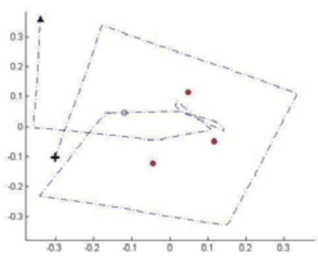


Fig. 10. A regular food-collection bout, the cross indicates the initiation of the behavior

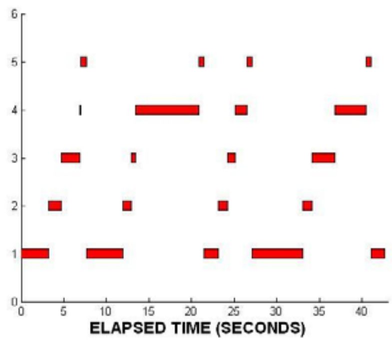


Fig. 11. Ethogram for a typical run of the foraging task

Table 1. Elementary statistics for a representative run of the Khepera with CASSF

Behavioral Elements	Freq	Latency	TotDur	TotDur%	Mean	StdDev	StdErr	MinDur	MaxDur
1. cylinder-seek	5.00	0.01	16.74	39.27	3.35	1.86	0.83	1.54	5.99
2. cylinder-pickup	4.00	3.25	4.59	10.75	1.15	0.20	0.10	1.04	1.45
3. wall-seek	4.00	4.71	6.24	14.63	1.56	1.03	0.52	0.50	2.68
4. wall-follow	4.00	6.87	12.58	29.50	3.14	3.17	1.59	0.14	7.37
5. wall-deposit	4.00	7.01	2.48	5.82	0.62	0.01	0.01	0.61	0.64
6. none	1.00	0.00	0.01	0.02	0.01	0.00	0.00	0.01	0.01
Total	22.00	0.00	42.64	100.00	1.94	1.91	0.41	0.01	7.37

Once the behaviors are evolved or learnt, the next step is the evolution of the matrix $\mathbf{w}^e$. Moreover, the co-evolution of this matrix, which defines the global control architecture, with the evolved behaviors can simultaneously be made. Finally, as a further step we aim to automatically obtain, by means of computer evolution, the most adequate definitions of the perceptual variables with the later objective of having a reduction in the number of decisions made by the human designer.

Acknowledgments

This work has been sponsored by CONACyT-MEXICO grant #SEP-2004-C01-45726.

References

1. 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.
2. Einstein, A. and L. Infeld, The Evolution of Physics. 1966: New York: Simon and Schuster.
3. Holland, J., Adaptation in Natural and Artificial Systems. 1975: University of Michigan Press, Ann Arbor.
4. Nolfi, S. and D. Floreano, Evolutionary Robotics. 2000: The MIT Press.
5. Santos, J. and R. Duro, Artificial Evolution and Autonomous Robotics (in Spanish). 2005: Ra-Ma Editorial.
6. Balakrishnan, K. and V. Honavar. Properties of Genetic Representation of Neural Architectures. in the World Congress on Neural Networks (WCNN '95). 1995.
7. Morrison, R.W. and K.A. De Jong. A Test Problem Generator for Non-Stationary Enviroments. in the Congress on Evolutionary Computation (CEC-99). 1999.
8. Mondana, F., E. Franzi, and I. P. Mobile robot miniaturisation: A tool for investigating in control algorithms. in Proceedings of the 3rd International Symposium on Experimental Robotics. 1993. Kyoto Japan: Springer Verlag.
9. Webots, *http://www.cyberbotics.com*. 2005, Commercial Mobile Robot Simulation Software.
10. Floreano, D. and F. Mondana. Automatic Creation of an Autonomous Agent: Genetic Evolution of a Neural-Network Driven Robot. in From Animals to Animats III: Proceedings of the Third International Conference on Simulation of Adaptive Behavior. 1994: MIT Press-Bradford Books, Cambridge MA.
11. Bajaj, D. and M.H. Ang Jr. An Incremental Approach in Evolving Robot Behavior. in The Sixth International Conference on Control, Automation, Robotics and Vision (ICARCV'2000). 2000. Singapore.
12. 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.
13. 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.
14. Prescott, T.J., F.M. Montes Gonzalez, K. Gurney et al., A robot model of the basla ganglia: behavior and intrinsic processing. Neural Networks (in press).
15. Nolfi, S., Evolving non-trivial behaviors on real robots: A garbage collection robot, in Robotics and Automation System. 1997. p. 187-198.

Formations in Collective Robotics

Yazid León Fernández de Lara* and Angélica Muñoz Meléndez

INAOE, Luis Enrique Erro No. 1
72840 Tonantzintla Puebla, México
yazid@ccc.inaoep.mx, munoz@inaoep.mx

Abstract. In this paper, the problem of coordination of groups of robots or agents that have to arrange themselves in spatial formations is addressed. The designing and implementation of mechanisms to enable a group of agents to create and keep formations is important for a number of applications such as collective exploration and monitoring of environments. Three models for the generation and keeping of formations using, respectively, semi-autonomous, composed autonomous and simple autonomous agents are presented and compared.
key words: agent, simulated robot, formations, spatial patterns.

1 Introduction

Collective robotics is interested in designing systems consisting of various robots able to solve problems jointly. The robots taking part in a multi-robot system are homogeneous, simpler in terms of design and control, and cheaper than specialized single systems. Multi-robot systems are oriented to solve problems in which the participation of a single robot is not sufficient or is very expensive, in terms of design and time for example, such as transportation of voluminous objects, hazardous material handling, environment exploration, monitoring and terrain coverage.

The design of mechanisms to enable groups of robots to generate and keep spatial formations is a line of research in the vast field of collective robotics. A formation is defined as a spatio-temporal structure constituted by a set of robots where each one represents a point or vertex of the structure. The design of multi-robot systems presents significant challenges. When a group of robots shares a common space work, each member of the group is a mobile obstacle for the others. In order to avoid this problem, coordination is required among robots. Multi-robot coordination consists in synchronizing the respective individual movements of the robots, in such a way that they execute their tasks avoiding collisions.

The rest of this article is organized as follows: Section 2 outlines related work, section 3 describes our proposal and methodology to generate and keep formations, section 4 discusses experimental results, and section 5 addresses conclusions and perspectives of the project.

* Supported by CONACyT, contract number 182899.

© *A. Gelbukh, R. Monroy. (Eds.)*
Advances in Artificial Intelligence Theory
Research on Computing Science 16, 2005, pp. 223-232

2 Related Work

The organized behavior exhibited by certain animal groups such as flocks, fish schools, army ants and the spontaneous formation of structures such as the rays in a zebra or the structure of the DNA involve mechanisms of self-organization and coordination [1].

With the purpose of studying self-organization, artificial colonies called multi-agent systems have been created. An agent is an autonomous virtual robot which is able to perceive its environment and to act on this.

Some mechanisms that have been proposed for self-organizing multi-agent systems are described below.

Reynolds [2] developed a simple egocentric behavioral model for flocking which is instantiated in each member of a group of simulated agents called *boids*. Reynolds' boids keep proper position and orientation within the flock by balancing their tendency to avoid collisions with neighbors, by matching the velocity of nearby neighbors and moving towards the center of the flock. The rules that these boids apply are very simple, e.g. avoid collisions, copy other boids' velocities and directions, search the center of a group and move laterally away from any boid that blocks the view. An important contribution of Reynolds' work is the generation of apparently orchestrated collective behavior in agents from local perception-based rules.

Another example of systems that work cooperatively in the construction of collective structures is Resnick's termites [3]. He proposed simple rules such as wander, carry and drop objects in order to obtain large groups of agents that cooperatively build structures that appear to have some sort of planning behind them. Resnick's termites wander randomly until they bump into a piece of wood. If a termite is carrying a piece of wood when it bumps, it drops the piece and continues to wander. If the termite is not carrying any piece, it picks up the one it bumped into and continues to wander.

Sugihara and Suzuki [4] performed experiments with multiple simulated robots that are able to form stable formations when each robot executed an identical algorithm for position determination within a group. Each robot of the group is able to perceive the relative position of all other robots and to move one grid position each time. Regular geometric shapes such as a circle can be formed by these robots.

Ünsal [5] developed an algorithm for self-organization of ants' colonies or multi-agent systems, where the agents are identical both physically and functionally. Each Ünsal's agent has limited capabilities and limited knowledge of the environment, but as a colony, they exhibit an intelligent behavior.

Ünsal's algorithm is based on beacon recognition, where beacons are used to identify the meeting point of agents. With this algorithm, separated groups of agents around several beacons are formed. An agent approaches the beacon until it is within a predefined distance d in a vicinity $d + \varepsilon$. If any agent is within the area defined by $d - \varepsilon$, it is rejected by the beacon. Within the ε area, each robot moves far away from their closest neighbor therefore creating formations such as circles, spheres and paraboloids.

Balch and Arkin [6] present several formations for a group of four robots such as: lines, columns, diamonds and wedges. These formations were tested using both simulated and physical robots. For each formation, each robot has a specific position based on its identification number (ID). These behaviors are identical for each of the four robots, except for a robot who is designated as the leader. The tasks of each robot are to move towards a goal location, avoid obstacles, avoid collisions with other robots and keep the original formation. Dynamic formations are kept through two steps: First perception process, here, the robot's proper position in a formation is determined based on its current environmental perception. During, the second step, the motion process, motion commands are generated in order to guide a robot towards its correct location. Each robot computes its proper position in the formation based on the other robot locations.

Mataric [7] uses a simple, general formation algorithm based only on local sensing and control. This algorithm has been tested using both simulated and physical robots. Each robot in the formation is autonomous and has no information about other robots. Each robot has an ID that is broadcasted regularly as a message. Robots' IDs are detectable by other robots. The robot that is located in front of the formation is considered the leader and any robot can take this role. Every robot knows at any time the total number of robots participating and the kind of formation they should have. Except for the leader, each robot follows a designated neighbor known as its friend, keeping certain distance and angle with respect to its heading. Every robot has at most one follower, except for the leader which can have one or two. In this way the robots connect in a "chain of friendships".

3 Proposal

In this section, a serie of experiments using simulated robots that have to form spatial formations is described. Two kind of formations were studied with the purpose of developing mechanisms to coordinate the collective exploration of large environments: static and dynamic. Static formations are generated by immobile agents, whereas dynamic formations are kept by mobile agents.

Experiments were performed on StarLogo ©, an environment of simulation developed at MIT by Resnick [8].

3.1 Agents

Three different agent models were studied in order to test the abilities required to generate and keep formations.

The first agent model was a **semi-autonomous agent**. This agent is a point in a two-dimensional space, it can move a certain number of positions and change its direction in a given angle. Semi-autonomous agents are also able to directly transmit their position to other agents at any time. The generation of a formation is based on this information, that is supposed to be transmitted and received without error. Because agents depend on this information and are unable to collect it by themselves, they are considered semi-autonomous. The agents have a radius of perception and information transmission of two positions in the environment (see Figure 1 a)).

The second model is denominated **composed autonomous agent** and was proposed to improve the first model. As semi-autonomous agents are unable to distinguish their partners, by themselves, agents were redefined to be able to perceive locally this information. A composed autonomous agent is a multi-agent system consisting of four agents: a head, a center and two arms. These agents are points in a two-dimensional space painted with predefined colors to identify the parts of a composed body. This information is used by composed autonomous agents to estimate their orientations. The perception of these agents is merged from the individual perception of each one of their parts. These agents do not directly transmit their position nor their direction, these are rather estimated based on the local agent's perception and the recognition of the colors that identify their bodies (see Figure 1 b)).

The third model, **simple autonomous agent** was proposed to capture the advantages of previous models. This model aims for autonomy and simplicity, as in the second and first agent models. These agents are points in a two-dimensional space. They can move a certain number of positions and change their direction in a given angle. Simple autonomous agents are painted of pre-defined red scale colors according to their orientation, e.g. a dark red agent is oriented towards the north, a red agent is oriented towards the east and a clear red agent is oriented towards the northwest. These agents do not directly transmit their position nor their direction, this information is rather estimated from local perception. The agents have a radius of perception of five positions in the environment (see Figure 1 c)).

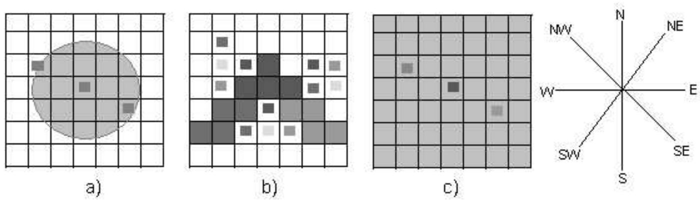


Fig. 1. Agent representation: semi-autonomous agents a), composed autonomous agents b), and simple autonomous agents c). Each figure shows an agent in the center surrounded by two agents: one oriented to the east and one oriented to the northwest. The radius of perception of the agent in the center is also indicated in gray color

Each one of the previous agents has an ID and a set of properties described in Table 1.

Table 1. Properties of semi-autonomous, composed autonomous and simple autonomous agents

Model	Perception	Action
Semi autonomous agent	Own absolute position Signals of neighbors Goal position	Move towards the goal position Transmit position Follow a neighbor
Composed autonomous agent	Own absolute position Marks of neighbors Goal position	Move towards the goal position Follow a neighbor
Simple autonomous agent	Own absolute position Goal position	Move towards the goal position Follow neighbor

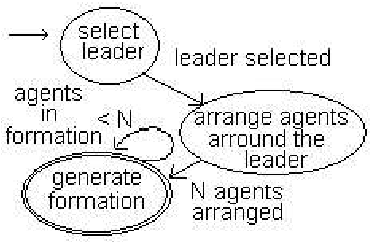


Fig. 2. General algorithm to generate static formations for N agents

3.2 Static Formations

Static wedge formations are important in applications where an exploration of the environment is required, e.g. military applications and terrain coverage.

Static wedge formations using semi-autonomous and composed autonomous agents were programmed. In order to do a wedge formation, agents have to be, arranged around a mark known as the *meeting point*. Agents must reach the meeting point in order to generate a static formation.

As soon as an agent reaches the meeting point, this agent takes the current leadership in the formation. Then, three rules are applied: first, the leader decides randomly the number of branches among 2, 4 and 6 in the formation and transmits the leadership to the agent who is in front of it and goes back two steps. Second, the other agents go towards the current leader in a succession of turns determined by their IDs. And third, if an agent who is formed perceives another agent within a predefined vicinity, it goes back two steps keeping its orientation.

Static formations using simple autonomous agents were also programmed. As these formations differ slightly from the previous ones, they are called approximate wedges. Neither arrangement around a meeting point nor communication among agents is required in order to generate approximate static formations.

Two rules are applied to generate this formation: first, agents wander avoiding collisions and turning randomly to their left or right. And second, an agent meets other agent if it is within its local radius of perception. These rules produce eventually the grouping of agents in "wedges".

The control of a group of agents whose main task is to generate a static formation is illustrated in Figure 2

3.3 Dynamic Formations

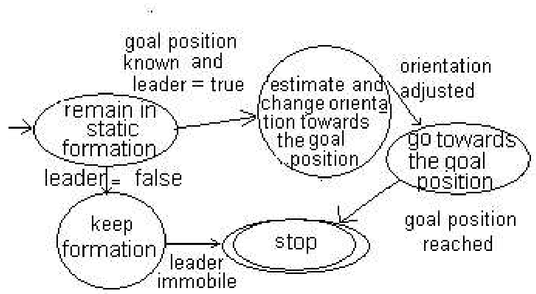


Fig. 3. General algorithm to *keepformation*. The state *keep formation* is instantiated according to the agent model (see Figure 4)

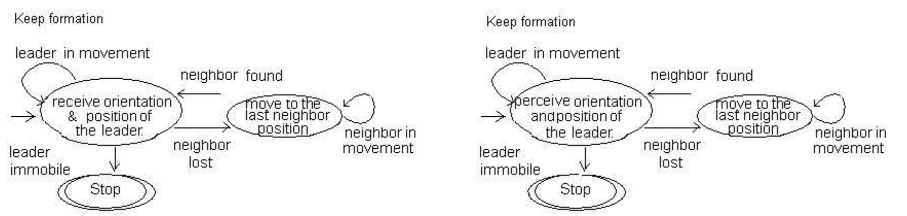


Fig. 4. Dynamic formation using semi-autonomous agents (left) and dynamic formation using composed autonomous and simple autonomous agents (right). Figures 5 and 6 illustrate the estimation of the leader's positions and orientation

Dynamic wedge formations consisting of semi-autonomous and composed autonomous agents were programmed.

In dynamic formations, a group of agents that has achieved a static formation moves towards a goal position keeping the initial formation as far as possible. If external perturbations, such as obstacles or mobile objects, force agents to undo their formation, they should know what to do in order to recover the initial or a similar formation. The control of agents that have to keep a dynamic formation in an environment without obstacles is illustrated in Figure 3.

In order to keep a dynamic formation, semi-autonomous agents have to communicate among them. Two rules are applied to keep a dynamic formation: first, the leader estimates and changes its orientation towards the goal position. Then it transmits its orientation and its movements to its partners while going towards the goal position. Second, the partners receive the leader's orientation and follow the movements transmitted by the leader (see Figures 3 and 4).

To keep a dynamic formation, composed autonomous agents have to estimate their neighbor's orientation and position by local perception from the marks on their bodies. The leader estimates the orientation towards the goal position and goes towards the goal position. Then, the partners copy their orientation and go to the last position where they perceived a partner (see Figures 3, 4, and 5).

Dynamic formation using simple autonomous agents were also programmed. In order to keep an approximate wedge formation, agents have to be arranged firstly, in a static formation. In this formation agents are autonomous, they decide their actions based on their local perception and their own estimations, e.g. the agent who is the leader estimates the orientation towards the goal position, and goes towards this position. Agents copy and follow their partners orientation that is perceived by their owns means, reforming (see Figures 3, 4 and 6).

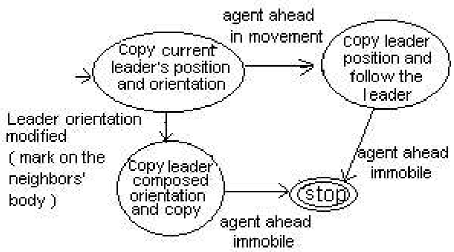


Fig. 5. Perception of the position and orientation of the leader in composed autonomous agent model

4 Experimental Results

The different agent models that were previously presented were tested in similar experiments whose purpose was keep dynamic wedge formations.

The number of agents that participate in these formations go from twenty to eighty in the experiments using semi-autonomous and simple autonomous agents and, from three to ten groups in the experiments using composed autonomous agents. The distance that agents have to move towards the goal position is about 30 steps in environments without obstacles.

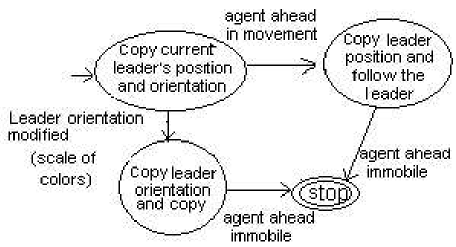


Fig. 6. Perception of the position and orientation of the leader in simple autonomous agent model

Semi-autonomous, compose autonomous and simple autonomous agents make formations having occasional leaders. They have to move comparable distances, from an initial position towards a goal position, keeping the formation. The time spent by semi-autonomous, composed autonomous and simple autonomous agents to do this task is shown in Figure 7.

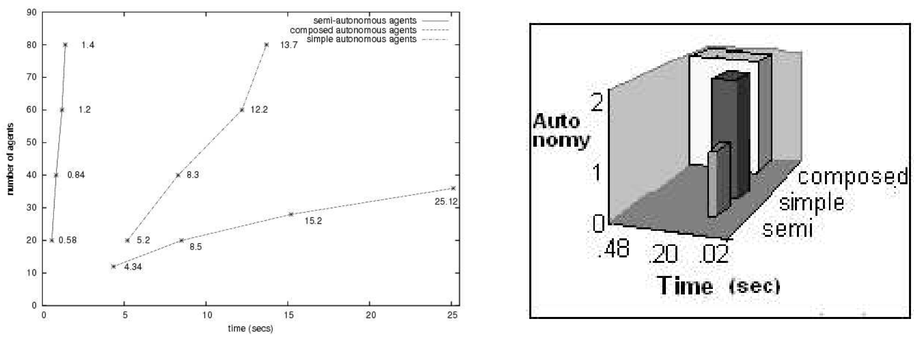


Fig. 7. Time spent by groups of agents who move from an initial position towards a goal position of the environment: semi-autonomous agents, composed autonomous agents and simple autonomous agents (left) and the time that agents of each model spend on average to do similar tasks and their levels of autonomy are compared (right)

As Figure 7 shows, the average time required for a semi-autonomous agent to keep a dynamic formation from an initial to a goal position is 0.02 secs, whereas the average time required by a composed autonomous agent to do the same task is 0.48 sec. That means that a semi-autonomous agent is in an average 24 times faster than a composed autonomous agent. This difference can be explained by the fact that the first agent does not spend time estimating the position and orientation of its partners in contrast with the second, which is in fact integrated with four agents who have to perceive their neighbors in addition to merge their perception in order to estimate their position and orientation. On the one hand,

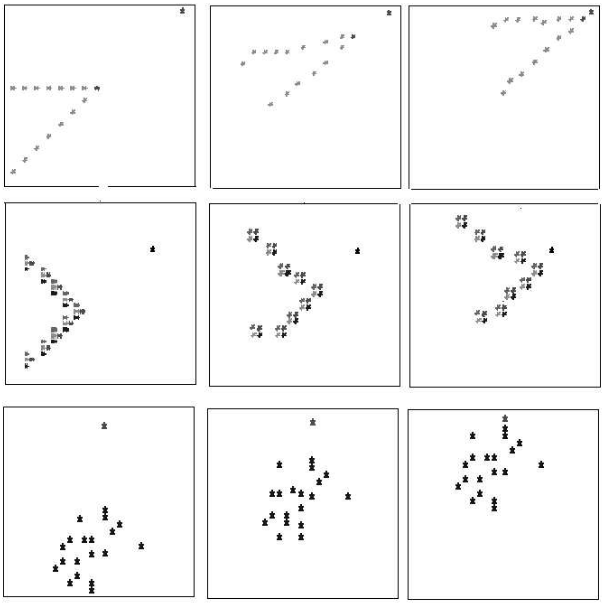


Fig. 8. Snapshots of semi-autonomous (top), composed autonomous (center) and simple autonomous (bottom) agents. Figures indicate from left to right respectively, the initial, an intermediate, and the goal position

semi-autonomous agents are faster than composed autonomous agents, but on the other hand, the second are able to operate by their own means.

Average total time was calculated from the averages spent by each group to complete a task, e.g. 0.029, 0.021, 0.02 and 0.0175 for agents of respectively 20, 40, 60 and 80 of the first model.

Figure 7 shows the performance of one agent of each model measured with respect to the time spent to do the same task and its degree of autonomy. Autonomous agents are able to operate based on their own sensor and actuators and have a high level of autonomy (2 in the graph), whereas semi-autonomous agents whose operation depends on direct communication have a medium degree of autonomy (1 in the graph). The best performing model was the third, the model that takes advantage of previous models. Simple autonomous agents are good enough to keep a dynamic formation, they are quite faster in comparison with semi-autonomous agents, and as independent as composed autonomous agents.

Figure 8 shows some snapshots of groups of agents controlled by the algorithms that were previously presented.

Local perception is not enough to enable autonomy. Agents must be externally distinguishable in order to enable them to perceive their orientation as it happens in second and third models.

5 Conclusions and Perspectives

In this paper, three different agent models, semi-autonomous, composed autonomous and simple autonomous agents, to generate and keep respectively, static and dynamic spatial formations in environments without obstacles were presented and compared.

These models were designed constructively, i.e. each model attempts to improve the previous one, taking the advantages that were detected. The third model, simple autonomous agents, was the best of the three because of its simplicity and autonomy.

The design of mechanisms to generate and keep spatial formations is important in order to define robust multi-agent systems able to transport objects, handle hazardous material and cover terrain.

In the literature, the advantages of decentralization over centralization, as well as the advantages of local perception-based systems over direct communication-based systems are taken for granted. The main contribution of this paper is the analysis of different agent model based on experimental results in order to compare their reliability.

The selection of an agent model depends on the environmental conditions, in environments where communication can be established with minimal error, semi-autonomous agents can operate efficiently, whereas composed autonomous and simple autonomous agents are better when communication is incomplete or noisy.

This work is in progress. Future work will consider richer environments including fixed and mobile obstacles, as well as mobile goal positions, and the definition of rules enabling agents to rearrange themselves in formations they were forced to undo.

References

1. Camazine, S., Deneubourg, J., Nigel, R., Sneyd, J., Theraulaz, G., Bonabeau, E.: Self-organization in Biological Systems, Princeton University Press (2001)
2. Reynolds, C.W.: Flocks, Herds, and Schools: A Distributed Behavioral Model. Computer Graphics (1987) 25-34
3. Resnick, M.: Termites, Turtles and Traffic Jams. Explorations in massive computation. MIT Press (1996)
4. Sugihara, K., Suzuki, I.: Distributed Motion Coordination of Multiple Robots, IEEE International Symposium Intelligent Control (1990) 138-143
5. Ünsal, C. and Bay, J. S.: Spatial Self-Organization in Large Populations of Mobile Robots, IEEE International Symposium on Intelligent Control, Columbus, Ohio (1994)
6. Balch, T., Arkin, R.: Behavior-Based Formation Control for Multirobot Teams, IEEE Transactions on Robotics and Automation, vol. 14, No. 6 (1998)
7. Mataric, M., Fredslund, J.: Huey, Dewey, Louie, and GUI - Commanding Robot Formations, IEEE International Symposium on Robotics and Automation (2002) 11-15
8. StarLogo 2.1 http://education.mit.edu/starlogo/

MINI-TRANS: A Multi-robot System with Self-assembling Capabilities

María Guadalupe Jiménez Velasco[1,2] and Angélica Muñoz Meléndez[1]

[1] Instituto Nacional de Astrofísica, Óptica y Electrónica
Coordinación de Ciencias Computacionales
Luis Enrique Erro No. 1
72840 Tonantzintla Puebla, México
marvel@ccc.inaoep.mx, munoz@inaoep.mx
[2] Instituto Tecnológico Superior de Comalcalco
Carr. Vecinal Comalcalco-Paraiso Km 2
86650 Comalcalco Tabasco, México

Abstract. Recently, reconfigurable multi-robot systems have gained interest due to advantages such as reconfigurability, robustness and modularity they offer in comparison with mono-robot systems. Self-reconfigurable multi-robot systems have potential applications in the fields of collective exploration, object transportation and rescue. However, the design of reconfigurable multi-robot systems presents many challenges such as the selection of materials, equipment, morphology and control to enable them to coordinate their actions, to assemble their bodies and to avoid collisions. This paper describes the development of a prototype of multi-robot system with self-assembling capabilities. Preliminary experimental results are also presented.

1 Introduction

Multi-robot systems refer to a group of integrated robots that work together. Modular robots are a subclass of multi-robot systems, they consist of self-assembling autonomous robots that have devices such as a grippers, attachment points or electromagnets to connect and disconnect to other robots. This capability enables modular robots to change the organization of their modules and is called reconfigurability. These kinds of systems have potential applications in rescue, object transportation and exploration of environments with irregular surfaces.

Reconfigurability can be static or dynamic. Static reconfigurable robots reorganize their structure off-line, whereas dynamic reconfigurable robots can reorganize their structure on-line, during run-time execution. A human user or another robot may assist dynamic reconfigurable systems with their reconfiguration. There are also dynamic self-reconfigurable systems able to achieve their reconfiguration by themselves.

Yim et al. [1] classified self-reconfigurable robots according to their form of configuration as described below.

© A. Gelbukh, R. Monroy. (Eds.)
Advances in Artificial Intelligence Theory
Research on Computing Science 16, 2005, pp. 233-242

Mobile reconfigurable systems consist of mobile autonomous robots able to work independently of each other. Occasionally, these modules assemble to perform tasks they are unable to do individually, e.g. Millibot-trains [2].

Lattice reconfigurable systems are arranged in a cell form, the modules connect and disconnect to their neighbors within the limits of their structure, e.g. MTRAN [3].

Chain reconfigurable systems consist of semi-independent modules that must be connected at least to another module, e.g. Polybot [1].

Recently, reconfigurable multi-robot systems have gained interest due to the advantages such as reconfigurability, robustness and modularity they offer in comparison with mono-robot systems. Sometimes, these robots are also self-repairing and self-maintining, i.e. they are able to replace one damaged module with another one [4].

The design of reconfigurable multi-robot systems presents many challenges such as the selection of materials, equipment, morphology and control to enable them to coordinate their actions, to assemble their bodies and to avoid collisions.

Coordination is an important issue when designing multi-robot systems. As far as we know, the design of mechanisms that enable a group of mobile robots to interact with them, is an open problem in robotics. We denote **coordination** as the capability of the members of a multi-robot system to perform actions jointly to reach a common objective, avoiding collisions and conflicts among them.

In this paper, the progress in the development of a prototype of a mobile self-reconfigurable multi-robot system is presented. The rest of the paper is organized as follows: Section 2 presents related work; Section 3 describes the development of our prototype of a modular reconfigurable system: Mini-trans; Section 4 shows the results obtained from experiments using our prototype, and finally, Section 5 addresses conclusions and perspectives of this work.

2 Related Work

Because the prototype presented in this paper is a mobile self-reconfigurable multi-robot system, recent works on these systems are summarized below.

Swarm-bot is a project sponsored by the Future and Emerging Technologies program of the European Community [5]. The purpose of this project is to design and build physically a mobile reconfigurable system consisting of 30 to 35 small and autonomous robots called S-bots. S-bots are equipped with 50 sensors and 9 actuators. S-bots are able to perceive their environment, communicate with other S-bots and assemble themselves into flexible or rigid forms. Each S-bot is able to perform different tasks such as navigation, exploration and grasping objects. The system uses a distributed adaptive control architecture inspired on ant colony behaviors.

Carnegie Mellon University developed a group of robots called **Millibot-trains** with self-assembling capabilities [2]. Each Millibot is approximately $11cm.$ long and $6cm.$ wide. The modules of Millibot-trains system are equipped with sensors such as sonars, IR sensors and video cameras. They also have capabilities to navigate, to explore, to assemble and to communicate in a wireless network. Each Millibot consists of three modules: one module for movement, one for perception of the environment and one for control. Each Millibot can assemble another one using a pin-hole based device, which provides stability but demands a lot of exactness in connections.

SMC rover is a mobile reconfigurable system developed by Kawakami et al. [6]. This system consists of exploring and navigating robots. Each robot is integrated with various modules called Unirovers which are solar cell modules or other types of support modules. As well as Millibot-trains, Unirovers assemble together using a pin-hole device.

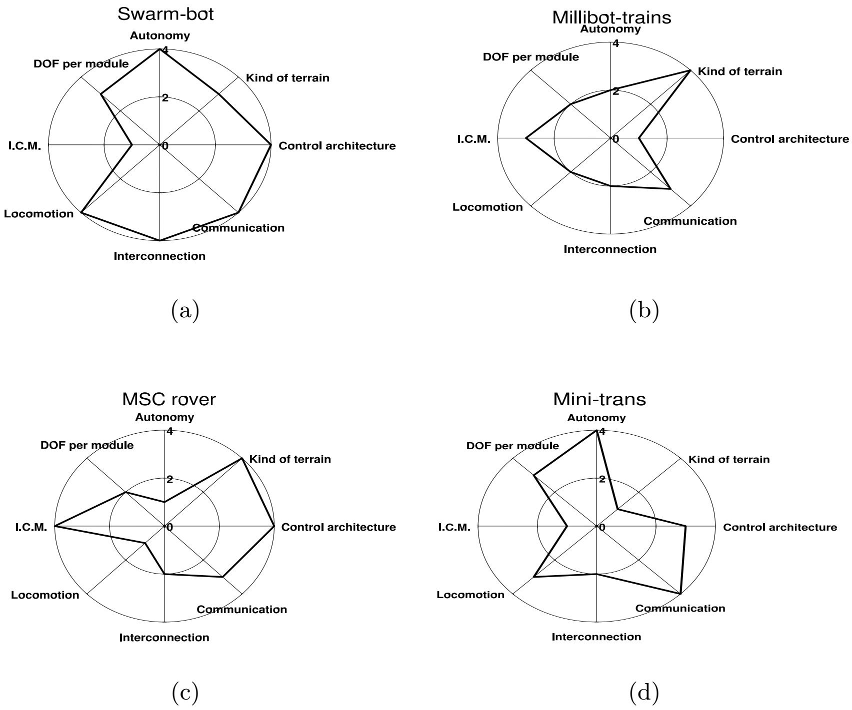
Figure 1 summarizes the main features of the modular robots reviewed above. The main features of Mini-trans, our prototype which is described in Section 3, are also presented in this figure. Important notions used in this paper are precised below.

- A robot that is able to make decisions based on its own perception and processing is considered autonomous. On the other hand there are the complete remote-controlled or non-autonomous robots. A semi-autonomous robot is able to make decisions based on external processing. A mainly remote-controlled robot is able to make certain decisions but it could also be controlled by a human operator.
- The communication based on explicit exchanges is considered direct communication, whereas the communication based on the observed behavior is considered indirect communication.
- The control architecture that depends on the multiple robot interactions is considered decentralized. The control architecture that depends on a central scheduler is considered centralized and fixed. A hybrid architecture comprises two layers, one containing centralized mechanisms able to plan actions, and other containing decentralized mechanisms able to react at immediate events. A centralized control architecture based on a non-fixed hierarchy is considered a dynamic hierarchy.

3 Mini-trans System

The Mini-trans system consists of three homogeneous and autonomous modules with self-organizing and self-assembling capabilities. Its development is divided into six steps: 1) design and building of one prototype, 2) definition of the system behaviors, 3) experimentation, 4) analysis of results, 5) modification and improvement and 6) generalization of behaviors. This work is in progress, its development is currently in step 5.

This section presents the design, implementation and control of the system.



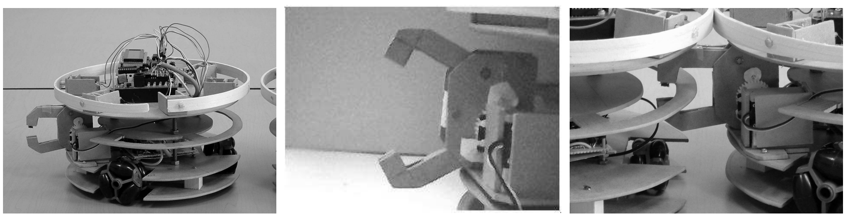
	Self-reconfigurable systems			
Feature	**Values**			
	1	2	3	4
Autonomy	Completely remote-controlled	Mainly remote--controlled	Semi-autonomous	Autonomous
DOF	1	2	3	> 3
Individual capabilities per module (I.C.M.)	Homogeneous hardware & software	Homogeneous hardware, heterogeneous software	Heterogeneous hardware & software	Complementary heterogeneous hardware & software
Locomotion	Wheels	Tracks	Omnidirectional wheels	Wheels & tracks
Interconnection	Rigid	Semi-rigid	Flexible	Rigid & flexible
Communication	Without communication	Indirect	Direct	Direct & Indirect
Control architecture	Centralized & fixed	Centralized based on a dynamic hierarchy	Hybrid	Decentralized
Kind of terrain	Flat	Few rough	Rough	Extremely rough

Fig. 1. Comparison of main features in four self-reconfigurable systems: Swarm-bots (a), Millibot-trains (b), SMC rover (c) and Mini-trans (d). Low values in the table (1 and 2) represent low adaptability, whereas high values (3 and 4) represent high adaptability. Swarm-bots is, as shown in Figure (a), the best performing system according to this classification. (d) Shows the features of the system proposed in this paper, which is described in Section 3

3.1 Hardware

Various versions of Mini-trans were designed and tested. The definitive version of the modules consists of four 26cm diameter wooden circular plates. These plates are arranged one over another supported by screws. Locomotion elements were placed on first plate. A gripper was placed on second plate. The fourth plate contains a contact sensor belt, an IR reflective optosensor and a Handyboard©. For its locomotion, the robot uses 3 motors and 3 commercial omnidirectional wheels of 6cm placed in form of a tricycle. This robot is able to move omnidirectionally and avoid obstacles. This version was equipped with 1 gripper, 2 IR reflective optosensors, and 1 IR emitter and 1 IR receiver for communication, whose range is of approximately $1.20m$.

The design and building of a gripper was a very important challenge in the building of Mini-trans due to the precision required in a servomotor to operate the gears that open and close the gripper. In order to design and build an efficient gripper for modular robots, an important number of issues should be considered. The form and size of the assembling devices, the length of opening, the resistance and position where the gripper will be fixed, are for instance, key issues to be considered. Five grippers were designed and built. The definitive gripper of Mini-trans is able to hold another robot through one of the three cuts located on the third plate of each module of Mini-trans. This way of assembling does not require the precision of a pin-hole or other specific connection devices. Figure 2 shows one module of Mini-trans (a), its gripper (b) and a connection between two modules (c).



(a) (b) (c)

Fig. 2. Module of Mini-trans system (a). Each module is equipped with one gripper (b). Connection between modules (c)

3.2 Individual Capabilities

To summarize, each module of Mini-trans system is a mobile and autonomous robot able to send and receive messages, connect and disconnect to other modules at any of its attachment points. These modules are also able to move omnidirectionally and detect collisions around themselves. Finally, robots are able to identify a black circle on the floor that represents an objective region (see Figure 5 (c)).

3.3 Collective Capabilities

The modules of Mini-trans system have collective capabilities such as communication, self-assembling and coordinated motion that are described below.

Communication. The modules can communicate sending and receiving different IR coded signals. While sending a signal, a robot is unable to receive any signal, and vice versa. The messages used by robots are "connect", "disconnect", "ready", "wait", "stop" and "move".

Self-assembling. Each module is able to assemble another through one of the three cuts on the third plate. When one module receives the message "connect" from another module, the module closes its gripper. Once the gripper is closed, the module detects that the assembling was done. Then, the module communicates to another module that the connection was successful. On the other hand, when the message "disconnect" is received, the module opens its gripper and moves away from the other module and notifies that the disconnection was successful. Two or more connected modules can assemble another one, making then a linear formation.

Coordinated motion. When two or more modules are connected in linear formation, they can move keeping this formation. The module that is connected to another one is considered a `follower module`. In contrast, the module to which a module is connected is considered a `leader module`.
When modules are connected, the `leader` communicates the `follower` that the collective movement will start. Then, the `follower` moves forward until the gap between itself and the `leader` is full up. If the `leader` collides with an obstacle, it stops and communicates "disconnect" to the `follower`. `Follower` disconnects from the `leader` and moves away from it.

4 Experiments and Results

In this section, some experiments designed to measure the performance of Mini-trans are presented. These experiments were conducted using two modules of the Mini-trans system.

The experiments were conducted in a $1.5m \times 1.5m$ area. Both robots started at the same time. The `leader` robot searched a goal position indicated by a black circle painted on the floor while the `follower` robot waited for a signal. When this signal was perceived, the `follower` robot went towards the `leader` robot which was waiting. When the `follower` robot reached the `leader`, it informed its arrival to the `leader`. Then, the `leader` robot requested the `follower` to connect to it. The `follower` connected and notified `leader` that its connection had been done. Then, `leader` requested `follower` that motion must start and both robots started to move keeping the connection until an obstacle was perceived by `leader`. When the `leader` collided against an obstacle, it stopped and requested `follower` a disconnection. When `follower` received the message "disconnect", it disconnected from `leader`, moved away and notified to the `leader` that disconnection had been done. Then, both robots continued their exploration individually. The control of both robots is illustrated in Figure 3.

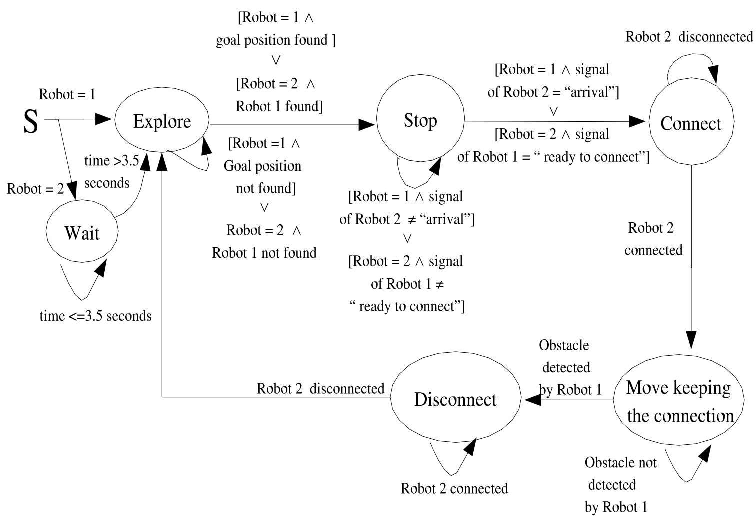


Fig. 3. Automaton that summarizes the general behavior of Mini-trans system. Behaviors such as **Wait**, **Explore**, **Stop**, **Connect**, **Move keeping the connection** and **Disconnect** are constantly executed. A robot may take one of two roles: `leader` (Robot 1) or `follower` (Robot 2). Robot 1 is the robot which waits for Robot 2 to connect to it. **Wait** consists in staying immobile during 3.5 seconds. **Explore** consists in moving within the environment until the goal position, an obstacle or robot 2 is found. **Stop** consists in finishing any activity. **Connect** is the process applied to connect to another robot using the gripper. **Move keeping the connection** consists in the movement executed by two connected robots until an obstacle is found by `leader`. Finally, **Disconnect** is the process applied to disconnect the modules of the system. **Connect**, **Move keeping the connection** and **Disconnect** require the exchange of messages and the waiting of certain time to send or receive a signal

The time that robots spent doing each behavior in 10 1-minute experiments is shown in Table 1.

Figure 4 illustrates the average time taken by the behaviors that were programmed, as well as the average time when robots exchanged messages and executed connection and disconnection. As this figure shows, robots are able to swap from individual action (**explore**) to collective actions (**connect**, **move keeping the connection** and **disconnect**) and to individual action again.

Table 1. Time in seconds spent by two robots, R1 and R2, during 10 tests. Robots had to go towards a goal position, connect, move connected, disconnect when an obstacle was perceived, and continue to explore individually during a constant time. Second robot (R2) had to wait 3.5 seconds before to start its operation

		Behaviors							
No. test		Wait	Explore	Stop	Connect	Move keeping the connection	Disconnect	Explore	Duration of the experiment (seconds)
1	R1	0.00	3.80	4.60	10.03	12.83	10.48	10.00	
	R2	3.50	4.91	3.26	10.81	12.20	13.98	10.00	58.66
2	R1	0.00	3.57	4.38	12.18	12.04	9.27	10.00	
	R2	3.50	4.97	4.14	13.34	9.15	13.98	10.00	59.08
3	R1	0.00	4.21	6.24	16.28	11.55	8.51	10.00	
	R2	3.50	3.94	10.32	14.86	8.72	13.98	10.00	65.32
4	R1	0.00	3.48	4.78	9.13	11.45	9.48	10.00	
	R2	3.50	2.75	4.90	10.43	11.18	13.97	10.00	56.73
5	R1	0.00	3.47	5.50	9.95	12.42	8.42	10.00	
	R2	3.50	5.58	4.30	11.07	9.53	13.98	10.00	57.96
6	R1	0.00	3.34	3.62	9.11	12.76	9.48	10.00	
	R2	3.50	4.13	6.26	10.30	11.57	13.97	10.00	59.73
7	R1	0.00	3.45	4.52	10.13	12.86	8.52	10.00	
	R2	3.50	4.47	3.07	12.96	10.14	13.97	10.00	58.11
8	R1	0.00	3.37	3.62	10.01	12.58	8.51	10.00	
	R2	3.50	3.90	7.40	10.45	11.80	13.98	10.00	61.03
9	R1	0.00	3.45	3.80	9.12	12.75	8.52	10.00	
	R2	3.50	4.31	4.14	10.29	9.95	13.97	10.00	56.16
10	R1	0.00	3.30	3.47	9.53	12.63	8.49	10.00	
	R2	3.50	3.53	3.24	12.12	9.76	13.98	10.00	56.13
$\bar{x}$	R1	0.00	3.54	4.45	10.55	12.39	8.97	10.00	
	R2	3.50	4.25	5.10	11.66	10.40	13.98	10.00	58.89

Figure 5 shows a sequence of snapshots taken during an experiment where robots were controlled by the automaton illustrated in Figure 3.

5 Conclusion and Perspectives

In this paper, the progress in the development of a multi-robot-system that consists of two modules with self-assembling capabilities has been described. The existing self-reconfigurable robots involve, as this paper has reviewed, considerable resources in the number of sensors and actuators, as well as in the

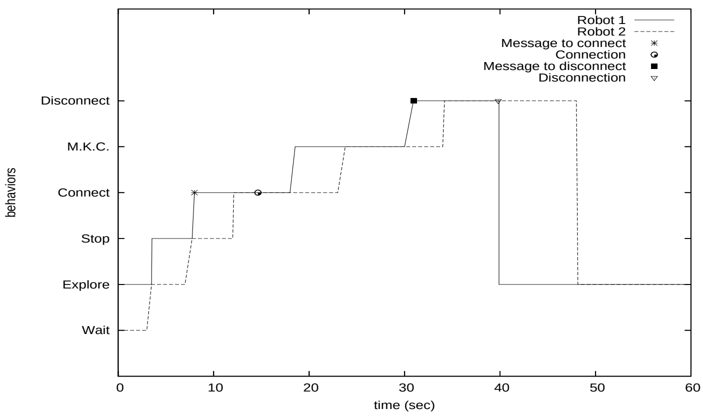
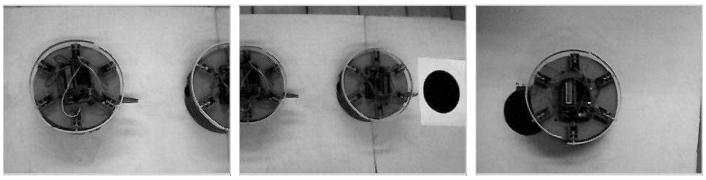
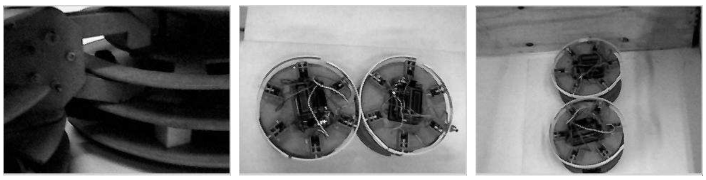


Fig. 4. Average duration of behaviors and average time in which robots exchange messages during the experiments described in Table 1. The alternation of behaviors illustrates the synchronization in the robots operation. M.K.C. means move keeping the connection



(a) (b) (c)



(d) (e) (f)

Fig. 5. Experiment divided into 6 steps: Wait (a), Explore (b), Stop (c) Connect (d), Move keeping the connection (e), Disconnect (f). The `leader` robot finds a goal position indicated by a black circle. When this position is found, the `leader` stops and waits for the `follower` robot. The `follower` robot waits 3.5 seconds to start, and then searches for the `leader` robot. When the `follower` finds the `leader` robot, both robots connect their bodies and move coordinating their movements. When the `leader` robot finds an obstacle, it stops and they disconnect their bodies. In order to coordinate their actions, robots exchange messages such as "ready to connect", "successful connection" and so on through their IR sensors

electronics to control them. Our prototype is much simpler compared with these systems. Various tests have been made in order to select and validate experimentally the simplest materials that are required to build a mobile reconfigurable multi-robot system.

The development of the Mini-trans system is in progress. Future work will focus on the design and implementation of more complex coordination mechanisms to enable robots to adapt their arrangement to the form of an object they are pushing, as well as to cope with obstacles when they are moving in assembled form. A third module is almost finished and ready for integration into the Mini-trans system.

Acknowledgments

We would like to thank Nancy Keranen and Blanca Téllez for the extensive proofreading done. The first author is supported by CONACyT, contract number 181509.

References

1. Yim, M., Zhang, Y., Roufas, K., Duff, D., Eldershaw, C.: Connecting and Disconnecting for Chain Self - Reconfiguration with PolyBot. IEEE/ASME Transactions on Mechatronics, Vol. 7, No. 4 (2002) 442- 451
2. Brown, H. B., Weghe, J. M. V., Bererton, C. A., Khosla, P. K.: Millibot Trains for Enhanced Mobility. IEEE / ASME Transactions on Mechatronics, Vol. 7, No. 4 (2002) 452 - 461
3. Kamimura, A., Yoshida, E., Murata, S., Kurokawa, H., Tomita, K., Kokaji, S.: A Self-Reconfigurable Modular Robot (MTRAN)- Hardware and Motion Planning Software. Distributed Autonomous Robotic System 5, Springer-Verlag (2002) 17-26
4. Fitch, F., Rus, D. Vona, M.: A Basis for Self-Repair Robots Using Self-Reconfiguring Crystal Modules, Proceedings of Intelligent Autonomous Systems, Dartmouth College, Massachusetts Inst. of Technology, USA (2000) 903-910
5. Mondada, F., Guignard, A., Bonani, M., Br, D., Lauria, M., Floreano, D.: SWARM-BOT: From Concept to Implementation. Proceedings of the International Conference on Intelligent Robots and Systems, IEEE Press (2003) 1626-1631
6. Kawakami, A., Torii, A., Motomura, K., Hirose, S.: SMC Rover: Planetary Rover with Transformable Wheels. Bruno Siciliano (ed): Experimental Robotics VIII. Advanced Robotic Series series, Springer (2003) 498-506

Biofeedback Agents for Electromyocontroled Teleoperated Robots

Sandra Díaz, Carlos Nieto-Granda, Marissa Díaz, Ricardo Swain-Oropeza

Mechatronics Research Center (CIMe)
Tecnológico de Monterrey - Campus Estado de México (ITESM-CEM)
Carretera Lago de Guadalupe Km. 3.5, Atizapán de Zaragoza,
Estado de México, 52926, México
{A00455576, A00461680, dmarissa, rswain}@itesm.mx

Abstract. Robot navigation and manipulation in unknown environments for non-autonomous applications is entirely dependant on the human controller and the data provided to control the robot. This task is complex and requires the development of specific skills and almost always alienates the human controller because it is linked to an interface that does not allow extending the user presence where the action is actually developing. On this paper we suggest a mixed technique to join virtual reality (VR) hardware and emulation techniques with biofeedback input to provide a suitable platform to tele-navigate a robot and an agent architecture that improves telecommunication and teleoperation. This approach suggests that it is possible to create intuitive and self-explained applications for robot navigation on which the users can explore the environment as if they were where the robot is and trough biofeedback, understand this environment better and extend the controllerś senses trough it. A biofeedback agent at the usersś terminal indicates the status of the teleoperated robot and using a hierarchical finite state machine it works as a link between the bare electronic response to the movement of the muscles and the robot movements and reactions.

1 Introduction and Previous Work

Robot teleoperation has been deeply studied by many areas of applied sciences like space exploration and medical surgery between many others. All these applications need a high degree of precision and interactivity between the operator and the remote site, but this represents an enormous challenge. [1] Many robot teleoperation systems have been developed based on visual feedback but they are strongly dependant of the transmission speed. This implicit delay can obstruct the biofeedback process because this process requires immediate correlation between cause (anoperator's action) and efect (change in the remote environment).

When we want teleoperated robots to interact with their environment in a more natural way, as if they had an animal instinct we think our implementation is capable of giving the robot an extra fluidness in its interaction. We can think of the specific case of soccer playing aibo robots, these robots act autonomously

© A. Gelbukh, R. Monroy. (Eds.)
Advances in Artificial Intelligence Theory
Research on Computing Science 16, 2005, pp. 243-252

but an intelligence algorithm can be developed to transmit human movements and decisions over specific conditions and events into the movement algorithm at a learning state during programming. This would give autonomous robots the ability to execute movements referred to a human action autonomously during a game. This can also be implemented in other applications like searching and exploration while dealing with specific conditions that can detriment the outcome of the robot's activity.

In the following sections we will describe the design and implementation of a complete human-robot interface designed to improve the interaction between an operator and a device. We describe the bases used to develop a biofeedback agent, which links the operator and the robot. In future work these agents will be able to use an expert systemś database to learn and modify a feedback loop in order to achieve a better execution of the activity and allow autonomous completion of specific expert routines.

First we will describe biofeedback, biofeedback agent and electromyocontroller. Then we will talk about our implementation and current results.

2 Biofeedback

Feedback can be defined as the response in a system, molecule, cell, organism or population that influences the activity or productivity of the system. To understand this concept from the biological point of view we can extend this concept and form a new one: Biofeedback, which is in essence the control of a biological reaction by the means of the products of the action. These products represent the information given instantaneously to an individual by a physiological process taking place on himself.

By monitoring the data that reflects the state of the physical conditions of an individual as temperature, pressure, pulse, etc. and retro feed the person by a monitor of level, light or sound the individual can modify this activity regardless that this activities are ruled by the nervous system in an involuntary way.

Now, lets define biofeedback, with biofeedback. We learn about the external world and our internal processes and it is the key to link feedback with kinesthetic sensation. Neal E. Miller developed a theory about the control of the involuntary functions of the body in the early 60s. By the means of training these behaviors, several illnesses as high blood pressure, stress, phobias, etc. can be controlled and treated.

We learn about the world trough biofeedback, we control our idea of the world by sensing it and acting to change it with notion of the effect we want as a result. There have been many clinical cases where there is a change in the perception and even in the interpretation of these signals of the external world by modifying the way people receives that information.

Robots, just like humans, need to interact with their environment and compute the information they receive in order to perform a certain task. Lets compare the robot perception with the human perception. Feedback can be defined as a response in a system that influences the activity or productivity of the system itself, in robot control feedback is needed to determine the external conditions of the robot and it is interpreted and provided by the sensor fusion array. Is explained in [2] that it is difficult for a robot to correctly respond to a given environment so they integrate robot sensors and effectors to enable the robot or a group of robots to provide a desired response.

When we talk about biological entities as cells, beings or populations the term used is Biofeedback, which is in essence the control of a biological reaction by the means of the products of the reaction. Integrating the information brought by our senses seems to be a very difficult task, but the brain can join all this information to give us an idea of what is near us. Biofeedback does not need to involve an ECG (Electro Cardio Graph) or EMG (Electro Myo Graph), for example, a mirror is a perfectly good biofeedback device for many aspects of gait retraining. Most of the records used in biofeedback are taken from the surface of the skin. The information recorded by surface sensors can be sent to a computer for processing and then displayed; in our case the biofeedback system will be provided by some muscle contraction sensors that will connect the user with the robot.

So, we can use Biofeedback providing real time information from psycho physiological recordings about the levels at which physiological systems are functioning Fig. 1. In this case we can use our own idea of position and direction to retrofeed the loop between the teleoperated robot and the human controller and get the correct kinesthesia between the robot and the human controller.

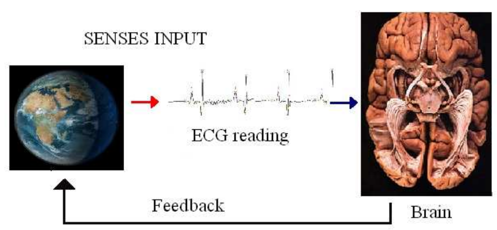


Fig. 1. Biofeedback process

Now we will move to the problem of motion control and kinesthetic sense compliance. First lets define kinesthesia as the sense of motion, the perception of body position and movement including muscular tensions, this classic definition is given by its greek roots kinesis (to move) and (esthesia) to perceive but some psychiatrists define it in a more complex manner considering it one of the somatic senses, (the somatic senses, also known as the somatosensory systems, include the skin senses of touch, temperature, and pain), responsible for the sensing of movement and the one that gives the relative position of objects and one self in a given place to the brain.

Kinesthesia, then, can be defined as the sense that tells the brain where the parts of the body are relatively to one another and where in the space we are. It is important to clarify that the vestibular system is the one that tells the brain about the position of the head in space not the kinesthetic one.

It is, then, possible to interconnect the human controller and the teleoperated robot in such a way that a biofeedback loop is attained. This biofeedback loop can be used to extend the senses and specifically the kinesthesia of the controller by reflecting the environment perceived by the robot into the perception of the human operator.

3 Biofeedback Agent

Now we will define some concepts to use them in the following section, first what is a biofeedback agent? We know so far that biofeedback is the process of control of a biological reaction by the means of seeing how it changes trough time; a biofeedback agent is the one that channels the biological information back to the source. It is an active agent with the potential for inducing both positive and negative changes in the interaction between the human operator and the robot. We define this biofeedback agent as both, a linking element that feedbacks the two edges of our system by signaling status levels to the user and the robot about the environment of the other, and as a control, decision and learning agent which can acquire information and modify the execution of the activity under certain constraints, a computational agent with adaptability capacities that work as medium to improve biofeedback. This biofeedback agent is in charge of merging both edges in the perception, sensation, action and interaction aspects.

4 Electromyocontrol

We can describe concept of an electromyocontroller as a human-computer interface built to sense the electrical activity in the muscles of a human user and traduce those signals into a pattern that can be analyzed and used for the purpose of controlling the execution of an activity by a certain device, in this case, the movement of an teleoperated Aibo robot. The objective of this electromyocontroller is to give the human operator the ability to manipulate the robot by moving his own muscles and to build a complete biofeedback interaction channel. The use of this type of controller enables us to give the robot humanlike decision taking and movement routines that the biofeedback agent can use to learn how to improve the execution of the activity.

As the fundamental concept behind the electrophysiological study of the human body is the detection of potentials in excitable cells, we decided to implement electromyosensors to detect the potentials generated at a certain muscle in the body of the human operator. These electromyosensors conform the recollection phase of our interface and integrate several elements in order to produce a detectable 5 Volts step output signal as the result of the registry of the pure myoelectrical signal generated at the muscle.

This type of myosensors has been previously used to build monitors of the human behavior as intent detectors used to complete danger measures during the evaluation of a dangerous real-time situation in human-robot interaction [3] [4] and to conform other systems like psychological therapy approaches.

5 Teleoperation and Electromyocontrol

To feel and see the changes in the environment where the robot is, as experienced by the sensors of the robot will make the navigation close looped giving the idea to the user that is the one exploring the distant location. To achieve that, we need to submerge the human controller into the reality of the robot, providing suitable controls and visual feedback to attain the sense of position and orientation from the robotś point of view. In the application shown in this paper we use the video stream provided by the robotś camera, a 3D model of the robot (local model) and intuitive muscle motion sensors at the userś side to control the reactions of our target. As we can see in the work of [5] [6] [7] [8] teleoperation has been successfully used in applications that require extreme precision successfully.

The myosensors give the user the idea of a really moving inside the environment because the controller relates movement directly with movement, he/she can see the result of their actions by watching the changes in the virtual reality environment where the 3D model exists and finally, the operator has a perspective of the remote scenario from the robotś point of view as seen in the camera stream. All these elements interlace to modify the perception of the user and change the way the operator senses the remote environment.

6 Implementing Biofeedback AGENTS

Hierarchical finite state machines, as traditional finite state machines, consist of states and transitions between states. The big difference is that HFSMś allow mixed states where there can be shared variables, functions and even transitions providing a higher level of abstraction impossible in common FSMś. These capabilities make them more efficient and a HFSM can need only the half states than its equivalent FSM.

We take lot of care on defining the kind of agentś brain so we can define a correct HFSM which describes the reactive motivation and objective that will allow the user to get a more adequate idea of the status of the remote robot, allowing the agent to be a biofeedback path ensuring correct reaction and interaction and avoiding problems surging from localized motion control, network latency and time delays.

We need to correlate the real movement of the AIBO with our biofeedback agent movements to describe on real time the interactive reactions of the robot allowing a realistic interaction. In this case we give our agent a 3D modeled body to allow instantaneous visual feedback, allowing reactive action to the interaction and make corrections and apply controls.

On this case we have to design the reactive control of a graphic agent equipped with external sensors that allows the identification of the current state which will depend on the user actions, the AIBO status, the interpretation of the AIBO environment and the previous state.

We want this agent to be capable of giving the correct controls to our AIBO robot, to achieve a more natural interaction of the AIBO in its environment, so the agent must acquire a reactive behavior that will act as the animal instinct provided by the input of the user and translated by the agent trough behavioral patterns into AIBOs movements.

There are many ways to implement a HFSM witch supports behavior models, one way to do it is to use HGSs to represent the behavioral unit or using PaT-Nets(Parallel Transition Network) [9] [10], but using behavioral primitives in our HFSM description optimized worked to our propose see Fig.2.

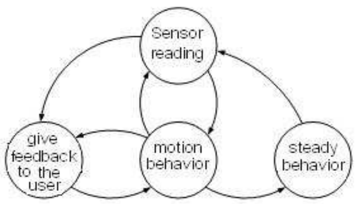


Fig. 2. Biofeedback process

As we can see on Fig. 3 our system consists of two main phases, the first one consist in capturing the action-reaction scheme of the teleoperation of the robot in an specific environment, the second one uses the motion-time tables to describe a series of robot controls that can be used to generate autonomous programs to solve specific tasks when there is no user connected to the system.

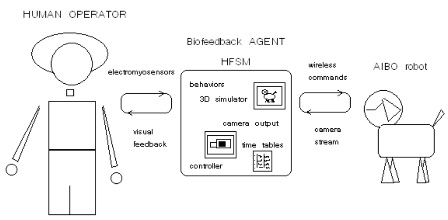


Fig. 3. Biofeedback Agent Simulation

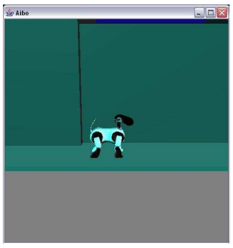


Fig. 4. Virtual Aibo

Fig. 5. Real Aibo

7 Results

We tested the operation of the biofeedback loop system we have described during the previous sections using two human operators and an AIBO robot as the teleoperated device. We achieved an important degree of involvement between the operators and the robot but the response time of the system was not good enough (around 4 seconds). To ensure biofeedback we need response time under 1 second. Here our agent plays a vital role covering the missing time space commands, returning faster visual response to the user improving the biofeedback mechanism and saving operatorś responses and controls executed on the robot on time tables. With a good amount of recorded behaviors and control sequences our agent can provide a more natural and better response to the actual reaction of the robot to the controls and foresee commands to enhance the navigation system.

We made our first experiment by exploring a local environment where the operators could have direct visual biofeedback of the movements of the robot.

Our next approach involved a remote location where the visual biofeedback elements took critical importance. The actions of the operators were successfully interpreted by the biofeedback agent, allowing a good performance in the exploration of the remote environment. The operators showed anxiety and uncertainty at the beginning while trying to communicate their commands through the myosensors, but the biofeedback elements helped reduce these emotions with time. The level of immersion in the remote location was acceptable but the use of other biofeedback elements, which interact with other senses different from sight, might increment greatly the operatorś awareness of the elements in the remote site.

Muscle	**Command**
Right Biceps Operator 1	Front movement / Stop front movement
Left Biceps Operator 1	Side movement left / Stop side front movement left
Right Biceps Operator 2	Side movement right / Stop side movement right

Table 1. Actions and Commands.

The previous table describes a model situation in which the operator commands the AIBO robot to move forward, stop, initiate forward again and move left.

Time	**Command**	**Biofeedback agent correction**	**Biofeedback agent's action to reflect condition on simulation**	**Source of action**
4 sec	Front movement	None	Initiate 3D model's front movement	Operator
10 sec	Stop front movement	Smooth transition between real and simulated stop.	Wait for Robot real stop time	Operator
11.5 sec	Stopped	Smooth transition between real and simulated stop.	Initiate 3D model's stop front movement	Robot
13 sec	Front movement	None	Initiate 3D model's front movement	Operator
16 sec	Side movement Left	Stop front movement Smooth transition between real and simulated direction change.	Wait for Robot real direction change time	Operator
17 sec	Stop front movement	Smooth transition between real and simulated direction change.	Wait for Robot real stop time	Agent
17.5 sec	Stopped	Smooth transition between real and simulated direction change.	Initiate 3D model's stop front movement	Robot
18.3 sec	Side movement Left	None	Initiate 3D model's side movement left	Agent

Fig. 6.

The compensations made by the biofeedback agent ensure that the simulation reflects accurately the user commands and the robotś real state, making transparent the delay times between the execution of commands and absorbing the intermediate steps required to execute them according to the robotś constraints.

The operator tends to think of less constrained movements by idealizing the robot as an holonomic device but this is not true, so the agent must fill the void between the desired movement and the actual movements required to achieve de robotś preferred configuration.

8 Conclusions

On this paper we have described how to use biofeedback to improve navigation in teleoperated robots, by enhancing the kinesthetic feeling to create better and faster responses to changes in the environments.

To measure how this navigation is improved we have to generate goal based excercises which rely on the human-controller's reactions and actions and measure time using a chronometer, we have to emulate diferent scenarios and tasks to see the efectiveness of our method and use virtual reality hardware to build a complete biofeedback loop between the robot and the user.

Using biofeedback to navigate remote robots can be useful in applications where the human user instinct is basic to correctly scan the area, in other words, when the place of exploration is completely unknown and has potential dangers and many bifurcations and obstacles.

Fig. 7. Human controlling an Aibo

A normal camera video can provide the necessary data to navigate remotely but it can not give the user the kinesthetic feedback needed to improve reactions where an immediate action is important.

9 Future Work

We want to extend our work using other methods to extend other senses of the user besides the kinesthetic one. We plan to make electromyo-stimulating modules to transmit to the user physically the existence of boundaries in the remote location detected by proximity or contact sensors in the robot. In further work it may be possible to coordinate multiple extensions for different senses using different biofeedback channels.

References

1. T. Nakanishi H. Hori T. Watanabe H. Kramer B. Mitsuishi, M. Watanabe. A tele-micro-surgery system across the internet with a fixed viewpoint/operation-point. In *IROS '95: Proceedings of the International Conference on Intelligent Robots and Systems-Volume 2*, page 2178, Washington, DC, USA, 1995. IEEE Computer Society.
2. Matson and DeLoach. Integrating robotic sensor and effector capabilities with multi-agent organizations. In *Proc. International Conference on Artificial Intelligence*, 2004.
3. Carnegie Mellon University. Tekkotsu. See: http://www-2.cs.cmu.edu/˜ tekkotsu/.
4. Carnegie Mellon University. Sony aibo legged robot team. See: http://www-2.cs.cmu.edu/˜robosoccer/legged/.
5. NHS. Robot sister mary. See: http://www.st-marys.nhs.uk.
6. Korb W. The worldś first robot milling procedure of complex trajectories on the head. In *Proc. IEEE International Conference on Robotics and Automation*, 2005.
7. J. Yuri S. Wang, J. Ding and Q. Li. Robot microsurgery system. In *Proc. IEEE International Conference on Robotics and Automation*, 2005.
8. Yumi Iwashita. Fast 2d-3d registration for navigation system of surgical robot. In *Proc. IEEE International Conference on Robotics and Automation*, 2005.
9. Tsukasa Noma, Liwei Zhao, and Norman I. Badler. Design of a virtual human presenter. *IEEE Comput. Graph. Appl.*, 20(4):79–85, 2000.
10. Justine Cassell, Catherine Pelachaud, Norman Badler, Mark Steedman, Brett Achorn, Tripp Becket, Brett Douville, Scott Prevost, and Matthew Stone. Animated conversation: rule-based generation of facial expression, gesture & spoken intonation for multiple conversational agents. In *SIGGRAPH '94: Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pages 413–420, New York, NY, USA, 1994. ACM Press.

Route Planning

Extending and applying PP language: An answer set planning problem language

Claudia Zepeda[1,2],Mauricio Osorio[1], Christine Solnon[2], and David Sol[1]

[1] Universidad de las Américas, CENTIA, Sta. Catarina Mártir, Cholula, Puebla, 72820 México
{josorio,sc098382,sol}@mail.udlap.mx,
[2] LIRIS UMR 5205 CNRS, Université Lyon 1 and INSA de Lyon, 43 bd du 11 novembre, 69622 Villeurbanne cedex, France
{claudia.zepeda,christine.solnon}@liris.cnrs.fr

Abstract. A useful approach for expressing preferences, temporal preferences and multidimensional preferences over plans is language $\mathcal{PP}$. In this paper we give an overview of a real application of this language: Evacuation planning. Moreover, we remark on some limitations of $\mathcal{PP}$ when the size of the preference is not known or is long. Then we extend $\mathcal{PP}$ by parametric preferences and we show how this extension can overcome these limitations. Also, we give a review about how $\mathcal{PP}$ can inherit all the working framework of Linear Temporal Logic to express preferences.
Key words: Preferences, Answer Set Planning, Linear Temporal Logic.

1 Introduction

Using Answer Set Programming (ASP) [4] makes it possible to describe a computational problem as a logic program whose answer sets correspond to the solutions of the given problem. Currently, there are several answer set solvers, such as: DLV[1] and SMODELS[2]. The objective of our work is to investigate and evaluate the applicability of ASP to represent disaster situations in order to give support in definition of evacuation plans. The goal of which is to find actions to perform to put out of risk the population living in the disaster zone.

Given a planning problem expressed in an action language, it is possible to define an answer set encoding of it [2]. Then, it is possible to obtain the solution of the planning problem (the plans) from the answer sets of its answer set encoding [2]. However, given a planning problem we may obtain a large number of solutions. In this case, we need to specify preferences to select the "best" of those plans. To specify such preferences among feasible plans, [9] introduced a new language named $\mathcal{PP}$. We consider language $\mathcal{PP}$ because it allows us to express temporal preferences over plans and at different levels: the preferences in $\mathcal{PP}$ are based on the occurrence of an action in a plan, on the fluents that define

[1] http://www.dbai.tuwien.ac.at/proj/dlv/
[2] http://www.tcs.hut.fi/Software/smodels/

© A. Gelbukh, R. Monroy. (Eds.)
Advances in Artificial Intelligence Theory
Research on Computing Science 16, 2005, pp. 255-264

a state in the plan, on the moment when an action occurs or a fluent holds in a state or on some combination of all them. The preferences representing time are expressed using the temporal connectives *next, always, until* and *eventually.* We think that in evacuation planning it is very useful to express preferences in terms of time. In particular they are useful, when it is not possible that evacuees follow the pre-defined evacuation plan since part of the route becomes blocked. Then, they should follow an alternative evacuation plan.

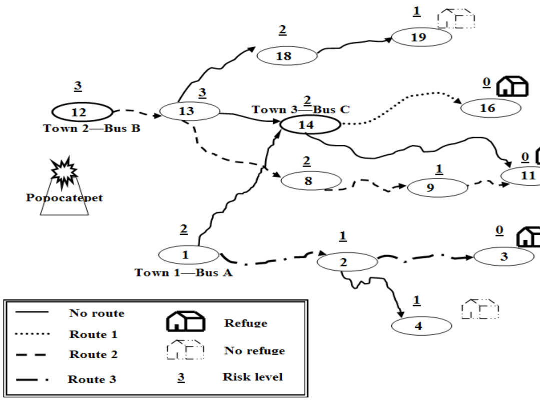


Fig. 1. Three evacuation routes: A short example.

For instance, let the directed graph in Figure 1 be a short representation of three evacuation routes in a particular zone. We define a preference about bus B, denoted as φ, to express: it is preferred that bus B travels by evacuation route 2 (path: 12, 13, 8, 9 and 11) *until* it arrives to its assigned refuge (node 11). Let us notice that preference φ is specifying a disjunction consisting of the different options that bus B has to travel by arcs belonging to evacuation route 2:

$$\mathbf{until}(\mathbf{occ}(travel(busB, 12, 13, 1)) \vee \mathbf{occ}(travel(busB, 13, 8, 1)) \vee \mathbf{occ}(travel(busB, 8, 9, 1)) \vee \mathbf{occ}(travel(busB, 9, 11, 1))\ ,\ position(busB, 11)).$$

However, let us suppose evacuation route 2 has a large number of arcs. Then, in order to express φ in a similar way we would have to specify a large disjunction consisting of all arcs in the new evacuation route 2. Moreover, if this evacuation route change then we have to change also the preference. Despite in [9] it is indicated that in $\mathcal{PP}$ fluents and actions with variables are shorthand representing the set of their ground instantiations, the idea of using fluents and actions with variables is not enough to represent this type of problems.

Hence, there are some problems that can not be expressed in $\mathcal{PP}$ in a simple and natural manner. In order to have a natural representation of these kind of preferences and inspired by [7], in this paper we define $\mathcal{PP}^{par}$ language an extension of $\mathcal{PP}$ language where propositional connectives and temporal connectives allow us to represent compactly preferences having a particular property. For instance, a natural and compact representation of preference φ using a *parametric or* would be:

$$\mathbf{until}(\bigvee\{occ(travel(busB, I, F, 1)) : road(I, F, 1)\}, position(busB, 11)).$$

As we have mentioned, we think that in evacuation planning it is very useful to express preferences in terms of time. Then, in order to illustrate the usefulness of $\mathcal{PP}^{par}$ language in evacuation planning, we shall consider the real problem of finding alternative evacuation routes in the risk zone of volcano Popocatepetl in Mexico.

Finally, since $\mathcal{PP}$ is useful to express preferences over plans where the satisfaction of these preferences depends on time, in this paper we present a review of the relationship between language $\mathcal{PP}$ and propositional Linear Temporal Logic (LTL) [6, 8]. We think that language PP could take advantage of the working framework of LTL to express preferences.

The rest of the paper is structured as follows. In Section 2, we introduce some fundamental definitions about planning problems and preferences. In Section 3, we present $\mathcal{PP}^{par}$ language. In section 4 we give an overview of a real application using $\mathcal{PP}^{par}$ language: evacuation routes in the risk zone of volcano Popocatepetl in Mexico. In Section 5 we review the relationship between language $\mathcal{PP}$ and LTL. Finally in Section 6, we present our conclusions.

2 Background: Planning problems and preferences

An action signature is a couple $\langle F, A\rangle$ where F is a set of fluents and A is a set of actions. A planning problem, over a signature $\langle F, A\rangle$, is a tuple $\langle T, I, G\rangle$ where $T \subseteq \mathcal{P}(\mathbf{F}) \times \mathbf{A} \times \mathcal{P}(\mathbf{F})$ is a transition function such that $(\sigma_i, a_j, \sigma_k) \in T$ means that action a_j allows one to go from state σ_i to state σ_k, $I \subseteq F$ is the initial state and $G \subseteq F$ is the goal. The solution of a planning problem $\langle T, I, G\rangle$ over a signature $\langle F, A\rangle$ is a plan or a sequence of actions $a_1, \ldots, a_n$ to achieve its goal G starting from the initial state I, i.e. there exists a sequence of states $\sigma_0, \sigma_1 \ldots, \sigma_n$ such that $\sigma_0 = I$, $\sigma_n = G$ and $\forall i \in [1; n]$, $(\sigma_{i-1}, a_i, \sigma_i) \in T$. The sequence $\sigma_0, a_1, \sigma_1 \ldots, a_n, \sigma_n$ where $\sigma_1, \ldots, \sigma_n$ are states and $(\sigma_{i-1}, a_i, \sigma_i) \in T$, $1 \leq i \leq n$ is called a *history* of the transition system T. A full description about action languages can be found in [5]. Given a planning problem expressed in an action language, it is possible to define an answer set encoding of it [2], denoted as $\Pi(T, I, G)$. Then, it is possible to obtain the solution of the planning problem (the plans) from the answer sets of $\Pi(T, I, G)$ [2].

Given a planning problem, we may obtain a high number of solutions. In this case, we need to specify preferences to select the "best" of those plans. To specify such preferences among feasible plans, [9] introduced a new language named $\mathcal{PP}$. We consider this language $\mathcal{PP}$ because it allows us to express temporal preferences over plans: the preferences in $\mathcal{PP}$ are based on the occurrence of an

action in a plan, on the fluents that define a state in the plan, on the moment when an action occurs or a fluent holds in a state or on some combination of all them. The preferences representing time are expressed using the temporal connectives *next, always, until* and *eventually.* The combination of them can be defined using three different classes of preferences:
—A *basic desire*, denoted as φ, is a $\mathcal{PP}$ formula expressing a preference about a trajectory with respect to the execution of some specific action or with respect to the states that the trajectory gets when an action is executed.
—An *atomic preference*, denoted as $\psi = \varphi_1 \lhd \varphi_2 \lhd \ldots \lhd \varphi_k$, is a formula that gives the order in which a set of basic desires formulas should be satisfied.
—A *general preference* is a formula based on atomic preferences.

The set of basic desires of language $\mathcal{PP}$ can be defined inductively by the following context-free grammar $G_{PP} := (N, \Sigma, P, S)$, such that $N := \{S\}$ is the finite set of non terminals; $\Sigma := \mathbf{A} \cup \mathcal{F}_F$ is the finite set of terminals ($N \cap \Sigma = \emptyset$) where $\mathbf{A}$ and $\mathcal{F}_F$ represent the set of actions of the planning problem and the set of all fluent formulas (propositional formulas based on fluent literals) respectively; $S \in N$ is the initial symbol of the grammar; and $P := \{S \longrightarrow p|\mathbf{goal}(p)|occ(a)|S \wedge S|S \vee S|\neg S|\mathbf{next}(S)|\mathbf{until}(S, S)|\mathbf{always}(S)|\mathbf{eventually}(S)\}$ is the finite set of productions or rules where $p \in \mathcal{F}_F$ and $a \in \mathbf{A}$.

Due to lack of space, we do not explain how to obtain the most preferred plan of a planning problem with respect to the different classes of preferences in $\mathcal{PP}$, however in [9] there are different alternatives to obtain them. Moreover, in [10] we show how we can obtain the most preferred plan with respect to an atomic preference using a simpler and easier encoding than the encoding proposed in [9].

3 $\mathcal{PP}^{par}$ language

Inspired in [7] we define $\mathcal{PP}^{par}$ language, an extension of $\mathcal{PP}$ language, where propositional connectives and temporal connectives allow us to represent compactly basic desires having a particular property.

3.1 Parametric basic desires

In $\mathcal{PP}$ fluents and actions with variables are shorthand representing the set of their ground instantiations. However, we need to specify when an action or fluent have variables. Let $\mathbf{F}^{vc}$ be the set of fluents with variables and/or constants. Let $\mathbf{A}^{vc}$ be the set of actions with variables and/or constants. A *desire set* B is $\{D : t_1, \ldots, t_n\}$ where D is (1) a fluent $f \in \mathbf{F}^{vc}$ or (2) a formula of the form $occ(\mathbf{a})$ where $\mathbf{a} \in \mathbf{A}^{vc}$ or (3) a formula of the form $goal(\mathbf{p})$ where $\mathbf{p} \in \mathbf{F}^{vc}$ and $t_1, \ldots, t_n$ is a conjunction of literals [3]. Let $\mathbf{B}$ be the set of desire sets. A *parametric and* is $\bigwedge B$ where B is a desire set. A *parametric or* is $\bigvee B$ where B is a desire set.

[3] A variable or a constant is a *term*. An *atom* is $p(t_1, \ldots, t_n)$ where p is a predicate of arity n and $t_1, \ldots, t_n$ are terms. A *literal* is either an atom a or the negation of an atom *not* a.

$G^{par} := (N^{par}, \Sigma^{par}, P^{par}, S)$ is the context-free grammar that defines the set of *parametric basic desires* where $N^{par} := \{S\}$; $\Sigma^{par} := \mathbf{A}^{vc} \cup \mathbf{F}^{vc} \cup \mathbf{B}$; $S \in N^{par}$ is the initial symbol of the grammar; and the finite set of productions or rules is P^{par} such that $P^{par} := \{S \longrightarrow p \mid \mathbf{goal}(p) \mid occ(a) \mid \bigwedge B \mid \bigvee B \mid S \wedge S \mid S \vee S \mid \neg S \mid \mathbf{next}(S) \mid \mathbf{until}(S,S) \mid \mathbf{always}(S) \mid \mathbf{eventually}(S)\}$ where $p \in \mathcal{F}_F$, $a \in \mathbf{A}$ as in language $\mathcal{PP}$ and $B \in \mathbf{B}$. For example, a parametric basic desire is:

$$\mathbf{until}(\bigvee\{occ(travel(busB, I, F, 1)) : road(I, F, 1)\}, position(busB, 11)),$$

where the desire set is: $\{occ(travel(busB, I, F, 1)) : road(I, F, 1)\}$.

3.2 Basic desire instantiation and satisfaction

Given a parametric basic desire φ of a planning problem $P = \langle T, I, G\rangle$ over a signature $\langle F, A\rangle$, let $C = F \cup A \cup BK$ denote the set of constants appearing in P where BK is the background knowledge of the problem (for instance the set of nodes and segments defining the directed graph could be part of the background knowledge). A *substitution* s is a mapping from a set of variables to the set C. Given a desire set $B = \{D : t_1, \ldots, t_n\}$, the *instantiation of set* B is the following ground set $B' = \{\{s(D) : s(t_1) \ldots s(t_n)\} \mid s \text{ is a substitution}\}$; B' is called *ground desire set*. A *ground instance* of a parametric basic desire φ is obtained if every desire set B in φ is replaced by its instantiation B'.

For instance, let φ be the parametric basic desire described in Section 1:

$$\mathbf{until}(\bigvee\{occ(travel(busB, I, F, 1)) : road(I, F, 1)\}, position(busB, 11)).$$

If we consider the directed graph in Figure 1 then, the ground instance of φ is the following:

$$\mathbf{until}(\mathbf{occ}(travel(busB, 12, 13, 1)) \vee \mathbf{occ}(travel(busB, 13, 8, 1)) \vee \mathbf{occ}(travel(busB, 8, 9, 1)) \vee \mathbf{occ}(travel(busB, 9, 11, 1)) \ , position(busB, 11)).$$

Given a history $\alpha = s_0 a_1 s_1 a_2 s_2 \ldots a_n s_n$ of a planning problem (see section 2) and φ a basic desire formula, in [9] is defined when α satisfies φ (written as $\alpha \models \varphi$). Since a parametric basic desire works as an abbreviation of a basic desire, given a parametric basic desire φ' and an history α it is enough to apply over the ground instance of φ' the definition of satisfaction for basic desires given in [9] to check whether α satisfies φ'.

4 Finding alternative routes in the risk zone of volcano Popocatepelt

Let us consider the real problem of finding alternative evacuation routes in the risk zone of volcano Popocatepetl in Mexico. Nowadays, "Plan Operativo Popocatepetl" office in Mexico (*POP office*) is responsible of assuring safety of the people living in the risk zone of the volcano in case of an eruption. For this purpose, POP office has defined ten evacuation routes. However, some hazards that can accompany volcano eruptions (mud flows, flash floods, landslides and rockfalls, etc.) can result on the blocking of the pre-established routes. The *alternative evacuation route problem* can be stated as follows: *There is a set of predefined evacuation routes for people living in the risk area. Evacuees should travel by these routes. In case of part of an evacuation route becomes inaccessible, then evacuees should search an alternative path. This alternative path can belong or not to another evacuation route. If it does not belong to an evacuation route then it should arrive to some point belonging to an evacuation route, to some refuge or to some place out of risk.*

Previously we have worked in this problem [4]. We have a detailed description of the problem. Also we have presented a partial solution to it using CR-Prolog [1], an extension of ASP with *consistency restoring rules*. Another partial solution to this problem shows how CR-Prolog programs can be translated into standard ordered disjunction logic programs as defined by Brewka [3] .

In this paper is given an overview of a more complete solution of the problem about finding alternative evacuation routes using language $\mathcal{PP}^{par}$.

We represent the network of roads between towns in the risk zone as a directed graph. This representation was created from an extract of our GIS database and contains real evacuation routes, towns (mostly in risk, but nearby towns not in direct risk are also included) and some additional segments that do not belong to any evacuation route, since these segments are necessary to obtain the alternative evacuation plans. We define a directed graph where nodes represent towns and evacuation routes are paths in the graph. Each segment is represented by `road(P,Q,R)` where `P` and `Q` are nodes and `R` is the route number. Segments with route number different of zero belong to some evacuation route. An exogenous action which causes `road(P,Q,R)` to become blocked results in a fact of the form `blocked(P,Q,R)`. The action `travel(P,Q,R)` allows to travel from `P` to `Q` if there is an unblocked segment of road from `P` to `Q`. We assumed that each action takes one unit of time. In particular, if we consider the directed graph in Figure 1 then we are considering three evacuation routes where the buses A, B and C travel from an initial location to the assigned refuge. Also we can define $\Pi(D, I, G)$ as follows:

```
% initial and final conditions
initially(position(busA, 1, 3)).
initially(position(busB, 12, 2)).
initially(position(busC, 14, 1)).
finally(position(B,N,R)) :- bus(B), node(N,R), refuge(N).
% fluents
 fluent(position(B,Q,R)) :- bus(B), node(Q,R).
 fluent(blocked (P,Q,R)) :- road(P,Q,R).
% actions travel by a segment of road
```

[4] We do not write the references of the work since it is not allowed write self-references in the paper.

```
 action(travel(B,P,Q,R)) :- bus(B),road(P,Q,R).
% Dynamic causal laws
 caused(position(B,Q,R),travel(B,P,Q,R)) :- bus(B),road(P,Q,R).
 caused(neg(position(B,P,R)),travel(B,P,Q,R)) :- bus(B),road(P,Q,R).
% Executability Conditions
 noaction_if(travel(B,P,Q,R),neg(position(B,P,R))):- bus(B),road(P,Q,R).
 noaction_if(travel(B,P,Q,R),blocked(P,Q,R)) :-  bus(B),road(P,Q,R).
```

Using $\mathcal{PP}^{par}$ we define the following parametric basic desires in order to define the associated atomic preference of this planning problem.

—*travelERass* to express that it is preferred that *always* buses travel by the evacuation route assigned by the government *until* they arrive to the refuge: $travelERass := \textbf{until(}\ \textbf{always(}\bigvee\{occ(travel(busA, I, F, 3)) : road(I, F, 3)\}\textbf{)}, position(busA, 3)\ \textbf{)}\wedge\ \textbf{until(}\ \textbf{always(}\bigvee\{occ(travel(busB, I, F, 2)) : road(I, F, 2)\}\textbf{)}, position(busB, 11)\ \textbf{)}\wedge\ \textbf{until(}\ \textbf{always(}\bigvee\{occ(travel(busC, I, F, 1)) : road(I, F, 1)\}\textbf{)}, position(busC, 16)\ \textbf{)}$

—*travelER* to express that it is preferred that *always* buses travel by roads belonging to some evacuation route and it is not important if they travel or not by its assigned evacuation route (*neq* denotes $\neq$): $travelER := \textbf{until(}\ \textbf{always(}\ \bigvee\{occ(travel(B, I, F, R)) : bus(B), road(I, F, R), neq(R, 0)\}\ \textbf{)},\ \bigwedge\{position(B, Fi) : bus(B), refuge(Fi)\}\ \textbf{))}$

—*arriveER* to express that it is preferred that buses travel out of the evacuation route assigned *until* they travel by an evacuation route to arrive at a refuge: $arriveER := \textbf{until (}\ \textbf{always(}\ \bigvee\{occ(travel(B, I, F, 0)) : bus(B), road(I, F, 0), \}\ \textbf{)},\ \textbf{until (}\bigvee\{occ(travel(B, I, F, R)) : bus(B), road(I, F, R), neq(R, 0)\}\ \textbf{,}\ \bigwedge\{position(B, Fi) : bus(B), refuge(Fi)\}\ \textbf{))}$

In a similar way we could express that *always* buses travel by a road out of an evacuation route until they arrive to any place with or without refuge (*arriveR*) or any other parametric basic desire.

A possible atomic preference ψ indicating the order in which the set of parametric basic desires formulas should be satisfied is the following:

$\psi = travelERass \lhd travelER \lhd arriveER \lhd arriveR$

The atomic preference ψ says that plans satisfying $travelERass$ are preferred, but otherwise plans satisfying $travelER$ are preferred, and so forth. Considering the set of segments of the directed graph in Figure 1 with no blocked segments, the most preferred trajectory w.r.t. ψ is:

time 1: travel(busB,12,13), travel(busC,14,16), travel(busA,1,2);
time 2: travel(busB,13,8), travel(busA,2,3);
time 3: travel(busB,8,9);
time 4: travel(busB,9,11).

We can see that this most preferred trajectory satisfies the parametric basic desire *travelERass* of the atomic preference ψ since all the buses travel by the evacuation route assigned by the government, exactly as *POP office* indicates. Now, if we consider the set of segments of the directed graph in Figure 1 but we add the initial condition $initially(blocked(1, 2, 3))$ to the program P (i.e. the first segment of road of evacuation route 3 is blocked). Then the most preferred trajectory w.r.t. ψ is the same for buses B and C but for bus A the route is $travel(busA, 1, 14)$ in time 1 and $travel(busA, 14, 16)$ in time 2. In this case, the most preferred trajectory satisfies the parametric basic desire *arriveER* of the atomic preference ψ, since bus A travels by a road out of any evacuation route until it arrives to node 14 of evacuation route 1 and then it get to a refuge.

5 The relationship between language $\mathcal{PP}$ and Temporal Logic

In this section, we review the relationship between language $\mathcal{PP}$ and propositional Linear Temporal Logic (LTL) [6, 8]. Since $\mathcal{PP}$ is useful to express preferences over plans where the satisfaction of these preferences depend on time, the goal of this review is to show how $\mathcal{PP}$ can inherit all the working framework of LTL. Then language PP could take advantage of the working framework of LTL to express preferences. In LTL all temporal operators are future time operators, meaning that at a given state one can only reason about the present and future states. Normally, in LTL is assumed that the set of time points is infinite, discrete an linearly ordered with a smallest element. However, in this paper we are interested in a LTL [8] where the set of time points is finite since plans of planning problems are finite that we will call $FLTL$.

5.1 Finite Temporal Logic

The language of $FLTL$ [6, 8] is given by the context-free grammar $G_T := (N, \Sigma, P, S)$ where $N := \{S\}$ is the finite set of non terminals; $\Sigma := \mathcal{V} \cup \{\bot, \rightarrow, \bigcirc, \wedge, \mathcal{U}\}$ is the finite set of terminals such that $\mathcal{V}$ is the set of atomic formulas and $(N \cap \Sigma \neq \emptyset)$; $S \in N$ is the initial symbol of the grammar; and $P := \{S \longrightarrow p|S \wedge S|S \rightarrow S| \bigcirc S|S\, \mathcal{U}\, S\}$ is the finite set of productions or rules where $p \in \Sigma$.

In order to omit superfluous parentheses, the priority order of the operators is established as usually. $FLTL$ formulas are denoted as $A, A_1, \ldots, B, B_1, \ldots$. The operator $\bigcirc A$, called *nexttime* operator, reads "A holds at time point immediately after the reference point (the present time)". The operator $A\, \mathcal{U}\, B$, called *until* operator, reads "A holds until B is true". Other operators can be introduced as abbreviations, e.g., $\wedge$, $\vee$, $\leftrightarrow$, $true$, $false$ as in classical logic; $\neg A$ for $A \rightarrow \bot$; $\Diamond A$ for $true\, \mathcal{U}\, A$; $\Box A$ for $\neg\Diamond\neg A$. The operator $\Diamond A$, called *eventually* operator, reads "There is a time point after a reference point at which A holds". The operator $\Box A$, called *always* operator, reads "A holds at all time points after the reference point".

The semantics of $FLTL$ is based on a *temporal structure* $\mathbf{K}$ that consists of a finite sequence $\{\eta_0, \ldots \eta_n\}$ of mappings $\eta_i : \mathcal{V} \rightarrow \{f, t\}$, the η_i are called *states*. η_0 is the *initial state*. The finite sequence of states formalizes the informal time scale; a state is a "time point". Every state is a valuation in the classical sence. For every temporal structure $\mathbf{K}$, every $i \in \mathbb{N}_0$ and every formula F, the truth value $\mathbf{K}_i(F) \in \{f, t\}$ is inductively defined , informally meaning the "truth value of F in state η_i " [6, 8]:

1. $\mathbf{K}_i(v) = \eta_i(v)$ for $v \in \mathcal{V}$.
2. $\mathbf{K}_i(A \wedge B) = \mathbf{t}$ iff $\mathbf{K}_i(A) = \mathbf{t}$ and $\mathbf{K}_i(B) = \mathbf{t}$.
3. $\mathbf{K}_i(A \to B) = \mathbf{t}$ iff $\mathbf{K}_i(A) = \mathbf{f}$ or $\mathbf{K}_i(B) = \mathbf{t}$.
4. $\mathbf{K}_i(\bigcirc A) = \mathbf{t}$ iff $i = n$ or $\mathbf{K}_{i+1}(A) = \mathbf{t}$.
5. $\mathbf{K}_i(A \,\mathcal{U}\, B) = \mathbf{t}$ iff $\mathbf{K}_j(B) = \mathbf{t}$ for some $i < j \leq n$ and $\mathbf{K}_k(A) = \mathbf{t}$ for every k, $i \leq k < j$.

In the axiomatization of temporal logic over finite sequences it is important to recognize the final state of the sequence. Moreover, for preferences about evacuation plans it is important to recognize the state where the goal is achieved. Then, the semantics for $\bigcirc A$ that we presented above has the property that $\bigcirc false$ is true at the final state [8]. However, we can define other two alternative semantics for $\bigcirc A$:
— $\mathbf{K}_i(\bigcirc A) = \mathbf{t}$ iff $\mathbf{K}_{i+1}(A) = \mathbf{t}$. This semantics has the property that $\neg \bigcirc true$ is true only at the final state. Also, this semantics is similar to the semantics of the operator $\mathbf{next}(\psi)$ of language $\mathcal{PP}$.
— $\mathbf{K}_i(\bigcirc A) = \mathbf{t}$ iff $\mathbf{K}_{i+1}(A) = \mathbf{t}$ for $0 \leq i < n$ and $\mathbf{K}_n(\bigcirc A) = \eta_n(A)$. In this semantics the truth value for $\bigcirc A$ at the final state depends on the true value of formula A. Then, the semantics of $\bigcirc A$ has the property that the final state of the plan is infinitely repeated. Using this semantics for $\bigcirc A$ we could abbreviate the operator $\mathbf{goal}(A)$ of language $\mathcal{PP}$ that reads "A holds at the final state" as follows: $A \wedge \neg \bigcirc true$. Moreover, we think that this semantics for $\bigcirc A$ could be the most suitable for evacuation planning, because of the fact that once we are in the final state this state remains without changes. For instance, once the evacuees have achieved the shelter assigned they will remain there.

5.2 Inheriting the $FLTL$ work framework to language $\mathcal{PP}$

In order to inherit all the $FLTL$ work framework to language $\mathcal{PP}$ we need to do the following: (1) transforming each history α of a planning problem into a finite temporal structure $\mathbf{K}_\alpha$; (2) transforming the basic desire formula φ into a temporal formula F_φ; and (3) obtain the truth value of $\mathbf{K}_i(F_\varphi)$.

Let P be a planning problem and $\alpha = s_0 a_1 s_1 a_2 s_2 \ldots a_n s_n$ a history of P. Let $\mathbf{F}$ be the set of fluents of P and $\mathbf{A}$ be the set of actions of P. Let $\mathbf{S}$ be the set of states in α, i.e., $\mathbf{S} = \{s_0, s_1, \ldots, s_n\}$. The transformation of the history α into a finite temporal structure $\mathbf{K}_\alpha$ is described as follows: First, we define a transformation function T of an action $a \in \mathbf{A}$ as follow: $T(a) := f_a$ where f_a is a fluent and $f_a \notin \mathbf{F}$. Also, we define a straightforward generalization of T over $\mathbf{A}$, the set of actions, as follows: $T(\mathbf{A}) = \{T(a) | a \in \mathbf{A}\}$. Then, the *finite temporal structure* $\mathbf{K}_\alpha$ consists of a finite sequence $\{\eta_0, \ldots \eta_n\}$ of mappings $\eta_i : \mathbf{S} \cup T(\mathbf{A}) \to \{f, t\}$. Every η_i is a valuation defined as follows: $\eta_i(f_{a_i}) = t$ iff $a_i \in \alpha$ and $\eta_i(s_i) = t$ iff $s_i \in \alpha$.

In order to transform a basic desire formula φ into a temporal formula F_φ, we only replace each occurrence of $occ(a)$ in φ for the fluent obtained from $T(a)$.

Finally, for every history $\alpha = s_0 a_1 s_1 a_2 s_2 \ldots a_n s_n$ and its finite temporal structure $\mathbf{K}_\alpha$, every $i \in \mathbb{N}_0$ and every temporal formula F_φ obtained from the basic desire formula φ, the truth value $\mathbf{K}_{\alpha i}(F_\varphi) \in \{f, t\}$ is inductively defined in the same way as it is defined for a formula in $FLTL$.

6 Conclusions

In this paper we give an overview of a more complete solution of the problem about finding alternative evacuation routes using language $\mathcal{PP}^{par}$. In particular we test the usefulness of $\mathcal{PP}^{par}$ finding alternative evacuation routes in the risk zone of volcano Popocatepetl in Mexico. Due to lack of space we only show an example with a short number of evacuation routes, however we have tested with a larger number of segments representing the real zone of volcano Popocatepetl. Also, we review the relationship between $\mathcal{PP}$ and Linear Temporal Logic. Since, it is useful to express preferences over plans where the satisfaction of these preferences depend on time, the goal of this review is to show how $\mathcal{PP}$ can inherit all the working framework of LTL. Then language PP could take advantage of the working framework of LTL to express preferences.

References

1. Marcello Balduccini and Michael Gelfond. Logic Programs with Consistency-Restoring Rules. In Patrick Doherty, John McCarthy, and Mary-Anne Williams, editors, *International Symposium on Logical Formalization of Commonsense Reasoning*, AAAI 2003 Spring Symposium Series, Mar 2003.
2. Chitta Baral. *Knowledge Representation, reasoning and declarative problem solving with Answer Sets.* Cambridge University Press, Cambridge, 2003.
3. Gerhard Brewka. Logic Programming with Ordered Disjunction. In *Proceedings of the 18th National Conference on Artificial Intelligence, AAAI-2002.* Morgan Kaufmann, 2002.
4. Michael Gelfond and Vladimir Lifschitz. The Stable Model Semantics for Logic Programming. In R. Kowalski and K. Bowen, editors, *5th Conference on Logic Programming*, pages 1070–1080. MIT Press, 1988.
5. Michael Gelfond and Vladimir Lifschitz. Action languages. *Electron. Trans. Artif. Intell.*, 2:193–210, 1998.
6. Fred Kroger. *Temporal Logic of Programs.* Springer Verlag, Berlin, 1987.
7. Nicola Leone and Simona Perri. Parametric Connectives in Disjunctive Logic Programming. In *ASP03 Answer Set Programming: Advances in Theory and Implementation*, Messina, Sicily, September 2003.
8. R. Pucella. Logic column 11: The finite and the infinite in temporal logic. *ACM SIGACT News*, 36(1):86–99, 2005.
9. Tran Cao Son and Enrico Pontelli. Planning with preferences using logic programming. In *LPNMR*, pages 247–260, 2004.
10. Claudia Zepeda, Mauricio Osorio, Juan Carlos Nieves, Christine Solnon, and David Sol. Applications of preferences using answer set programming. In *Submmited to Answer Set Programming: Advances in Theory and Implementation (ASP 2005).*

Planning motions for animated characters

Abraham Sánchez[1], Josué Sánchez[1] and René Zapata[2]

[1] Facultad de Ciencias de la Computación, BUAP
14 Sur esq. San Claudio, CP 72550
Puebla, Pue., México
asanchez@cs.buap.mx
[2] LIRMM, UMR5506 CNRS, 161 rue Ada 34392,
Montpellier Cedex 5, France
zapata@lirmm.fr

Abstract. Human or more generally articulated figure animations have been seen in a variety of application fields including advertising, entertainment, education, and simulation. The primary research goal now to further the animation is providing a system which allows animators to easily and interactively design and get desired movements. We present a method for animating human characters, especially dedicated to walk planning in virtual environments. Our method is integrated in a probabilistic roadmap method scheme. The success of our approach is demonstrated through several examples.

Keywords: Motion planning, autonomous characters, probabilistic roadmaps.

1 Introduction

A typical motion planning problem asks for computing a collision-free motion between two given placements of a given robot in an environment populated with obstacles. The problem is typically solved in the *configuration space* $\mathcal{C}$.

Over the years many different approaches to solve this problem have been suggested. The probabilistic roadmap planner (PRM) is a relatively new approach to motion planning [1]. It turns out to be very efficient, easy to implement, and applicable for many different types of problems.

Motion planning has application in many other areas, such as assembly planning, design for manufacturing, virtual prototyping, computer animation, medical surgery simulation, and computational biology. As stressed by Latombe [2], non-robotics applications (e.g. graphics animation, surgical planning and computational biology) are growing in importance and are likely to shape future motion planning research at least as much as robotics.

This work presents a solution for the planning human walk through virtual environments. The solution is based on probabilistic roadmap motion planning techniques combined with a character motion controller.

Advances in Artificial Intelligence Theory
Research on Computing Science 16, 2005, pp. 265-274

2 Related work

An early effort at addressing the motion planning problem in computer graphics was proposed in [3]. In [4] a path planner is developed for several cooperating arms to manipulate a movable object between two configurations in the context of computer animation.

Later, J. Kuffner [5] proposed a new technique for computing collision-free navigation motions from task-level commands for animated human characters in interactive virtual environments. His approach is efficient and consists in splitting the problem in two parts: the digital actor is bounded by a cylinder to compute a collision-free trajectory and a motion controller is used to animate the actor along the planned trajectory.

Foskey et al [6] proposed a hybrid motion planning algorithm for rigid bodies translating and rotating in a 3D workspace. The method generates a Voronoï roadmap in the workspace and combine it with "bridges" computed by randomized path planning with Voronoï-based sampling.

A practical motion planner for humanoids and animated human figures was proposed in [7]. In this work, the workspace is modelled as a multi layered grid and several types of digital actors bounding boxes are considered according to predefined locomotion behaviours (walking, crawling,...). Once a path has been found in the grid, the cyclic motion patterns are used to animate a trajectory throughout the path. Finally, the animation is modified using dynamic filters to make it consistent.

A similar approach that combines path planner and motion controller has been proposed in [8]. A solution to the locomotion planning problem for digital actors was recently proposed in [9]. The solution is based both on probabilistic motion planning and on motion capture blending and warping.

3 Model and motion data

We know that many dynamics models were designed in the nineties in order to synthesize human figure motion, but it is unlikely that dynamics simulation will solve all animation problem. Researchers have thus turned to other kinds of approaches. The recent progress in motion capture techniques makes it possible to directly use human motion data. Animating very complex model such as virtual human is usually done by extracting a simpler representation of the model, a "skeleton", that is an articulated figure made of rigid links connected by hinges. We first prepare a skeleton model represented with a hierarchical structure of rotational joints. The number of joints of the model and the degrees of freedom depend largely on the desired reality or quality. In this paper we employ a relatively simple model with 52 degrees of freedom (dofs) detailed on Fig. 1. Motion capture data depend on the model and represent the evolution of the configuration through the time, in several cases: walks, turns, runs, etc.

The structure of the character is modeled in two levels. Pelvis and legs are used for the locomotion, all the 18 dofs are said to be active dofs. The 34 other

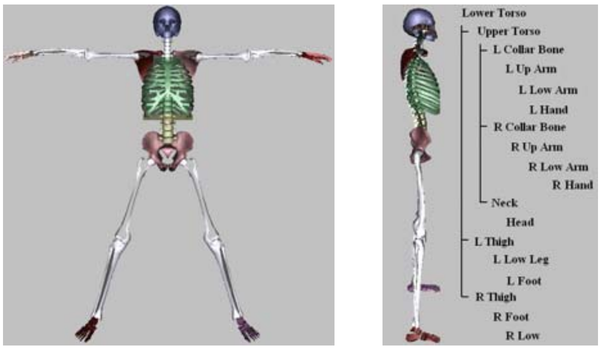


Fig. 1. Computer model of the character.

ones are said to be reactive dofs, they deal with the control of the arms and the spine [9]. The pelvis is the root of five kinematics chains modelling respectively the arms, the legs and the spine. Root's trajectory problem concerns only $[x, y, z, \theta, \phi, \psi]$ parameter's evolution. The inputs are two collision-free configurations of the character in the configuration space. The local navigation of the character is modeled with a third degree Bezier curve.

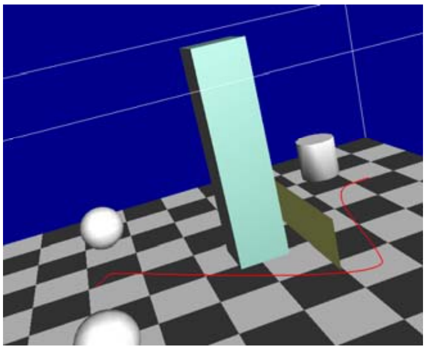


Fig. 2. A path made of few Bezier curves.

Bounding the character's geometry by a cylinder allows motion planning for navigation to be reduced to planning collision-free trajectories for this cylinder in 3D.

Motion capture yields an unstructured representation, a sequence of sampled positions for each degree of freedom, or through pre-processing using inverse kinematics, sequences of joint angle values. Editing this kind of iconic description poses a problem analogous to that of editing a bitmapped image or a sampled hand-drawn curve. We call the motion warping all techniques that take well-known trajectories and modify them in order to change the motion. Witkin and

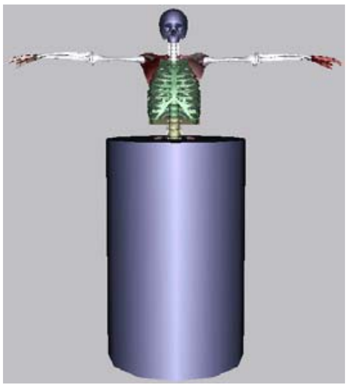


Fig. 3. The character's geometry is bounded by an appropriate cylinder.

Popović [10] modified a reference trajectory $\theta_i(t)$ (where i represents the i^{th} parameter of the system) by interactively tuning the position of selected key-frames and by scaling and shifting $\theta_i(t)$.

$$\begin{array}{c} \forall i, \theta_i'(t) = a(t)\theta_i(t) + b(t) \\ t = g(t') \end{array} \tag{1}$$

Function $a(t)$ is used to scale the signal and $b(t)$ is used to change the center of scaling of a. The deformation from time t to t' is a constrained interpolation based on Cardinal splines. Thus, the resulting sequence satisfies the constraints of new key-frames with respect to the pattern of the initial motion. Moreover, blending of several motion can be obtained by weighted sums.

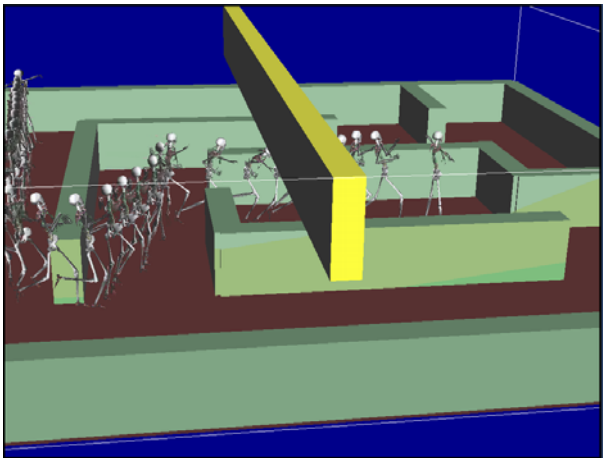


Fig. 4. The result of the warping module implemented.

4 RMP3D architecture

RMP3D is composed of diverse modules associated with functionalities such as the modeling of the mechanical systems (geometric modeling, steering methods), geometrical tools (collision detection, distance calculation), motion planning, and a graphic interface that allows to define the problems, call the algorithms, and to display the produced results. Fig. 5 shows the structure of the motion planning software RMP3D.

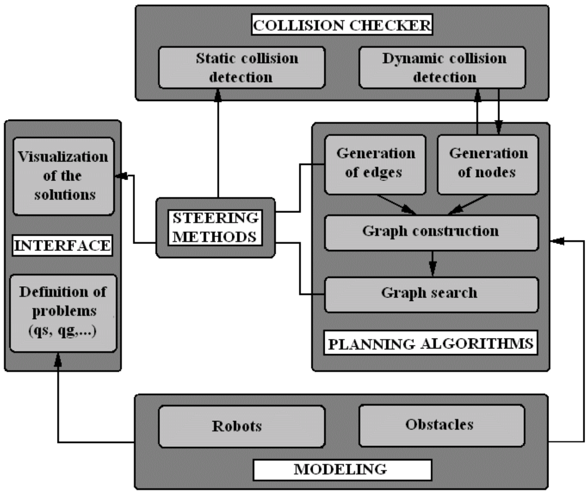


Fig. 5. Architecture of RMP3D.

- the modeling module enables the user to describe mechanical systems and environments.
- the geometric tools for the collision detection algorithms.
- the steering methods allows to compute local paths satisfying the kinematic constraints of the mechanical systems.
- the planning algorithms module contains many procedures based on randomized techniques such as PRM, Lazy PRM, Visibility PRM.

The following steering methods are actually integrated within RMP3D:

- **Linear** computes a straight line segment between two configurations, this method works for any holonomic system.
- **Nonholonomic** computes smooth paths for both car models, Reeds & Shepp and Dubins.

Other methods could be easily integrated into this library.

The planning module integrates three of the randomized techniques proposed recently. These techniques are based on the probabilistic roadmap methods that first construct a roadmap connecting collision-free configurations picked at random, and then use this roadmap to answer multiple or single queries.

- Basic-PRM is based on the basic PRM scheme [1]. The key idea of this scheme is to randomly distribute a set of nodes in $\mathcal{C}$ and then connect these nodes using a simple local planner (or a steering method), to form a graph (or a tree) known as a roadmap.
- Gaussian-PRM is meant to add more samples near obstacles [11]. The idea is to take two random samples, where the distance between the sampless is chosen according to a Gaussian distribution. Only if one of the samples lies in the $\mathcal{C}_{free}$ and the other lies in $\mathcal{C}_{obstacles}$ do we add the free sample. It has been shown that this leads to a favorable sample distribution.
- Lazy-PRM [12], the idea is not to test whether the paths are collision free unless they are really needed. The goal of this variant is to minimize the number of collision checks. The rational behind this is that for most paths we only need to consider a small part of the graph before a solution is found.

The collision checker integrated is PQP[1] for determining whether a given path is collision-free or not (this is performed by multiple calls to the interference detection algorithm). PQP is a library for performing three types of proximity queries on a pair of geometric models composed of triangles.

5 Planning algorithm

Navigation in unstructured environments entails some particular challenges. Global and local solutions can be strongly linked; the choice of a particular route towards a goal is predicated on the route being viable every step along the way. Planning algorithms for such problems thus require the ability to plan motions across both small and large time scales. A second challenge is that creating motions involves both discrete and continuous decisions. An example of a discrete decision is that of deciding whether to step on or over an obstacle, or simply deciding which of a finite set of possible hand-holds to use. Once the contact points of a character with the environment have been chosen, the remaining decisions shaping the motion can be regarded as being continuous in nature. This particular environment requires the alternating use of several modes of locomotion in order to navigate towards the goal.

Given start and goal positions in a virtual environment, our objective is to find a sequence of motions of a human character to move from the start and to the goal. Conventional motion planning techniques in robotics typically generate very efficient mechanical movements rather that lifelike natural-looking motions desired in computer animation applications. On the other hand, motion editing techniques in computer graphics are not equipped with a high-level planning

[1] A collision detection package from University of North Carolina at Chapel Hill

capability to yield a desired motion. To rapidly plan convincing motions of the human-like character with high-level directives, we use a novel combination of probabilistic path planning and motion capture editing techniques.

Our scheme consists of the following three steps: roadmap construction, roadmap search, and motion generation.

1. **Roadmap construction**: Given a virtual environment, we randomly sample valid configurations of the cylinder bounding the lower part of the character. The roadmap can be modeled as a directed graph whose nodes represent valid samples of the configuration space. A pair of nodes are connected by and edge if the character can move from one node to the other with a prescribed motion while preserving its lifelikeness.
2. **Roadmap search**: Once the roadmap reflects the connectivity of $\mathcal{C}_{free}$ it can be used to answer motion planning queries. Then a path in the roadmap is found which corresponds to a motion of the character. The first path is optimised by a classical dichotomy technique. The output of this step is a sequence of continuous composition of Bezier curves.
3. **Motion generation**: Simply computing a collision-free path in the environment is not enough to produce realistic animation. This step transforms the obtained path into a discrete set of time stamped positions for the character along the trajectory, respecting some criteria of velocities and accelerations. Finally, a locomotion control procedure transforms a set of time parametrised positions into a walk sequence, for example. It is based on a motion capture blending technique [9].

Although the motion planning and following concept generally applies to many types of characters and motions, we will concentrate on generating walking or running motions for a human-like character. We would like the character's motion to be smooth and continuous, natural-looking, and follow the computed path as closely as possible. Though many kinds of path following techniques could potentially be used [13].

6 Implementation and experimental results

The motion planning platform RMP3D was implemented on an Intel © Pentium IV processor-based PC running at 2.6 GHz with 1Gb RAM, using Builder C++ and OpenGL.

During an interactive session, the user can choose the inputs (initial and final configurations), and the motion planner will calculate a collision-free path (if one exists) in approximately some seconds (this time depends on the used planner and the geometry of the environment). The path is then smoothed by means of Bezier curves and sent directly to the controller, and the character will immediately begin follow the new path.

Our first experiments is for planning a walking motion. Figures 6 and 7 show an example of the construction of an animation.

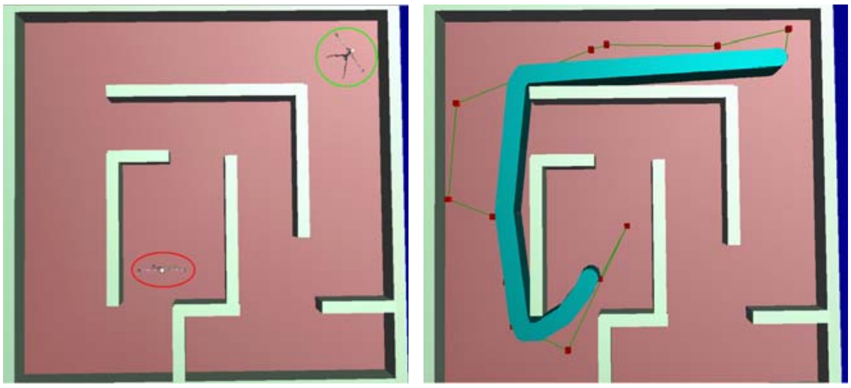


Fig. 6. Start and goal configurations (inputs) and root's trajectory.

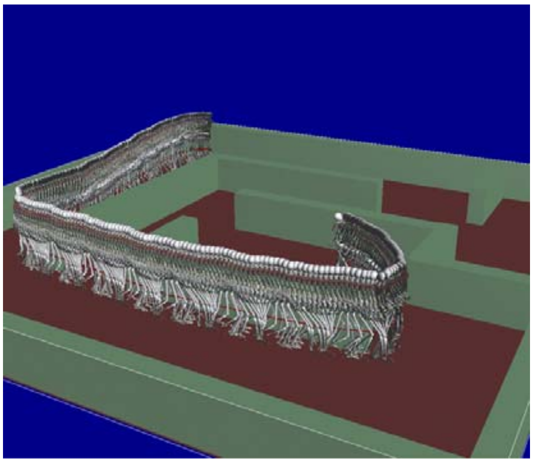


Fig. 7. Walk cycle introduction.

Good performance has been achieved, even on complex scenes with multiple characters and environments composed of more than 191, 410 triangle primitives. A result of complete trajectory (composition of several local paths) is presented on Fig. 8. One can notice several specificities in this result: the model strictly respects the initial and final configurations required (the initial and final phases of walk are thus influenced by these data).

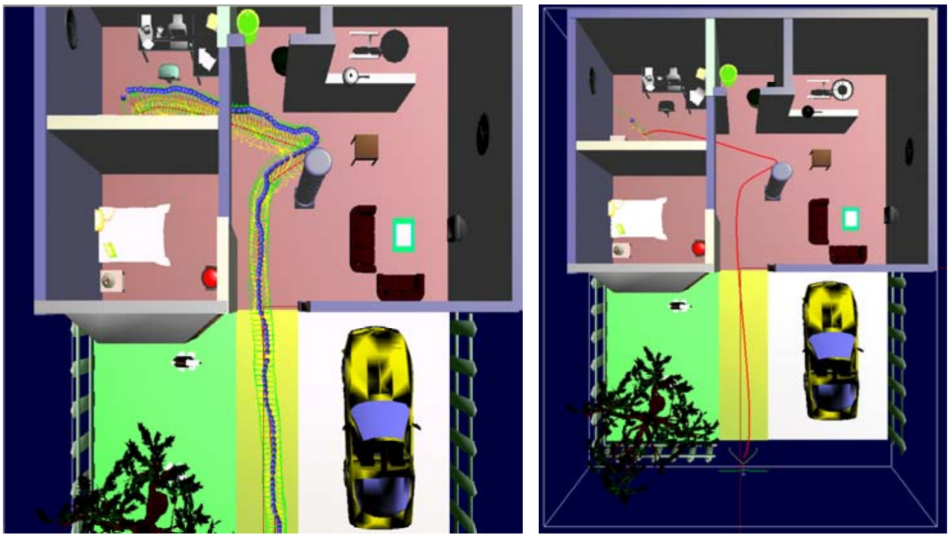


Fig. 8. A complete walk planning through a complex environment (191, 410 triangles).

We have presented a human walk planning method associating randomized motion planning and motion capture editing techniques. Our first results obtained are very promising.

7 Conclusions and future work

Animation of human walking is a crucial problem in computer graphics: many synthetic scenes involve virtual humans, form special effects in the film industry to virtual reality and video games. Synthesizing realistic human motion is a challenge. The study of walking motions has been generated much interest in other fields such as biomechanics and robotics. The interest for this area has never decreased in the computer animation community, even though the task is not easy, since techniques based on kinematics, dynamics, biomechanics, robotics and even signal processing may be required.

This work presented a method for animating human characters. This approach satisfies some computer graphics criteria such as realistic results, collision-free motion in cluttered environments and low response time. The approach has been implemented in our RMP3D architecture and successfully demonstrated on several examples. The combination of randomized motion planning techniques and motion capture editing techniques offer promising results [9].

RMP3D uses Basic-PRM, Gaussian-PRM or Lazy-PRM techniques that captures the configuration space connectivity into a roadmap. Through the examples we have demonstrated that our approach works well. The computing time vary according to the selected scheme of planning.

Although some promising results are shown in its present form, the algorithm could be improved in a number of important ways. We are interested in investigating reactive planning methods to deal with dynamic environments. The generation of collision-free motions for two or more characters, is another objective to the future and also the cooperation between virtual characters handling bulky objects, etc.

References

1. Kavraki, L., Švestka, P., Latombe, J-C., Overmars, M. H. "Probabilistic roadmaps for path planning in high-dimensional configuration spaces", IEEE Transactions on Robotics and Automation. Vol 12, No. 4, (1996) 566-579
2. Latombe J. C:: "Motion planning: A journey of robots, molecules, digital actors, and other artifacts", The International Journal of Robotics Research, Vol. 18, no. 11, (1999) 1119-1128
3. Lengyel J, Reichert M, Donald B., Greenberg D. P.: "Real-time robot motion planning using rasterizing computer graphics hardware", ACM SIGGRAPH, Vol. 24 (1990) 327-335
4. Koga Y, Kondo K, Kuffner J, Latombe J. C.: "Planning motions with intentions", ACM SIGGRAPH, (1994) 395-408
5. Kuffner J. J.: "Goal-directed navigation for animated characters using real-time path planning and control", CAPTECH 98, (1998)
6. Foskey M, Garber M, Lin M, Manocha D.: "A Voronoï-based hybrid motion planner for rigid bodies", IEEE Int. Conf. on Robotics and Automation, (2000)
7. Shiller Z, Yamane K, Nakamura Y.: "Planning motion patterns of human figures using a multi-layered grid and the dynamics filter", IEEE Int. Conf. on Robotics and Automation, (2001) 1-8
8. Choi M. G., Lee J, Shin S. Y.: "Planning biped locomotion using motion capture data and probabilistic roadmaps", ACM Transactions on Graphics, Vol. 5, No. N (2002) 1-25
9. Pettré J., Laumond J. P., Siméon T.: "A 2-stages locomotion planner for digital actors", Eurographics/SIGGRAPH Symposium on Computer Animation, (2003)
10. Witkin A, Popović Z.: "Motion warping", SIGRAPH 95, (1995)
11. Boor V., Overmars M., Van der Steppen F. "The gaussian sampling strategy for probabilistic roadmap planners", IEEE Int. Conf. on Robotics and Automation, (1999) 1018-1023
12. Bohlin, R., Kavraki, L. "Path planning using lazy PRM", IEEE Int. Conf. on Robotics and Automation, (2000) 521-528
13. Multon F, France L, Cani-Gascuel M. P., Debunne G.: "Computer animation of human walking: A survey", Journal of Visualization and Computer Animation (1999)

Author Index

Índice de autores

Aciar, Silvana Vanesa 55
Aguirre, José Luis 23
Alarcón Avila, Enrique 205
Andrade Zaldívar, Zeus 87
Brena, Ramon F. 23,107
Cantú Ortiz, Francisco J. 65
Castro, Marco 75
Chia Shyang, Lin 119
Chun Pin, Chang 139
Corchado, J.M. 129
Díaz, F. 129
Diaz, Marissa 243
Diaz, Sandra 243
Domínguez, Jesús Héctor 23
Dong Her, Shih 119,139
Dudek, Gregory 151
Espinosa Carrillo, Enrique David 87
Fernández Caballero, Antonio 33
Fernández Riverola, F. 129
Fernández, Luis Marcelo 23
Frydman P., Jose A. 65
Galaviz, Jose 173,193
Gamboa, Marco 75
García, Karen 97
Gascueña, José M. 33
González, Pascual 33
Herrera Alcántara, Oscar 183,193
Herrera, Francisco 75
Hsiu Sen, Chiang 119,139
Iglesias, E. L. 129
Jiménez Velasco, María Gpe. 233
Jiménez, Antonio 3
Jui Chi, Shen 119
Junco Rey, María 87
Kuri Morales, Angel 173,183,193
León Fernández de Lara, Yazid 223
Lluis de la Rosa, Josep 55
López Herrera, Josefina 55
Martinez Cuevas, Stivalis Anahi 205
Mateos, Alfonso 3
Méndez, J. R. 129
Monroy, Raúl 97
Montes González, Fernando 213
Morales Menéndez, Rubén 65
Muñoz Meléndez, Angélica 223,233
Nieto, Carlos 243
Ortiz Posadas, Martha 193
Osorio Lama, María A. 13
Osorio, Mauricio 255
Ramírez Uresti, Jorge Adolfo 87
Ramirez, Eduardo H. 107
Rangel Huerta, Alejandro 205
Ríos Insua, Sixto 3
Rodríguez, Luis C. 3
Sánchez López, Abraham 13,265
Sánchez Texis, Josué 265
Santos Reyes, José 213
Segovia, Karla L. 161
Sheng Fei, Hsu 139
Sierra Cavazos, Jorge H. 65
Sol, David 255
Solnon, Christine 255
Swain, Ricardo 243
Tejeda Gómez, José Arturo 87
Torres Méndez, Luz Abril 151
Uribe, Diego 45
Vázquez, Diana M. 161
Vázquez, Jesús 97
Vázquez, Roberto A. 161
Zapata, René 265
Zepeda, Claudia 255

Gabriela Ochoa Meier
Mauricio Osorio Galindo
Andrés Pérez Uribe
Manuel Palomar
Luis Alberto Pineda
Andre Ponce de Leon F. de Carvalho
David Poole
Bhanu Prasad
Jorge Adolfo Ramírez Uresti
Fernando Ramos
Carlos Alberto Reyes García
Abdennour El Rhalibi
Maria Cristina Riff
Roger Z. Rios
Dave Robertson
Horacio Rodríguez
Riccardo Rosati
Isaac Rudomín
Alessandro Saffiotti
Gildardo Sánchez
Alberto Sanfeliú Cortés
Andrea Schaerf
Thomas Schiex
Leonid Sheremetov
Grigori Sidorov
Carles Sierra
Alexander V. Smirnov
Maarten van Someren
Juan Humberto Sossa Azuela
Rogelio Soto
Thomas Stuetzle
Luis Enrique Sucar Succar
Ricardo Swain Oropeza
Hugo Terashima
Demetri Terzopoulos
Manuel Valenzuela
Juan Vargas
Felisa Verdejo
Manuel Vilares Ferro
Toby Walsh
Alfredo Weitzenfeld
Nirmalie Wiratunga
Franz Wotawa
Kaori Yoshida
Claus Zinn
Berend Jan van der Zwaag

Additional Reviewers

Árbitros adicionales

Juan C. Acosta Guadarrama
Héctor Gabriel Acosta Mesa
Teddy Alfaro
Miguel A. Alonso
José Ramón Arrazola
Stella Asiimwe
Séverine Bérard
Fco. Mario Barcala Rodríguez
Axel Arturo Barcelo Aspeitia
Adam D. Barker
Alejandra Barrera
Gustavo E. A. P. A. Batista
Abderrahim Benslimane
Arturo Berrones
Bastian Blankenburg
Pascal Brisset
Andreas Bruening
Mark Buckley
Olivier Buffet
Diego Calvanese
Hiram Calvo
Niccolo Capanni
Carlos Castillo
Sutanu Chakraborti
Carlos Chesñevar
Federico Chesani
Wu Feng Chung
Murilo Coelho Naldi
Mark Collins
Jean François Condotta
Miguel Contreras
Sylvie Coste Marquis
Anne Cregan
Ronaldo Cristiano Prati
Juan A. Díaz
Víctor Manuel Darriba Bilbao
Michael Dekhtyar
Deepak Devicharan
Luca Di Gaspero
Marissa Diaz
Luigi Dragone
Edgar Duéñez
Mehmet Önder Efe
Arturo Espinosa Romero
Katti Faceli
Antonio Fernandez Caballero
Antonio Ferrández
Armin Fiedler
Alfredo Gabaldón
Arturo Galván Rodríguez
Ariel García
Cormac Gebruers
Karina Gibert
Andrea Giovannucci
Fernando Godínez
Giorgi Goguadze
Miguel González
Jorge Graña
Federico Guedea
Alejandro Guerra Hernández
Daniel Gunn
Everardo Gutiérrez
Christian Hahn
Emmanuel Hebrard
Benjamín Hernández
Martin Homik
Rodolfo Ibarra
Boyko Iliev
Bartosz Jablonski
Jean Yves Jaffray
Sylvain Jasson
Daniel Jolly
Narendra Jussien
Lars Karsson
Ryszard Klempous
Jerzy Kotowski
A. Krizhanovsky
Juan Carlos López Pimentel
David Lambert
Darío Landa Silva
Jérme Lang
Huei Diana Lee
Domenico Lembo
Paul Libberecht
Ana Carolina Lorena
Robert Lothian

Henryk Maciejewski
Fernando Magan Muñoz
Michael Maher
Donato Malerba
Salvador Mandujano
Ana Isabel Martinez Garcia
Patricio Martinez Barco
Jarred McGinnis
Andreas Meier
Manuel Mejia Lavalle
Corrado Mencar
Thomas Meyer
Thomas Meyer
Erik Millan
Monica Monachini
Rebecca Montanari
Andrés Montoyo
Jaime Mora Várgas
José Andrés Moreno Pérez
Rafael Muñoz
Martin Muehlenbrock
Rahman Mukras
Amedeo Napoli
Gonzalo Navarro
Adeline Nazarenko
Juan Carlos Nieves
Peter Novak
Slawomir Nowaczyk
Oscar Olmedo Aguirre
Magdalena Ortiz de la Fuente
María Osorio
Joaquín Pacheco
Marco Patella
Jesús Peral
Mats Petter Pettersson
Steven Prestwich
Bernard Prum
José Miguel Puerta Callejón
Alonso Ramírez Manzanárez
Fernando Ramos
Orión Fausto Reyes Galaviz
Francisco Ribadas Peña
Fabrizio Riguzzi
Leandro Rodríguez Liñares
Juan A. Rodríguez Aguilar
Raquel Ros
Maximiliano Saiz Noeda
S. Sandeep
P. Sanongoon
Cipriano Santos
Vitaly Schetinin
Marvin Schiller
Przemyslaw Sliwinski
Jasper Snoek
Thamar Solorio
Claudia Soria
Eduardo J. Spinosa
Cyrill Stachniss
Ewa Szlachcic
Armagan Tarim
Choh Man Teng
Paolo Torroni
Elio Tuci
Carsten Ullrich
L. Alfonso Ureña López
Diego Uribe
Mars Valiev
Maria Vargas Vera
Wamberto Vasconcelos
José Luis Vega
José Luis Vicedo
Jesús Vilares Ferro
Mario Villalobos Arias
Nic Wilson
Sean Wilson
Claudia Zepeda
Juergen Zimmer

Impreso en los Talleres Gráficos
de la Dirección de Publicaciones
del Instituto Politécnico Nacional
Tresguerras 27, Centro Histórico, México, D.F.
Noviembre de 2005.
Printing 500 / Edición 500 ejemplares.