Studies in Computational Intelligence 451

Oscar Castillo
Patricia Melin
Janusz Kacprzyk (Eds.)

Recent Advances on Hybrid Intelligent Systems

Springer

Studies in Computational Intelligence 451

For further volumes:
http://www.springer.com/series/7092

Oscar Castillo, Patricia Melin,
and Janusz Kacprzyk (Eds.)

Recent Advances on Hybrid Intelligent Systems

Springer

Editors
Prof. Oscar Castillo
Department of Computer Science
Tijuana Institute of Technology
Tijuana
Mexico
and
Chula Vista CA
USA

Prof. Patricia Melin
Department of Computer Science
Tijuana Institute of Technology
Tijuana
Mexico
and
Chula Vista CA
USA

Prof. Janusz Kacprzyk
Polish Academy of Sciences
Systems Research Institute
Newelska
Warszawa
Poland

ISSN 1860-949X
ISBN 978-3-642-33020-9
DOI 10.1007/978-3-642-33021-6
Springer Heidelberg New York Dordrecht London

e-ISSN 1860-9503
e-ISBN 978-3-642-33021-6

Library of Congress Control Number: 2012945328

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

We describe in this book, recent advances on hybrid intelligent systems using soft computing techniques for intelligent control and robotics, pattern recognition, time series prediction and optimization complex problems. Soft Computing (SC) consists of several intelligent computing paradigms, including fuzzy logic, neural networks, and bio-inspired optimization algorithms, which can be used to produce powerful hybrid intelligent systems. The book is organized in five main parts, which contain a group of papers around a similar subject. The first part consists of papers with the main theme of hybrid intelligent systems for control and robotics, which are basically papers that propose new models and concepts, which can be the basis for achieving intelligent control and mobile robotics. The second part contains papers with the main theme of hybrid intelligent systems for pattern recognition and time series prediction, which are basically papers using nature-inspired techniques, like evolutionary algorithms, fuzzy logic and neural networks, for achieving efficient pattern recognition or time series prediction. The third part contains papers with the theme of bio-inspired and genetic optimization methods, which basically consider the proposal of new methods and applications of bio-inspired optimization to solve complex optimization problems. The fourth part contains papers that deal with the application of intelligent optimization techniques in real world problems. The fifth part contains papers with the theme of evolutionary methods and intelligent computing, which are papers considering soft computing methods for applications related to diverse areas, such as natural language processing, recommending systems and optimization.

In the part of hybrid intelligent systems for control and robotics there are 8 papers that describe different contributions that propose new models and concepts, which can be the considered as the basis for achieving intelligent control and mobile robotics for real world problems. In the part of hybrid intelligent systems for pattern recognition and time series prediction there are 9 papers that describe different contributions on achieving efficient pattern recognition and accurate time series prediction using hybrid intelligent systems based on soft computing techniques. In the part of bio-inspired and genetic optimization methods there are 9 papers that describe different contributions of new algorithms for optimization and their

application to diverse complex optimization problems. The bio-inspired methods include variations of ant colony optimization and particle swarm optimization. In the part of intelligent optimization applications, there are 7 contributions that describe the development of new models and algorithms relevant to complex optimization problems, as well as the application of these intelligent optimization techniques in real-world applications. In the part of evolutionary methods and intelligent computing there are 9 contributions on models and algorithms based on computational intelligent techniques, including novel evolutionary approaches, that are presented, as well as their applications to different real-world problems.

In conclusion, the edited book comprises papers on diverse aspects of bio-inspired models, soft computing and hybrid intelligent systems for control, mobile robotics, pattern recognition, time series prediction and other complex real world problems. There are theoretical aspects as well as application papers.

July 2, 2012

Oscar Castillo
Tijuana Institute of Technology, Mexico

Patricia Melin
Tijuana Institute of Technology, Mexico

Janusz Kacprzyk
Polish Academy of Sciences, Poland

Contents

Part II: Hybrid Intelligent Systems for Pattern Recognition and Time Series Prediction

Part III: Bio-inspired and Genetic Optimization Methods

Part IV: Intelligent Optimization Methods and Applications

Part I

Hybrid Intelligent Systems for Control and Robotics

Optimization of a Fuzzy Tracking Controller for an Autonomous Mobile Robot under Perturbed Torques by Means of a Chemical Optimization Paradigm

Leslie Astudillo, Patricia Melin, and Oscar Castillo

Tijuana Institute of Technology, Tijuana México
leslie.astudillo@suntroncorp.com,
{epmelin,ocastillo}@hafsamx.org

Abstract. This paper addresses the tracking problem for the dynamic model of a unicycle mobile robot. A novel optimization method inspired on the chemical reactions is applied to solve this motion problem by integrating a kinematic and a torque controller based on fuzzy logic theory. Computer simulations are presented confirming that this optimization paradigm is able to outperform other optimization techniques applied to this particular robot application.

1 Introduction

Optimization is an activity carried out in almost every aspect of our life, from planning the best route in our way back home from work to more sophisticated approximations at the stock market, or the parameter optimization for a wave solder process used in a printed circuit board assembly manufacturer optimization theory has gained importance over the last decades. From science to applied engineering (to name a few), there is always something to optimize and of course, more than one way to do it.

In a generic definition, we may say that optimization aims to find the "best" available solution among a set of potential solutions in a defined search space. For almost every problem exists a solution, not necessarily the best, but we can always find an approximation to the "ideal solution", and while in some cases or processes is still common to use our own experience to qualify a process, a part of the research community have dedicated a considerably amount of time and efforts to help find robust optimization methods for optima finding in a vast range of applications.

It has been stated the difficulty to solve different problems by applying the same methodology, and even the most robust optimization approaches may be outperformed by other optimization techniques depending on the problem to solve.

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 3–20.
springerlink.com

When the complexity and the dimension of the search space make a problem unsolvable by a deterministic algorithm, probabilistic algorithms deal with this problem by going through a diverse set of possible solutions or candidate solutions. Many metaheuristic algorithms can be considered probabilistic, while they apply probability tools to solve a problem, metaheuristic algorithms seek good solutions by mimicking natural processes or paradigms. Most of these novel optimization paradigms inspired by nature were conceived by merely observation of an existing process and their main characteristics were embodied as computational algorithms.

The importance of the optimization theory and its application has grown in the past few decades, from the well known Genetic Algorithm paradigm to PSO, ACO, Harmonic Search, DNA Computing, among others, they all were introduced with the expectation of improving the results obtained with the existing strategies.

There's no doubt that there could be some optimization strategies pre-sented at some point that were left behind due their complexity and poor performance. Novel optimization paradigms should be able to perform well in comparison with another optimization techniques and must be "easily adaptable" to different kinds of problems.

Optimization based on chemical processes is a growing field that has been satisfactorily applied to several problems. In [25] A DNA based algorithm was to solve the small hitting set problem. A catalytic search algorithm was explored in [30], where some physical laws such as mass and energy conservation were taken into account. In [19],the potential roles of energy in algorithmic chemistries were illustrated. An energy framework was introduced, which keeps the molecules within a reasonable length bounds, allowing the algorithm to behave thermodynamically and kinetically similar to real chemistry. A chemical reaction optimization was applied to the grid scheduling problem in [29], where molecules interact with each other aiming to reach the minimum state of free potential and kinetic energies. The main difference between these metaheuristics is the parameter representation, which can be explicit or implicit.

In this paper we introduce an optimization method inspired on the chemical reactions and its application for the optimization of the tracking controller for the dynamic model of the unicycle mobile robot.

The importance of applying this chemical optimization algorithm is that different methods have been applied to solve motion control problems. Kanayama et al. [13] propose a stable tracking control method for a non-holonomic vehicle using a Lyapunov function. Lee et al. [15] solved tracking control using backstepping and in [17] with saturation constraints. Furthermore, most reported designs rely on intelligent control approaches such as fuzzy logic control [3][12][16][23][27][28] and neural networks [10][26].

However the majority of the publications mentioned above, have concentrated on kinematic models of mobile robots, which are controlled by the velocity input, while less attention has been paid to the control problems of nonholonomic dynamic systems, where forces and torques are the true inputs: Bloch and Drakunov [4] and Chwa [8], used a sliding mode control to the tracking control problem. Fierro and Lewis [9] propose a dynamical extension that makes possible the

integration of kinematics and torque controller for a nonholonomic mobile robot. Fukao et al. [11], introduced an adaptive tracking controller for the dynamic model of mobile robot with unknown parameters using backstepping methodology, which has been recognized as a tool for solving several control problems [24] [31].Motivated by this, a mamdani fuzzy logic controller is introduced in order to drive the kinematic model to a desired trajectory in a finite-time, considering the torque as the real input, a chemical reaction optimization paradigm is applied and simulations are shown.

Further publications [2][18][6] have applied bio-inspired optimization techniques to find the parameters of the membership functions for the fuzzy tracking controller that solves the problem for the dynamic model of a unicycle mobile robot, using a fuzzy logic controller that provides the required torques to reach the desired velocity and trajectory inputs.

In this paper, the main contribution is the representation of the fuzzy controller in the chemical paradigm to search for the optimal parameters. Simulation results show that the proposed approach outperforms other nature inspired computing paradigms, such as genetic algorithms, particle swarm and ant colony optimization.

The rest of this paper is organized as follows. Section 2 illustrates the proposed methodology. Section 3 describes the problem formulation and control objective. Section 4 describes the proposed fuzzy logic controller of the robot. Section 5 shows some experimental results of the tracking controller and in section 6 some conclusions and future work are presented.

2 The Chemical Optimization Paradigm

The proposed chemical reaction algorithm is a metaheuristic strategy that performs a stochastic search for optimal solutions within a defined search space. In this optimization strategy, every solution is represented as an element (or compound), and the fitness or performance of the element is evaluated in accordance with the objective function. The general flowchart of the algorithm is shown in Figure 1.

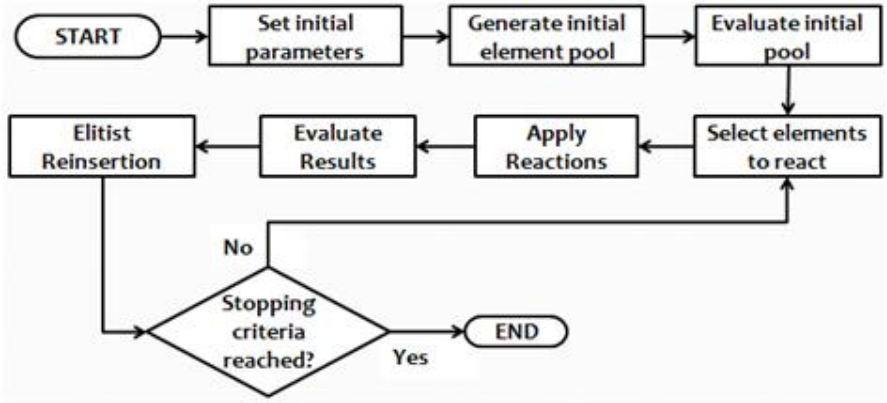


Fig. 1. General flowchart of the chemical reaction algorithm

The main difference with other optimization techniques [25][30][19][29] is that no external parameters are taken into account to evaluate the results, while other algorithms introduce additional parameters (kinetic/potential energies, mass conservation, thermodynamic characteristics, etc), this is a very straight forward methodology that takes the characteristics of the chemical reactions (synthesis, decomposition, substitution and double-substitution) to find for optimal solution.

This approach is a static population-based metaheuristic that applies an abstraction of the chemical reactions as intensifiers (substitution, double substitution reactions) and diversifying (synthesis, decomposition reactions) mechanisms. The elitist reinsertion strategy allows the permanence of the best elements and thus the average fitness of the entire element pool increases with every iteration. The algorithm may trigger only one reaction or all of them, depending on the nature of the problem to solve, in example; we may use only the decomposition reaction subroutine to find the minimum value of a mathematical function.

In order to have a better picture of the general schema for this proposed chemical reaction algorithm, a comparison with other nature inspired paradigms is shown in Table 1.

Table 1. Main elements of several nature inspired paradigms

Paradigm	Parameter Representation	Basic Operations
GA	Genes	Crossover, Mutation
ACO	Ants	Pheromone
PSO	Particles	Cognitive, Social Coefficients
GP	Trees	Crossover, Mutation (In some cases)
CRM	Elements, Compounds	Reactions (Combination, Decomposition, Substitution, Double-substitution)

3 The Mobile Robot

Mobile robots are non-holonomic systems due to the constraints imposed on their kinematics. The equations describing the constraints cannot be integrated symbolically to obtain explicit relationships between robot positions in local and global coordinate's frames. Hence, control problems that involve them have attracted attention in the control community in recent years [14].

The model considered is that of a unicycle mobile robot (see Figure 2) that has two driving wheels fixed to the axis and one passive orientable wheel that are placed in front of the axis and normal to it [5].

The two fixed wheels are controlled independently by the motors, and the passive wheel prevents the robot from overturning when moving on a plane.

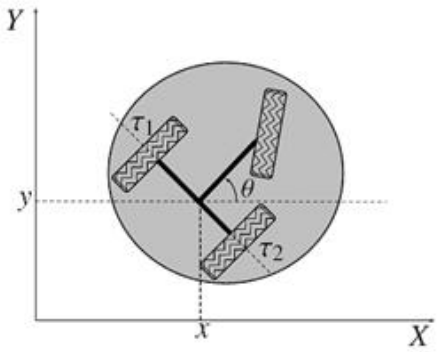


Fig. 2. Diagram of a wheeled mobile robot

It is assumed that the motion of the passive wheel can be ignored from the dynamics of the mobile robot, which is represented by the following set of equations [9]:

$$\dot{q} = \begin{vmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{vmatrix} \begin{vmatrix} v \\ w \end{vmatrix} \tag{1}$$

$$M(q)\dot{v} + V(q,\dot{q})v + G(q) = \tau$$

Where $q = [x, y, \theta]^T$ is the vector of generalized coordinates which describes the robot position, *(x,y)* are the Cartesian coordinates, which denote the mobile center of mass and θ is the angle between the heading direction and the *x*-axis (which is taken counterclockwise form); $v = [v, w]^T$ is the vector of velocities, *v* and *w* are the linear and angular velocities respectively; $\tau \in R^r$ is the input vector, $M(q) \in R^{nxn}$ is a symmetric and positive-definite inertia matrix, $V(q,\dot{q}) \in R^{nxn}$ is the centripetal and Coriolis matrix, $G(q) \in R^n$ is the gravitational vector. Equation (1.a) represents the kinematics or steering system of a mobile robot.

Notice that the no-slip condition imposed a non holonomic constraint described by (2), that it means that the mobile robot can only move in the direction normal to the axis of the driving wheels.

$$\dot{y}\cos\theta - \dot{x}\sin\theta = 0 \tag{2}$$

The control objective will be established as follows: Given a desired trajectory $q_d(t)$ and the orientation of the mobile robot we must design a controller that applies an adequate torque τ such that the measured positions $q(t)$ achieve the desired reference $q_d(t)$ represented as (3):

$$\lim_{t\to\infty}\|q_d(t)-q(t)\|=0 \tag{3}$$

To reach the control objective, the method is based on the procedure of [9], we are deriving a τ(t) of a specific vc(t) that controls the steering system (1.a) using a Fuzzy Logic Controller (FLC). A general structure of tracking control system is presented in Figure 3.

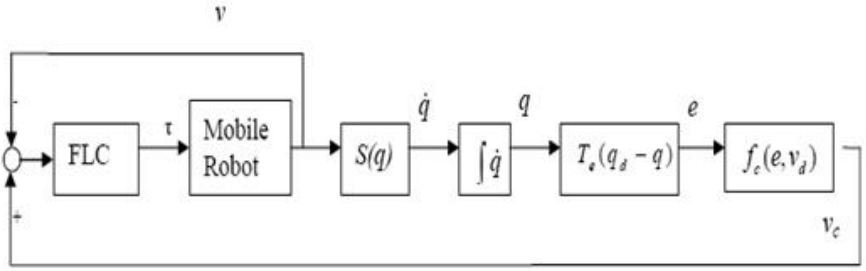


Fig. 3. Tracking control structure

The control is based on the procedure proposed by Kanayama et al. [13] and Nelson et al. [21] to solve the tracking problem for the kinematic model $v_c(t)$. Suppose that the desired trajectory q_d satisfies (4):

$$\dot{q}_d=\begin{vmatrix}\cos\theta_d & 0\\ \sin\theta_d & 0\\ 0 & 1\end{vmatrix}\begin{vmatrix}v_d\\ w_d\end{vmatrix} \tag{4}$$

Using the robot local frame (the moving coordinate system x-y in figure 1), the error coordinates can be defined as (5):

$$e=T_e(q_d-q),\begin{vmatrix}e_x\\ e_y\\ e_\theta\end{vmatrix}=\begin{vmatrix}\cos\theta & \sin\theta & 0\\ -\sin\theta & \cos\theta & 0\\ 0 & 0 & 1\end{vmatrix}\begin{vmatrix}x_d-x\\ y_d-y\\ \theta_d-\theta\end{vmatrix} \tag{5}$$

And the auxiliary velocity control input that achieves tracking for (1.a) is given by (6):

$$v_c=f_c(e,v_d),\begin{vmatrix}v_c\\ w_c\end{vmatrix}=\begin{vmatrix}v_d+\cos e_\theta+k_1e_x\\ w_d+v_dk_2e_y+v_dk_3\sin e_\theta\end{vmatrix} \tag{6}$$

Where k_1, k_2 and k_3 are positive gain constants.

The first part for this work is to apply the proposed method to obtain the values of k_i (i = 1, 2, 3) for achieving the optimal behavior of the controller, and the second part is to optimize the fuzzy controller.

4 Fuzzy Logic Controller

The purpose of the fuzzy logic controller (FLC) is to find a control input τ such that the current velocity vector v is able to reach the velocity vector v_c and this is denoted as:

$$\lim_{t\to\infty} \|v_c - v\| = 0 \tag{7}$$

The inputs variables of the FLC correspond to the velocity errors obtained of (10) (denoted as e_v and e_w: linear and angular velocity errors respectively), and 2 outputs variables, the driving and rotational input torques τ (denoted by F and N respectively). The initial membership functions (MF) are defined by 1 triangular and 2 trapezoidal functions for each variable involved. Figure 4 and Figure 5 depicts the MFs in which N, Z, P represent the fuzzy sets (Negative, Zero and Positive respectively) associated to each input and output variable.

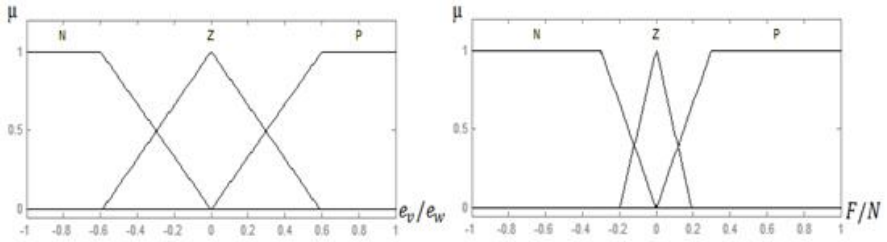


Fig. 4. Membership functions of the (a) input e_v and e_w, and (b) output variables *F* and *N*

The rule set of the FLC contain 9 rules, which govern the input-output relationship of the FLC and this adopts the Mamdani-style inference engine. We use the center of gravity method to realize defuzzification procedure. In Table 2, we present the rule set whose format is established as follows:

Rule i: If e_v is G1 and e_w is G2 then F is G3 and N is G4

Where G1…G4 are the fuzzy sets associated to each variable and i= 1 ... 9. In this case, P denotes “Positive”, N denotes “Negative”, and Z denotes “Zero”.

Table 2. Fuzzy rule set

e_v/e_w	N	Z	P
N	N/N	N/Z	N/P
Z	Z/N	Z/Z	Z/P
P	P/N	P/Z	P/P

5 Experimental Results

Several tests of the chemical optimization paradigm were made to test the performance of the tracking controller. First, we need to find the values of k_i (i = 1, 2, 3) showed in equation 6, which shall guarantee convergence of the error e to zero.

For this step of the application, only the decomposition reaction mechanism was triggered and the decomposition factor was varied; this factor is the quantity of resulting elements after applying a decomposition reaction to a determined "compound"; the only restriction here is that let x be the selected compound and x_i' (i=*1 2, …, n*), the resulting elements; the sum of all values found in the decomposition must be equal to the value of the original compound. This is shown in equation 12.

$$\sum_{i=1}^{n} x_i' = x \tag{12}$$

The test parameters can be observed in Table 3. For statistical purposes, every experiment was executed 35 times.

The decomposition rate represents the percentage of the pool to be candidate for the decomposition and the selection was made using a stochastic universal sampling technique, which uses a single random value to sample all of the solutions by choosing them at evenly spaced intervals.

Table 3. Parameters of the Chemical Reaction Optimization

No.	Elements	Iterations	Dec. Factor	Dec. Rate
1	2	10	2	0.3
2	5	10	3	0.3
3	2	10	2	0.4
4	2	10	3	0.4
5	5	10	2	0.4
6	5	10	3	0.4
7	5	10	2	0.5
8	10	10	2	0.5

As mentioned in section 2, an elitist reinsertion strategy was applied, keeping the compounds/elements with better performance through all the iterations, unless new elements/compounds with better performance are generated.

In example, for a pool containing 5 initial compounds, the vector length of decomposed elements when the decomposition factor is 3 and the decomposition rate is 0.4 will be of 6 elements. After applying this criteria, the need of increasing the initial element pool size was not necessary. Table 4 shows the results after applying the chemical optimization paradigm.

Table 4. Experimental Results of the proposed method for optimizing the values of the gains k_1, k_2, k_3

No.	Best Error	Mean	k_1	k_2	k_3
1	0.0086	1.1568	519	46	8
2	4.79e-004	0.1291	205	31	31
3	0.0025	0.5809	36	328	88
4	0.0012	0.5589	2	206	0
5	0.0035	0.0480	185	29	5
6	8.13e-005	0.0299	270	53	15
7	0.0066	0.1440	29	15	0
8	0.0019	0.1625	51	3	0

Figure 5 shows the resulting simulation obtained using Simulink for experiment No. 3, which is the best overall result so far, considering the average error and the position errors in *x*, *y* and *theta*.

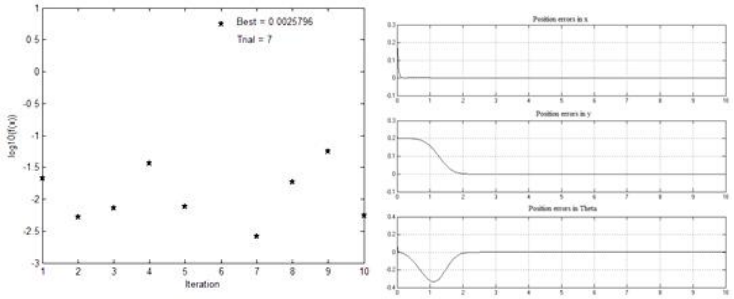


Fig. 5. (a) Convergence and (b) *x, y* and *theta* position errors of simulation No. 3

In previous work [22], the gain constant values were found by means of genetic algorithms. In Table 4 we have a comparison of the best results obtained with both algorithms.

Figure 6 shows the result in Simulink for the experiment with the best overall result, applying GAs as optimization method. We can observe that the result with the chemical optimization outperforms the GA in finding the best gain values.

Once we have found optimal values for the gain constants, the next step is to find the optimal values for the input/output membership functions of the fuzzy controller. Our goal is that in the simulations, the lineal and angular velocities reach zero.

Table 5 shows the parameters used in the first set of simulations and Figure 7 shows the behavior of the chemical optimization algorithm throughout the experiment.

Table 5. Comparison of the Best Results

Parameters	Genetic Algorithm	Chemical Optimization Algorithm
Individuals	5	2
Iterations	15	10
Crossover Rate	0.8	N/A
Mutation Rate	0.1	N/A
Decomposition Rate	N/A	0.4
Decomposition Factor	N/A	3
k1, k2, k3	43, 493, 19	36, 328, 88
Final Error	0.006734	0.0025796

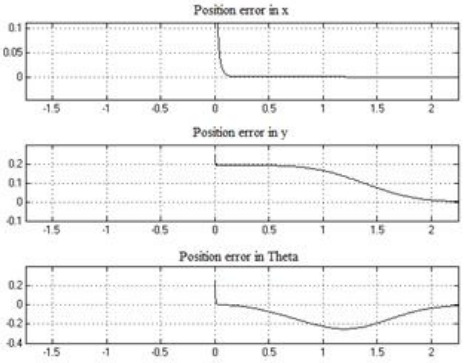


Fig. 6. Best experiment using Gas

Table 6. Parameters of the simulations for Type-1 FLC

Parameters	Value
Elements	10
Trials	15
Selection Method	Stochastic Universal Sampling
k_1	36
k_2	328
k_3	88
Error	0.077178

Figure 8 shows the resulted input and output membership functions found by the proposed optimization algorithm.

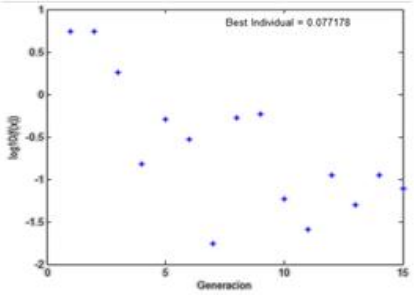


Fig. 7. Best simulation of experiments with the chemical optimization method

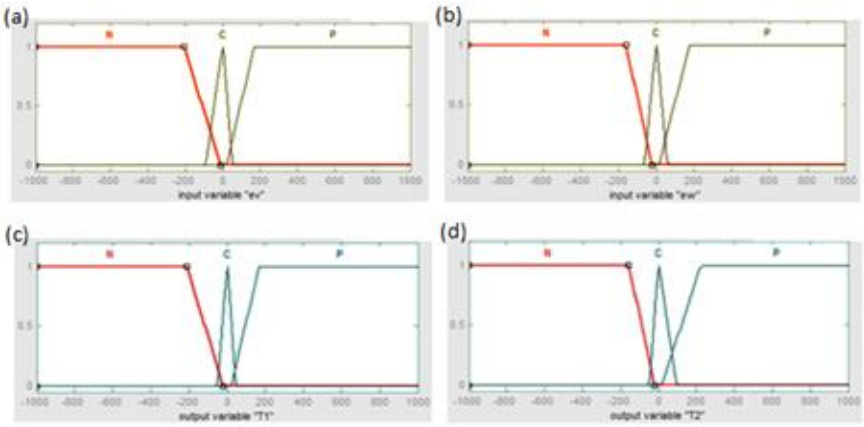


Fig. 8. Resulting input membership functions: (a) linear and (b) angular velocities and output (c) right and (d) left torque

Figure 9 (a) shows the obtained trajectory when simulating the mobile control system including the obtained input and output membership functions; (b) shows the best trajectory reached by the mobile when optimizing the input and output membership functions using genetic algorithms.

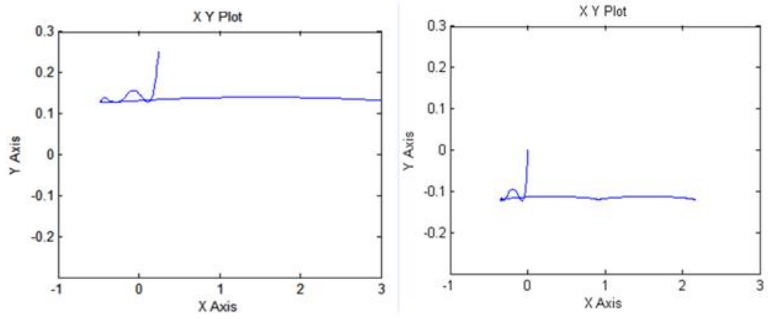


Fig. 9. (a) Obtained trajectory when applying the chemical reaction algorithm, (b) Obtained trajectory using genetic algorithms

A Type-2 fuzzy logic controller was developed using the parameters of the membership functions found for the FLC type-1. The parameters searched with the chemical reaction algorithm were for the footprint of uncertainty (FOU).

Table 7 shows the parameters used in the simulations and Figure 10 shows the behavior of the chemical optimization algorithm throughout the experiment.

Figure 11 shows the resulting type-2 input and output membership functions found by the proposed optimization algorithm and Figure 11 shows the obtained trajectory reached by the mobile robot.

Table 7. Parameters of the simulations for Type-2 FLC

Parameters	Value
Elements	10
Trials	10
Selection Method	Stochastic Universal Sampling
k_1	36
k_2	328
k_3	88
Error	2.7736

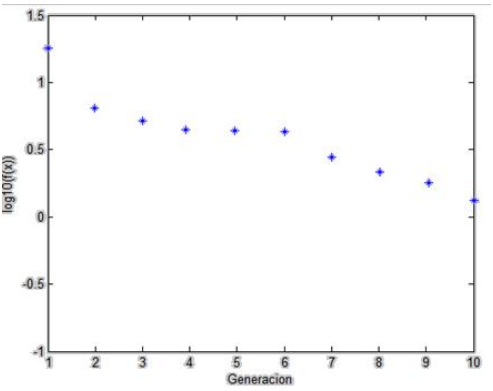


Fig. 10. Behavior of the algorithm when optimizing the type-2 FLC

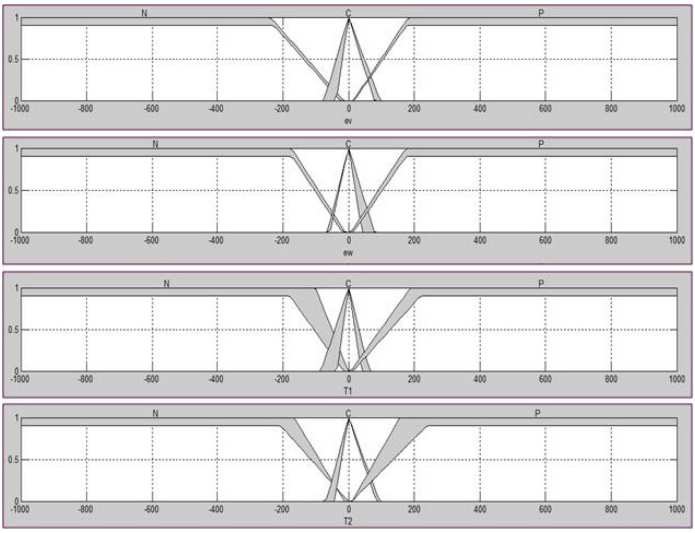


Fig. 11. Resulting type-2 input membership functions, from top to bottom: (a) linear and (b) angular velocities and output (c) right and (d) left torque

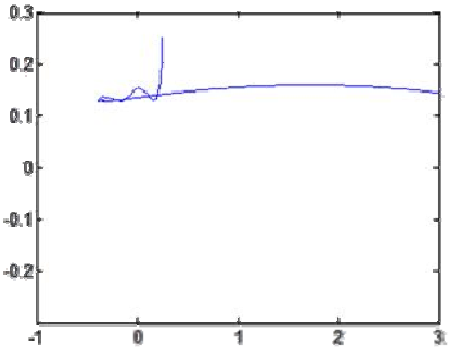


Fig. 12. Obtained trajectory for the mobile robot when applying the chemical reaction algorithm to the type-2 FLC

As observed in Table 7, the final error obtained is not smaller that the final error found for the type-1 FLC. Despite this, the trajectory obtained and showed in Figure 12 is acceptable taking into account that the reference trajectory is a straight line. In Figure 13 we can observe an "unacceptable" trajectory that was found in the early attempts of optimization for the type-1 FLC applying this chemical reaction algorithm. Here, we can observe that the parameters found were not the adequate to make the FLC follow the desired trajectory.

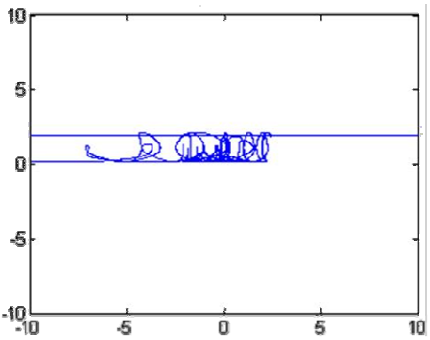


Fig. 13. Unaccepted resulting trajectory in early optimization trials

In order to test the robustness of the type-1 and type-2 FLC, we added an external signal given by equation (13).

$$F_{ext}(t) = \varepsilon \times \sin \omega \times t \tag{13}$$

This represents an external force applied in a period of 10 seconds to the obtained trajectory that will make the mobile robot to be out of its path. The idea of adding this disturbance is to measure the errors obtained with the FLC and to test the behavior of the mobile robot under perturbed torques. Table 8 shows the parameters for the simulations and the errors obtained during the run of the simulation.

Table 8. Simulation parameters and errors obtained under disturbed torques

ε	Velocity errors	Type-1 (GA)	Type-1 (CRA)	Type-2 (CRA)
0.05	Final error	4.0997	0.9815	29.5115
	Average error	4.1209	1.5823	26.6408
5	Final error	4.1059	0.9729	29.52
	Average error	3.1695	1.8679	26.1646
10	Final error	4.1045	0.9745	29.51
	Average error	3.0985	1.7438	24.9467
30	Final error	4.0912	0.9783	29.51
	Average error	2.2632	1.9481	24.6032
32	Final error	3273	0.9748	29.52
	Average error	3.4667e+003	2.8180	24.6465
34	Final error	1.5705e+004	566.8	29.51
	Average error	1.1180e+004	215.8198	24.9211
40	Final error	2.534e+004	3.5417e+04	29.51
	Average error	186.0611	5.7492e+003	23.8938
41	Final error	8839	3168	685.1
	Average error	2.0268e+004	0.0503e+003	16.5257

Figure 14 show the obtained trajectories for the type-1 FLC optimized with Genetic Algorithms.

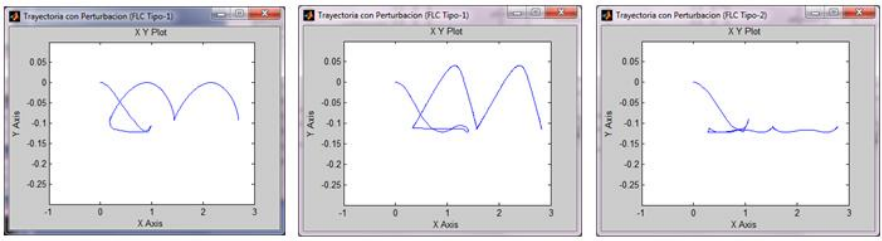


Fig. 14. From left to right, trajectory obtained with the type-1 FLC optimized with GA's. (a) ε = 30, (b) ε = 32, (c) ε = 34.

Figure 15 shows the obtained trajectories for the type-1 FLC optimized with the chemical reaction algorithm.

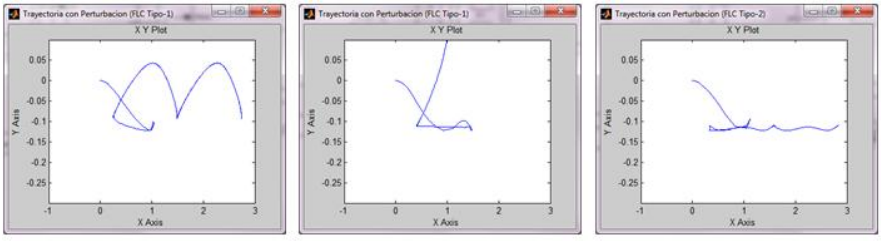


Fig. 15. From left to right, trajectory obtained with the type-1 FLC optimized with CRA. (a) ε = 30, (b) ε = 32, (c) ε = 34.

Figure 16 shows the obtained trajectories for the type-2 FLC optimized with Genetic Algorithms.

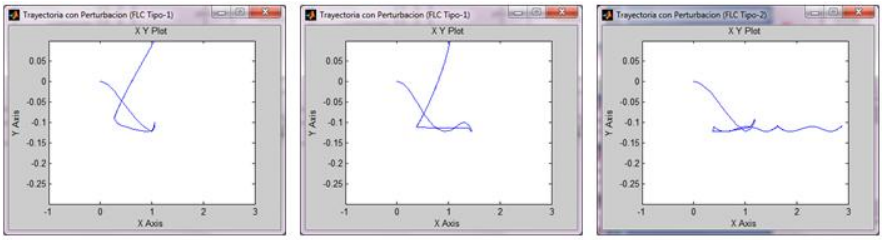


Fig. 16. From left to right, trajectory obtained with the type-2 FLC optimized with CRA. (a) ε = 30, (b) ε = 32, (c) ε = 34.

When observing Table 8 and Figures 14 to 16 we can observe that the type-2 FLC was able to maintain a more controlled trajectory in despite of the "large" error found by the algorithm (e=2.7736). For larger epsilon (ε) values, it was difficult for the type-1 FLC's to keep in the path and in a determined time, the controller was not able to return to the reference trajectory.

6 Conclusions

In this paper, we presented simulation results from an optimization method that mimics chemical reactions applied to the problem of tracking control. The goal was to find the gain constants involved in the tracking controller for the dynamic model of a unicycle mobile robot. In the figures of the experiments we are able to note de behavior of the algorithm and the solutions found through all the iterations. Simulation results show that the proposed optimization method is able to outperform the results previously obtained applying a genetic algorithm optimization technique. The optimal fuzzy logic controller obtained with the proposed chemical paradigm has been able to reach smaller error values in less time than genetic algorithms. Also, the type-2 fuzzy controller was able to perform better under the presence of disturbance for this problem in despite of the "large" error obtained (e=2.7736). The design of optimal type-2 fuzzy controllers is being performed at the time.

Acknowledgement. The authors would like to thank CONACYT and Tijuana Institute of Technology for the facilities and resources granted for the development of this research.

References

[1] Aliev, R.A., Pedrycz, W., Guirimov, B.G., Aliev, R.R., Ilhan, U., Babagil, M., et al.: Type-2 fuzzy neural networks with fuzzy clustering and differential evolution optimization. Information Sciences 181(9), 1591–1608 (2011)

[2] Astudillo, L., Castillo, O., Aguilar, L.: Intelligent Control for a Perturbed Autonomous Wheeled Mobile Robot: a Type-2 Fuzzy Logic Approach. Nonlinear Studies 14(1) (2007)

[3] Bentalba, S., El Hajjaji, A., Rachid, A.: Fuzzy Control of a Mobile Robot: A New Approach. In: Proc. IEEE Int. Conf. on Control Applications, Hartford, CT, pp. 69–72 (October 1997)

[4] Bloch, A.M., Drakunov, S.: Tracking in NonHolonomic Dynamic System Via Sliding Modes. In: Proc. IEEE Conf. on Decision & Control, Brighton, UK, pp. 1127–1132 (1991)

[5] Campion, G., Bastin, G.: D'Andrea-Novel, B., Structural Properties and Classification of Kinematic and Dynamic Models of Wheeled Mobile Robots. IEEE Trans. on Robotics and Automation 12(1) (February 1996)

[6] Castillo, O., Martinez-Marroquin, R., Melin, P., Soria, J.: Comparative Study of bio-inspired algorithms applied to the optimization of type-1 and type-2 fuzzy controllers for an autonomous mobile robot. In: Bio-inspired Hybrid Intelligent Systems for Image Analysis and Pattern Recognition. SCI, vol. 256, pp. 247–262 (2009)

[7] Cazarez-Castro, N.R., Aguilar, L.T., Castillo, O.: Fuzzy logic control with genetic membership function parameters optimization for the output regulation of a servomechanism with nonlinear backlash. Expert Systems with Applications 37(6), 4368–4378 (2010)
[8] Chwa, D.: Sliding-Mode Tracking Control of Nonholonomic Wheeled Mobile Robots in Polar coordinates. IEEE Trans. on Control Syst. Tech. 12(4), 633–644 (2004)
[9] Fierro, R., Lewis, F.L.: Control of a Nonholonomic Mobile Robot: Backstepping Kinematics into Dynamics. In: Proc. 34th Conf. on Decision & Control, New Orleans, LA (1995)
[10] Fierro, R., Lewis, F.L.: Control of a Nonholonomic Mobile Robot Using Neural Networks. IEEE Trans. on Neural Networks 9(4), 589–600 (1998)
[11] Fukao, T., Nakagawa, H., Adachi, N.: Adaptive Tracking Control of a Non-Holonomic Mobile Robot. IEEE Trans. on Robotics and Automation 16(5), 609–615 (2000)
[12] Ishikawa, S.: A Method of Indoor Mobile Robot Navigation by Fuzzy Control. In: Proc. Int. Conf. Intell. Robot. Syst., Osaka, Japan, pp. 1013–1018 (1991)
[13] Kanayama, Y., Kimura, Y., Miyazaki, F., Noguchi, T.: A Stable Tracking Control Method For a Non-Holonomic Mobile Robot. In: Proc. IEEE/RSJ Int. Workshop on Intelligent Robots and Systems, Osaka, Japan, pp. 1236–1241 (1991)
[14] Kolmanovsky, I., McClamroch, N.H.: Developments in Nonholonomic Non-trol Problems. IEEE Control Syst. Mag. 15, 20–36 (1995)
[15] Lee, T.-C., Lee, C.H., Teng, C.-C.: Tracking Control of Mobile Robots Using the Backsteeping Technique. In: Proc. 5th. Int. Conf. Contr., Automat., Robot. Vision, Singapore, pp. 1715–1719 (December 1998)
[16] Lee, T.H., Leung, F.H.F., Tam, P.K.S.: Position Control for Wheeled Mobile Robot Using a Fuzzy Controller. IEEE, 525–528 (1999)
[17] Lee, T.-C., Tai, K.: Tracking Control of Unicycle-Modeled Mobile robots Using a Saturation Feedback Controller. IEEE Trans. on Control Systems Tech-nology 9(2), 305–318 (2001)
[18] Martinez, R., Castillo, O., Aguilar, L.: Optimization of type-2 fuzzy logic controllers for a perturbed autonomous wheeled mobile robot using genetic algorithms. Information Sciences 179(13), 2158–2174 (2009)
[19] Meyer, T., Yamamoto, L., Banzhaf, W., Tschudin, C.: Elongation Control in an Algorithmic Chemistry. In: Kampis, G. (ed.) ECAL 2009, Part I. LNCS, vol. 5777, pp. 273–280. Springer, Heidelberg (2011)
[20] Mohammadi, S.M.A., Gharaveisi, A.A., Mashinchi, M., Vaezi-Nejad, S.M.: An evolutionary tuning technique for type-2 fuzzy logic controller. Transactions of the Institute of Measurement and Control 33(2), 223–245 (2011)
[21] Nelson, W., Cox, I.: Local Path Control for an Autonomous Vehicle. In: Proc. IEEE Conf. on Robotics and Automation, pp. 1504–1510 (1988)
[22] Oh, S., Jang, H., Pedrycz, W.: A comparative experimental study of type-1/type-2 fuzzy cascade controller based on genetic algorithms and particle swarm optimization. Expert Systems with Applications 38(9), 11217–11229 (2011)
[23] Pawlowski, S., Dutkiewicz, P., Kozlowski, K., Wroblewski, W.: Fuzzy Logic Implementation in Mobile Robot Control. In: 2nd Workshop on Robot Motion and Control, pp. 65–70 (October 2001)
[24] Sahab, A.R., Moddabernia, M.R.: Backstepping method for a single-link flexible-joint manipulator using genetic algorithm. In: IJICIC 2011, vol. 7(7(B)), pp. 4161–4170 (2011)

[25] Shi, N.-Y., Chu, C.-P.: A molecular solution to the hitting-set problem in DNA-based supercomputing. Information Sciences 180, 1010–1019 (2010)
[26] Song, K.T., Sheen, L.H.: Heuristic fuzzy-neural Network and its application to reactive navigation of a mobile robot. Fuzzy Sets Systems 110(3), 331–340 (2000)
[27] Tsai, C.-C., Lin, H.-H., Lin, C.-C.: Trajectory Tracking Control of a Laser-Guided Wheeled Mobile Robot. In: Proc. IEEE Int. Conf. on Control Applications, Taipei, Taiwan, pp. 1055–1059 (September 2004)
[28] Ulyanov, S.V., Watanabe, S., Ulyanov, V.S., Yamafuji, K., Litvintseva, L.V., Rizzotto, G.G.: Soft Computing for the Intelligent Robust Control of a Robotic Unicycle with a New Physical Measure for Mechanical Controllability. In: Soft Computing, vol. 2, pp. 73–88. Springer (1998)
[29] Xu, J., Lam, A.Y.S., Li, V.O.K.: Chemical Reaction Optimization for the Grid Scheduling Problem. In: ICC 2010, pp. 1–5. IEE Communication Society (2010)
[30] Yamamoto, L.: Evaluation of a Catalytic Search Algorithm. In: Proc. 4th Int. Workshop on Nature Inspired Cooperative Strategies for Optimization, NICSO 2010, pp. 75–87 (2010)
[31] Yu, J., Ma, Y., Chen, B., Yu, H., Pan, S.: Adaptive Neural Position Tracking Control for Induction Motors via Backstepping. In: IJICIC 2011, vol. 7(7(B)), pp. 4503–4516 (2011)

Evolutionary Optimization of the Fuzzy Integrator in a Navigation System for a Mobile Robot

Abraham Meléndez and Oscar Castillo

Tijuana Institute of Technology, Tijuana México
abraham.ms@gmail.com, ocastillo@tectijuana.mx

Abstract. This paper describes the optimization of an Integrator control block within the proposed navigation control system for a mobile robot. The control blocks that the integrator will combine are two Fuzzy Inference Systems (FIS) in charge of tracking and reaction respectively. The integrator block is call Weighted Fussy Inference System (WFIS), and assigns weights to the responses on each behavior block, to combine them into a single response.

1 Introduction

The use of mobile robots has increased over the last decades in many areas from industrial work to research and household and one reason for this is that they have proved useful in each of these areas from doing very specific task to ongoing monotonous shores, they help their human counterpart be more productive and efficient. Also as hardware technology is moving forward and developing more capable robots at lower cost, this is another reason for this increase and why we are seeing them in more common places.

The mobile robot, needs to move around its environment and this is why a great deal of research has been invested on testing them with control systems that allow the robots to navigate on their own, and different methodologies have been applied from traditional control such as PD, PID [4, 9] to soft computing methods like Fuzzy Logic [10, 25, 13, 17, 15, 18, 12, 11, 21, 5, 19, 20], Neural Networks [11] and hybrid ones also [6,23,24].

In this paper, the navigation control system has been designed to combine two key behaviors that are considered to be required for any navigation control system of a mobile robot. The first one is a tracking controller, this is an obvious one since there is no point of having a navigation system on a robot, that can´t go to a desired location, the second one is a reactive controller and here we considered this one to be off great importance also, since the tracking controller can get the robot to the destination, but that will be on an ideal situation where there are no obstacles present on the robots path.

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 21–31.
springerlink.com

The reactive controller is for those cases where an obstacle free path cannot be guaranteed; this is where the reactive controller will do its work providing a behavior that will make the robot react to any type of obstacle so that the robot can continue on its journey. In this paper we describe the integration method for these two controls as part of the complete Navigation Control System, the control blocks are fuzzy inference systems of type-1 and type-2, and a general GA (Genetic Algorithm) is applied to the optimization of each of the controller blocks with a specific fitness function for each part that will evaluate the corresponding individual performance.

As related work, we can find that of Cupertino et al [8] developed a Fuzzy controller of a mobile robot, based on 3 FLCs (Fuzzy Logic Controller) and one Fuzzy Supervisor that was in charge of determining which FLC behavior will be active, there the FLCs are of Type-1 and the Fuzzy Supervisor mainly acts as a switch. In our proposed method the fuzzy integrator acts more like a fusion block. S. Coupland et al [7] proposed a Type-2 Fuzzy Control of a Mobile Robot, which is based on W Payton et al [22] Command Fusion, where the idea is that a behavior should work with others to find a mutually beneficent solutions, where each behavior takes into consideration every possible output with its corresponding activation value (positive or negative), and a winner takes all network is use to select the winning responses for each behavior. Coupland suggests using two FISs one for goal seeking and another for obstacle avoidance. The activation value for each of the FIS output will be a Fuzzy set that will be passed to the command fusion block to later be defuzzifed and that crisp value pass to the Actuator block, being a difference with our proposed control the integration method of the two behaviors. The control navigation of a mobile robot is a topic that has been extensively investigated over the years, the method proposed in this paper is based on the idea that separation and the cooperation between key behaviors produces a better result than the use of a single behavior and it differs from previous approaches from the integration perspective done by a FIS that is in charge of the weighted system, that will assign a weight to each response from each controller by each control step that is combined to obtain a unified single response to the robot.

This paper is organized as follows: In section 2 we describe the mobile robot used in these experiments, section 3 describes the development of the evolutionary method. Section 4 shows the simulation results. Finally, section 5 shows the Conclusions.

2 Mobile Robot

The particular mobile robot considered in this work. The robot is based on the description of the Simulation toolbox for mobile robots [26], which assumes a wheeled mobile robot consisting of one conventional, steered, unactuated and not-sensed wheel, and two conventional, actuated, and sensed wheels (conventional wheel chair model). This type of chassis provides two DOF (degrees of freedom) locomotion by two actuated conventional non-steered wheels and one unactuated steered wheels. The Robot has two degrees of freedom (DOFs): y-translation and

either x-translation or z-rotation [26]. Fig. 1 shows the robot's configuration, it has 2 independent motors located on each side of the robot and one castor wheel for support located at the front of the robot.

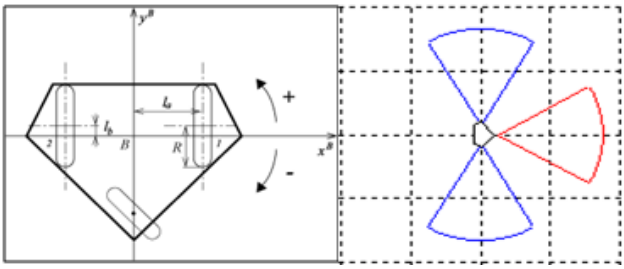


Fig. 1. Kinematic coordinate system assignments[26]

The kinematic equations of the mobile robot are as follows:

Eq. 1: The sensed forward velocity solution [26]

$$\begin{pmatrix} V_{B_x} \\ V_{B_y} \\ \omega_{B_z} \end{pmatrix} = \frac{R}{2l_a} \begin{bmatrix} -l_b & l_b \\ -l_a & -l_a \\ -1 & -1 \end{bmatrix} \begin{pmatrix} \omega_{W_1} \\ \omega_{W_2} \end{pmatrix} \tag{1}$$

Eq. 2: The Actuated Inverse Velocity Solution [26]

$$\begin{pmatrix} \omega_{W_1} \\ \omega_{W_2} \end{pmatrix} = \frac{1}{R(l_b^2+1)} \begin{bmatrix} -l_a l_b & -l_b^2 - 1 & -l_a \\ l_a l_b & -l_b^2 - 1 & l_a \end{bmatrix} \begin{pmatrix} V_{B_x} \\ V_{B_y} \\ \omega_{B_z} \end{pmatrix} \tag{2}$$

Under the Metric system are define as:

V_{B_x}, V_{B_y} Translational velocities $[\frac{m}{s}]$,
ω_{B_z} Robot z-rotational velocity $[\frac{rad}{s}]$,
ω_{W_1}, ω_{W_1} Wheel rotational velocities $[\frac{rad}{s}]$,
R Actuated wheel radius[m],
l_a , l_b Distances of wheels from robot's axes [m].

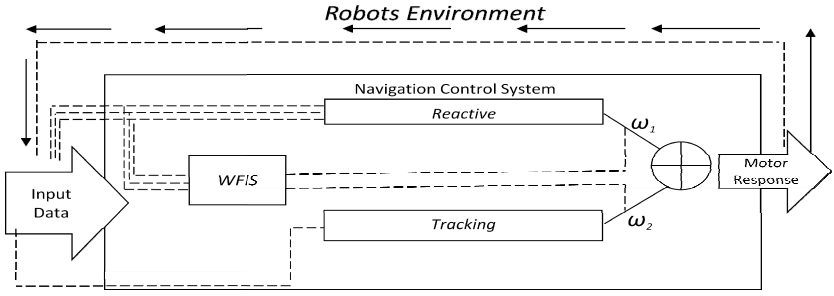
3 Navigation Control System

The proposed control system consists of three main fuzzy blocks, two are behavior based and the other one is in charge of the response integration, the behaviors are the reactive and tracking blocks, and each one will provide its specific behavior that will be combined into one response by the integration block.

Each behavior block is in charge of its own task, the problem is that they seem to be in conflict with each other when an unexpected obstacle arises, because if at the time of planning the route the obstacles are present then the route can be designed to avoid them, but when there are obstacles that we where un aware off, the two behaviors enter in contradiction one is designed to avoid the object and the other to keep the robot on its track.

The most common solution will be to just switch between controllers when need it, however, this approach is not very efficient due to the lack of awareness the two blocks have of each other, the reactive will effectively keep the robot from the collision but it may redirect the robot farther away from its destination to a point where the tracking controller can no longer find its way back to the reference, or the tracking controller can guide the robot straight into the obstacle if the reactive control is not activated on time. The proposed referral for control navigation is to always have both controls active and their responses are combined and generate the movement of the robot, the integration is done with another fuzzy block call WFIS[15] (Weight-Fuzzy Inference System) and what this controller does is to assign response weights to each of the controllers crisp response value.

The inputs are gathered from the information that we can collect from the robot (sensors) or the environment by other means (cameras) and from this we need to create the knowledge rule base to give higher activation values to the response we want to take the lead on the robot movement one example of the rule is the following (*if Front_Sensor_Distance is Close Then TranckingWeight is Medium and ReactiveWeight is Medium*), both off our controls provide the right and left motor speed and we combine each one with the weight given by the WFIS block. Figure 2 shows the proposed navigation control.



$$RMSR = \frac{\left[\left(\frac{\sum_i \mu_A(y_i)\times y_i}{\sum_i \mu_A(y_i)}\right)*\omega_1\right] + \left[\left(\frac{\sum_i \mu_A(y'_i)\times y'_i}{\sum_i \mu_A(y'_i)}\right)*\omega_2\right]}{2}$$

$$LMSR = \frac{\left[\left(\frac{\sum_i \mu_A(y'''_i)\times y'''_i}{\sum_i \mu_A(y'''_i)}\right)*\omega_1\right] + \left[\left(\frac{\sum_i \mu_A(y''_i)\times y''_i}{\sum_i \mu_A(y''_i)}\right)*\omega_2\right]}{2}$$

Fig. 2. Navigation Control System [15]

4 Genetic Algorithms

The Genetic Algorithm (GA) was applied to each of the design problems, of finding the best fuzzy reactive and tracking controllers [16]; however this paper will only focus on the WFIS controller.

The purpose of using an evolutionary method is to find the best possible controllers of each type and this can be obtained using the GA, as it searches along the solution space, combining the attributes from the best controllers in generating new ones, this concept taken from the building blocks theory.

The idea was to optimize the parameters in the Membership Functions, but also the number of Membership functions and this means to also optimize the number of rules making this a multi objective problem. For this we will take advantage of the HGA (Hierarchical Genetic Algorithm) intrinsic characteristic to solve multi objective problems.

The work of the GA was divided in two main modules, one that handles all the operations related to the selection and chromosome manipulation, which we use for all our controllers that we work on, the other module is the one where we evaluated the performance of each chromosome and this part is different on each case. With this approach we utilize the generality of the GA and just have a specific evaluation method for each controller. Figure 3 shows the 2 main modules.

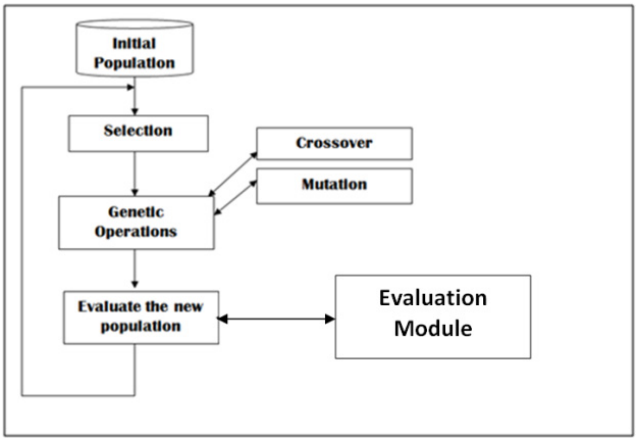


Fig. 3. Genetic Algorithm process

The GA module is in charge of initializing the population, selecting the chromosomes that will be used for the genetic operations and letting the Evaluation Module know which chromosomes are ready to be evaluated and reinserting them to the population pool.

4.1 Chromosome Encoding

Each individual on the population will represent a FIS controller, each of which will be encoded on a vectorial structure that will have “n” main sections, one for each variable (input and output). Each main section will contain 2 subsections (control genes, Connection genes). The section and subsection sizes depend on the controller that they represent.

4.2 WFIS Controller

The function of the WFIS control is to correctly combine the 2 behaviors of tracking and reaction and obtain a new global behavior that resembles the same ability that we apply when we are driving, that is to react to unexpected objects, but in a more basic concept and ability, to the problem that is the navigation of the robot. A forward moving behavior response out off the global control is desired. The objective is to guide the robot through the reference avoiding any collision with any obstacle present. It’s not our objective to optimize the robot to find the maze exit, we use a closed space where the robot cannot easily wonder off and each wall is considered an obstacle to the robot that it must avoid while it moves around. The FISs are Mamdani type-1 fuzzy systems [14], each consisting of 3 inputs, which are the distances obtained by the robots sensors described on section 2, and 2 outputs that are the weights that will be used to integrate the responses of the other 2 controllers, all this information is encoded into each chromosome.

4.2.1 Type-1 Fuzzy Weight Controller Chromosome Architecture

The control genes consist of 5 bit vectors, this will indicate which fuzzy membership is or not active, the connection genes are divided in 5 subsections, 5 is the maximum number of membership functions that are allowed per variable, each of which can be trapezoidal or triangular membership function, and each of these subsection is divided into 2 sections one that indicates the type of the membership function and the other the parameters for the function, see Figure 4.

4.3 WFIS Controller Objective Function

The WFIS controller performance is measured with the RMSE between the reference and the robots trajectory. We apply the test three times and take the average, on each of the three tests the robot and the reference vertical position is random, but we ensure that on one test the robots vertical position is above the reference and on another test is below it, we do this to ensure the controller works properly for any case the robot may need it when it´s above or below (Figure 5).

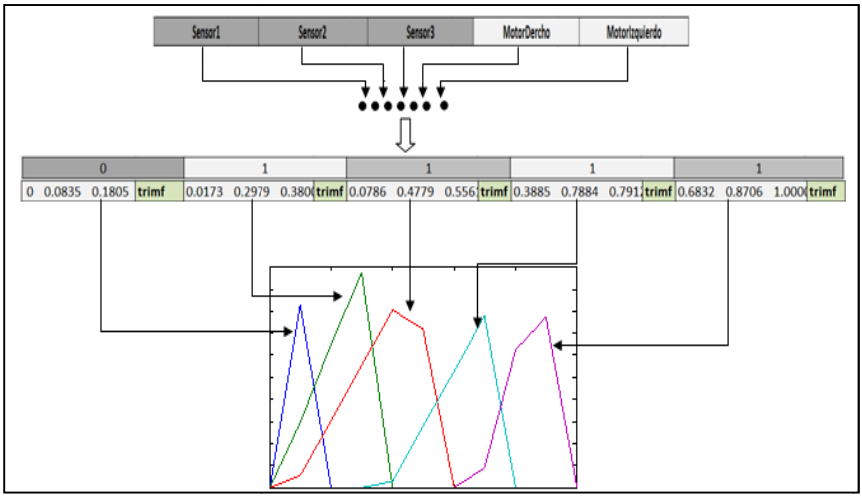


Fig. 4. Type-1 WFIS Controller Chromosome Architecture

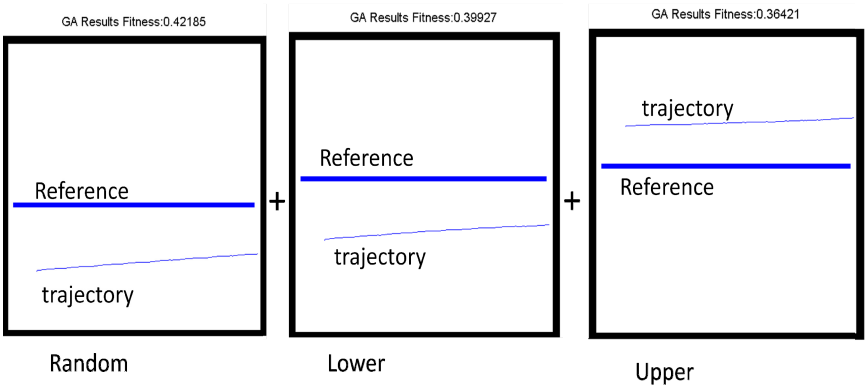


Fig. 5. Fitness Functions for the WFIS Controller

5 Simulation Results

For the simulation experiment the GA and the evaluation process were separated into two different parts, the generic GA process was developed on the C# language with .net 4, where a GA and Fuzzy System library where created with a GUI to setup the GA parameters, there the GA operations and cycle are run and the FIS are created. When a chromosome is ready to be evaluated it lets Matlab know and a modified version of the Simulation toolbox for mobile robots [26] is used to run each test, where the performance is measured and a Fitness value is returned to the GA process, and the communication between both process is done using a SQL server queue table.

5.1 WFIS Controller

For the type-1 WFIS controller, a GA was setup with high number of generations and a low value of population size, this because of the large solution space. The reasoning behind this is that with a relative small group of individuals it will cover focused sections of the solution and can move around the space, A constrain for inputs and outputs of maximum 10 and minimum 2 FMS was set, on the outputs, the evaluation as described on section 4 is based upon each individual performance on the particular maze problem.

Table 1. Summary of Type-1 WFIS Results

	Membership Chromosome		Fuzzy Rule Chromosome
	Control Genes	Connections Genes	
Representation	Binary	Real Number	Integer
Population Size	10		
No. of Offspring	3		
Crossover	One Point	One Point	
Crossover Rate	1.0	1.0	
Mutation	Bit Mutation	Random Mutation	Shift index operation
Mutation Rate	0.02	0.02	
GA Parameters			
Generation	1500		
Selection	Roulette Wheel with Ranking		
Results			
Rank	Fitness	Active FM's (S_1+S_2+S_3+W_1+W_2)	Active Rules
1	0.2393	(3+4+3+3+3)=16	36
2	0.2450	(3+4+3+3+3)=16	36
3	0.2514	(4+5+4+4+3)=20	80
4	0.2543	(4+3+3+4+3)=17	36
5	0.2551	(3+4+3+4+3)=17	36
6	0.2633	(4+4+3+4+4)=19	48
7	0.2643	(4+3+4+3+5)=19	48
8	0.2691	(3+3+3+4+3)=16	27
9	0.2742	(3+3+3+4+3)=16	27

Table 1 shows the GA configuration and the top 9 Results, where we have the fitness value and the number of membership functions of each input and output, where the S represents the inputs and indicate the sensor number and W the outputs and indicate the Weight number, and the total rules that are active on each controller.

Fig 6 shows the set of 3 tests during the evaluation process of the GA, where the red line is the reference, the blue dotted line is robot path on each run and the gray squares are obstacle located around the reference path.

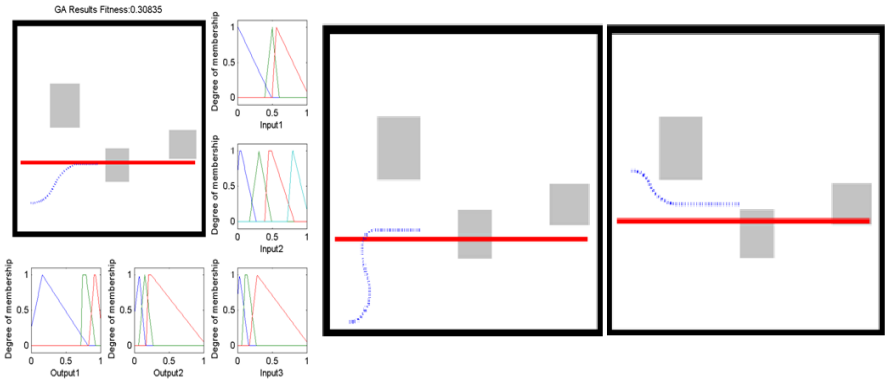


Fig. 6. Type-1 WFIS Controller Results

6 Conclusions

In this paper we have been able to optimize the Type-1 and Type-2 Reactive and Tracking Controllers and developed a GUI to optimize the Fuzzy Inference System, and we are currently working on the Type-1 WFIS Optimization.

Future work will consist in the Optimization of the WFIS based on a Type-2 fuzzy system.

Acknowledgment. We would like to express our gratitude to CONACYT, and Tijuana Institute of Technology for the facilities and resources granted for the development of this research.

References

[1] Aceves, A., Aguilar, J.: A Simplified Version of Mamdani's Fuzzy Controller The Natural Logic Controller. IEEE Transactions on Fuzzy Systems 14(1), 16–30 (2006)

[2] Aguilar, L., Melin, P., Castillo, O.: Intelligent Control of a stepping motor drive using a hybrid neuro-fuzzy ANFIS approach. Applied Soft Computing 3(3), 209–219 (2003)

[3] Astudillo, L., Castillo, O.T., Aguilar, L.: Intelligent Control of an Autonomous Mobile Robot Using Type-2 Fuzzy Logic. Engineering Letters 13(2), 93–97 (2006)

[4] Bell, M., Toriu, T., Nakajima, S.: Image-Based Robot Map Building and Path Planning with an Omnidirectional Camera Using Self-Organising Maps. International Journal of Innovative Computing, Information and Control, 3845–3852 (2011)

[5] Castillo, O., Melin, P.: New fuzzy-fractal-genetic method for automated mathematical Modeling and Simulation of Robotic Dynamic Systems. IEEE, International Conference on Fuzzy Systems, vol. 2, 1182–118 (1998)
[6] Castillo, O., Melin, P.: New fuzzy-fractal-genetic method for automated mathematical Modeling and Simulation of Robotic Dynamic Systems. In: IEEE, International Conference on Fuzzy Systems, vol. 2, pp. 1182–1118 (1998)
[7] Coupland, S.: Type-2 Fuzzy Control of a Mobile Robot. PhD Transfer Report. De Montfort University, UK (2003)
[8] Cupertino, F., Giordano, V., Naso, D., Delfine, L.: Fuzzy control of a mobile robot. IEEE Robotics & Automation Magazine, 74–81 (2006)
[9] Fate, M.: Robust Voltage Control of Electrical Manipulators in Task-Space. International Journal of Innovative Computing, Information and Control, 2691–2700 (2010)
[10] Ishikawa, S.: A Method of Indoor Mobile Robot Navigation by Fuzzy Control. In: Proc. Int. Conf. Intell. Robot. Syst., Osaka, Japan, pp. 1013–1018 (1991)
[11] Klir, J.G., Yuan, B.: Fuzzy Sets, Fuzzy Logic, and Fuzzy Systems: Selected Papers by Lotfi A. Zadeh. In: Advances in Fuzzy Systems: Application and Theory, vol. 6. World Scientific Publishing Company (1996)
[12] Kim, C., Lee, K.: Robust Control of Robot Manipulators Using Dynamic Compensators under Parametric Uncertainty. International Journal of Innovative Computing, Information and Control, 4129–4137 (2011)
[13] Leyden, M., Toal, D., Flanagan, C.: A Fuzzy Logic Based Navigation System for a Mobile Robot. In: Proceedings of Automatisierungs Symposium (1999)
[14] Mamdani, E.: Applications of fuzzy logic to approximate reasoning using linguistic synthesis. In: Proc. 6th Int. Symp. on Multiple Value Logic, pp. 196–202. Utah State University (1976)
[15] Melendez, A., Castillo, O., Soria, J.: Reactive and Tracking Control of a Mobile Robot in a Distributed Environment Using Fuzzy Logic. In: IEEE International Conference on FUZZ, pp. 1–5 (2010)
[16] Melendez, A., Castillo, O., Melin, P.: Evolutionary Optimization of the Fuzzy Controllers in a Navigation System for a Mobile Robot. International Journal of Innovative Computing, Information and Control, International Journal (in press)
[17] Meléndez, A., Castillo, O., Soria, J.: Reactive Control of a Mobile Robot in a Distributed Environment Using Fuzzy Logic. In: Fuzzy Information Processing Society, NAFIPS 2008. Annual Meeting of the North American, vol. (19-22), pp. 1–5 (2008)
[18] Melin, P., Castillo, O.: Intelligent Systems with Interval Type-2 Fuzzy Logic. International Journal of Innovative Computing, Information and Control, 771–784 (2008)
[19] Melin, P., Castillo, O.: Intelligent control of aircraft dynamic systems with a new hybrid neuro-fuzzy fractal approach. Information Sciences 142(1-4), 161–175 (2002)
[20] Melin, P., Castillo, O.: Adaptive Intelligent control of aircraft systems with a hybrid approach combining neural networks, fuzzy logic and fractal theory. Applied Soft Computing 3(4), 353–262 (2003)
[21] Pishkenari, H.N., Mahboobi, S.H., Meghdari, A.: On the Optimum Design of Fuzzy Logic Controller for Trajectory Tracking Using Evolutionary Algorithms. In: Cybernetics and Intelligent Systems, 2004 IEEE Conference on Publication Date: 1-3, vol. 1, pp. 660–665 (2004)
[22] Payton, D.W., Rosenblatt, J.K., Keirsey, D.M.: Plan guided reaction. IEEE Transactions on Systems, Man and Cybernetics 20(6), 1370–1382 (1990)

[23] Shafiei, S., Soltanpour, M.: Neural Network Sliding-Mode-PID Controller Design for Electrically Driven Robot Manipulators. International Journal of Innovative Computing, Information and Control, 511–524 (2011)
[24] Shafiei, S., Soltanpour, M.: Robust Task-Space Control of Robot Manipulators under Imperfect Transformation of Control Space. International Journal of Innovative Computing, Information and Control, 3949–3960 (2009)
[25] Thomson, A., Baltes, J.: A path following system for autonomous robots with minimal computing power. University of Auckland, Private Bag 92019, Auckland, New Zealand, Technical Report (2001)
[26] Mobile robotics toolbox for Matlab 5 (2001), http://www.uamt.feec.vutbr.cz/robotics/simulations/amrt/simrobot_en.html

Particle Swarm Optimization for Average Approximation of Interval Type-2 Fuzzy Inference Systems Design in FPGAs for Real Applications

Yazmin Maldonado, Oscar Castillo, and Patricia Melin

Tijuana Institute of Technology, Calzada Tecnológico S/N, Tijuana, México
yaz.maldonado@gmail.com, ocastillo@tectijuana.mx

Abstract. This paper proposes the particle swarm optimization (PSO) of type-2 membership functions for average approximation of an interval type-2 fuzzy logic system (AT2-FIS), the AT2-FIS is synthesized in VHDL code for FPGAs. The PSO method considers three objective functions, overshoot, undershoot and steady state error because the real application is to control the speed of a DC motor. Several experiments were performed to optimize the AT2-FIS in FPGA. Experiments were conducted by changing the number of bits for encoding the AT2-FLC in VHDL.

1 Introduction

Fuzzy inference systems are based on rules; these rules incorporate linguistic variables, linguistic terms and fuzzy rules. The acquisition of rules is not an easy task for the expert and is of vital importance in the operation of the controller. The process of adjusting these linguistic terms and rules is usually done by trial and error, which implies a difficult task, and for this reason there have been methods proposed to optimize those elements that over time have taken importance, such as particle swarm optimization [1].

Most of the fuzzy logic applications with physical systems require a real time operation, the simple way to implement these systems is to realize them as software programs on a personal computer or higher density programmable logic devices, such as the field programmable gate array (FPGA).

The research of different optimization techniques for type-2 fuzzy systems have increased, however there is the problem of runtime, and the runtime decreases when the implementation is processed in parallel, as in the FPGA. There are some works related to the optimization of a particular problem [4][7][9].

This paper explains the design of T2-MFs optimization of the AT2-FLC for regulation speed of a DC motor (ReSDCM) in FPGA, based on an a-verage

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 33–49.
springerlink.com

approximation of intervals of type-2 fuzzy systems method [8]. The main goal of this paper is to compare the results (average errors, runtime and resolution for number of bits) of the AT2-FLC optimized with PSO.

The proposed methodology for this paper is to synthesize the AT2-FLC in FPGA, the optimized parameters of the T2-MFs are sent to the FPGA, once the AT2-FLC was optimized, the optimization process is disconnected from the PC and the AT2-FLC is ready for use. Figure 1 shows the methodology diagram used for the optimization of T2-MFs for AT2-FLC in the FPGA.

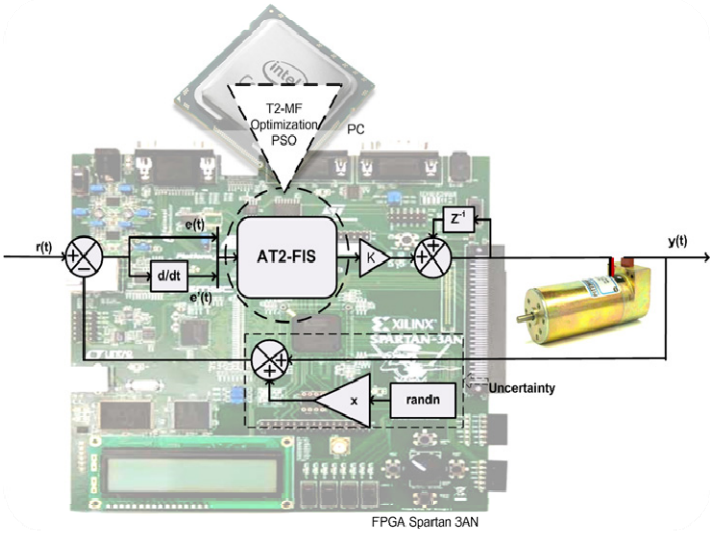


Fig. 1. Fuzzy System for control

The paper is organized as follows. In section 2 we present an introduction to type-2 FISs and the PSO method, in section 3 we present the description of the problem, in this case regulation of speed of the DC motor in VHDL for FPGA. The AT2-FIS in VHDL code is present in section 4. The design of the PSO method for the AT2-FLC for ReSDCM in shown in section 5, the T2-MF optimization results for AT2-FLC are shown in section 6, finally section 7 offers conclusions about this work.

2 Type-2 Fuzzy Inference Systems and Optimization Methods

The interval type-2 fuzzy inference system (IT2-FIS) consist of four stages: Fuzzification, Inference, Type Reduction and Defuzzification.

The fuzzification stage maps a numeric value $\times=(\times_1..\times_p)^T \in X_1 \times X_2 \times..X_p \equiv X$, into a type-2 fuzzy set $\tilde{A}_x$ in X, where $\tilde{A}_x$ is a singleton fuzzy set, if $\mu_{\tilde{A}_x}(\times)=1/1$ for $\times=\times'$ and $\mu_{\tilde{A}_x}(\times)=1/0$ for all others $\times\neq\times'$ [2].

The inference stage consists of two blocks, the rules and the inference engine, it works the same way as for type-1 fuzzy systems, except the antecedent fuzzy sets and the consequent are represented by type-2 fuzzy sets. The process consists of combining the rules and maps the input to the output (interval type-2 fuzzy sets), using the Join and Meet operations [5]. For an IT2-FIS with p inputs $x_1 \in X_1, x_2 \in X_2, ..., x_p \in X_p$ and one output $y \in Y$, it is assumed that there are M rules, the lth rule in an IT2-FIS and can be written as:

$$R^1: \text{If } x_1 \text{ is and } \tilde{F}_1^l \text{ and} \cdots \text{and } x_p \text{ is } \tilde{F}_p^l \text{ , Then y is } \tilde{G}^l \quad (1)$$

where l=1,…,M. Once we have the rules is necessary to calculate the operations Join(ц) and Meet(п) as well as sup- star composition(★), where l=1,…,M. Once we have the rules is necessary to calculate the operations Join(ц) and Meet(п) as well as sup- star composition(★)[6].

The type reductor stage is used to convert all type- 2 fuzzy sets to type-1 fuzzy intervals on the output. There are several methods to calculate the reduced set, such as the joint center, center of sums, height, among others.

The Defuzzification stage consists in obtaining a numeric value for the output. Using the COS type reductor, the defuzzification is an average value since the range is given by $[y_l, y_r]$ [6].

In the method of average approximation of an interval type-2 fuzzy system (AT2-FIS), the AT2-FIS is replaces by a type-2 fuzzy system using the average of two T1-FIS, this method [8] is performed as follows: 1. Replace each T2-MF for two T1-MFs with different degrees of membership in order to obtain the footprint of uncertainty. 2. To replace the type-2 inference stage, the inference from each T1-FIS must be obtained. 3. To replace the type-reduction system and defuzzification stage of the IT2-FIS, we obtain the defuzzification of each T1-FIS and the results of the two systems are averaged. An IT2-FIS and AT2-FIS can be implemented on a general purpose computer, or by a specific use of a microelectronics realization such as the FPGA. In this work we use the AT2-FIS for FPGA synthesis. Figure 2 shows the block diagram of the AT2-FIS.

The easiest way to get a design of a type-2 fuzzy system is to use software, the problem arises when you have a particular application and the response is not the best, this is when there is the need to optimize the original design. There are many optimization methods [1][3], such as particle swarm optimization.

The Particle Swarm Optimization (PSO) is a bio-inspired optimization method. PSO finds the optimal solution by simulating social behavior. PSO is developed through simulation of birds that come in two-dimensional space, each particle has position and speed.

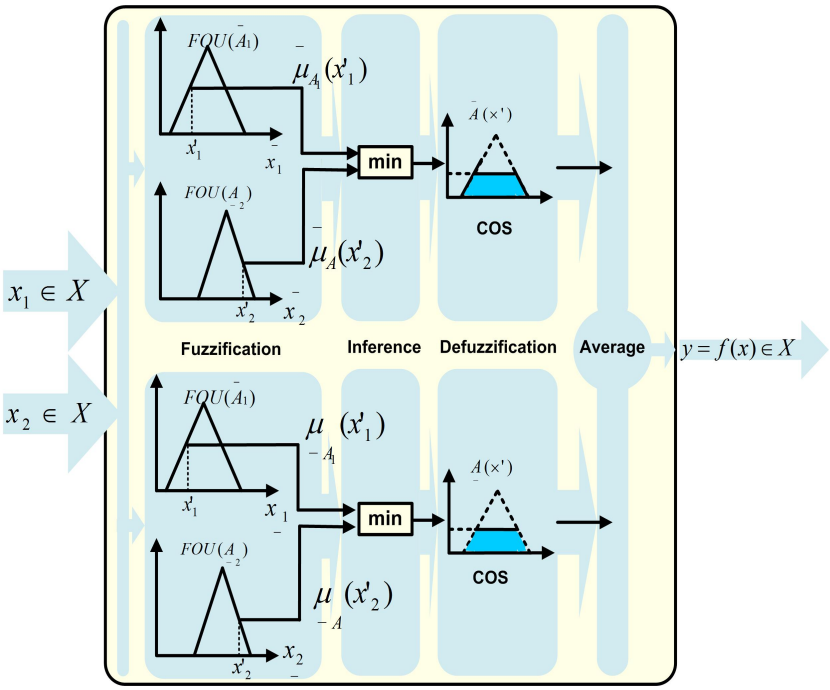


Fig. 2. AT2-FIS

A PSO algorithm maintains a swarm of particles, where each particle represents a possible solution. In analogy with the paradigms of evolutionary computation, the particles are transported through a multidimensional search space, where the position of each particle is adjusted according to their experience and of their neighbors, $x_i(t)$ represents the position of particle i in the search space at time t, t denotes the discrete time. The position of the particle is modified by the addition of a velocity $v_i(t)$, i.e. the current position [1], Equation 2 shows the position of the particle.

$$x_i(t+1) = x_i(t) + v_i(t+1) \tag{2}$$

where $x_i(0) \sim U(x_{min}, x_{max})$. The velocity vector reflects both the experimental knowledge of the particle and the exchanged social information. The experimental knowledge of a particle is often referred to as the cognitive component, which is proportional to the distance of the particle from its best position (referred to as the best personal position of the particle) found from the beginning.

PSO can be described as follows, each swarm knows the best position of the particle (*Plbest*) and the best global position of the swarm (*Pgbest*). The speed of each particle can be calculated using the Equation 3 [1].

$$v_{ij}(t+1) = v_{ij}(t) + c_1 r_{1_j}(t)\left[y_{ij}(t) - x_{ij}(t)\right] + c_2 r_{2_j}(t)\left[\hat{y}_j(t) - x_{ij}(t)\right] \tag{3}$$

where v_{ij} *(t)* is the velocity of the particle *i* from *j = 1, ..., n_x* at time *t*, x_{ij} *(t)* is the position of particle *i* in dimension *j* at time *t*, c_1 and c_2 are the positives constants acceleration used for cognitive and social components respectively, $r_{1j}(t)$, $r_{2j}(t) \sim U(0,1)$, which are random values in the range [0,1]. These random values in the algorithm introduce stochastic elements. y_{ij} is the *Plbest*, is associated with the particle *i*, is the best position of the particle, is the best global position of the particle swarm *Pgbest.*

3 Description of the Problem

This paper proposes the T2-MF optimization with the PSO method for the AT2-FIS codified to VHDL for FPGA. To validate the optimized AT2-FIS we applied the proposed approach to a real problem, such as the regulation of speed of a DC motor (ReSDCM). Figure 3 shows the AT2-FLC for ReSDCM.

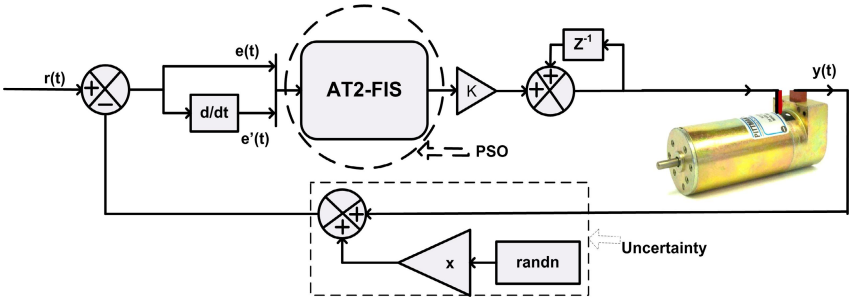


Fig. 3. AT2-FLC for ReSDCM

Figure 3 has the following inputs, error ($e_{(t)}$) and change of error ($e'_{(t)}$), and the output is the control signal ($y_{(t)}$), the control objective AT2-FLC is:

$$\lim_{t \to \infty} \left| y_{(t)} - r_{(t)} \right| = 0 \tag{4}$$

where t is the sampling time.

The inputs are calculated as follows:

$$e_{(t)} = r_{(t)} - y_{(t)} \tag{5}$$

$$e'_{(t)} = e_{(t)} - e_{(t-1)} \tag{6}$$

The reference signal *r(t),* is given by [0,70] revolutions per minute (rpm).

The uncertainty is represented by:

$$\hat{y}_{(t)} = y_{(t)} + x * randn \tag{7}$$

where x is the uncertainty level factor [0,1].

4 AT2-FIS Design in VHDL Code

For the design for AT2-FIS in VHDL we used the average approximation for interval type-2 fuzzy systems. Below we explain the stages: Fuzzification, Inference and Defuzzification.

4.1 Fuzzification

We present an algorithm that works with the calculation of the slopes of the triangular and trapezoidal MFs. The main advantages of this algorithm is that it works for symmetrical and not symmetrical T2-MFs, the value of the slope is calculated on line, therefore is possible to optimize the T2-MF using this method because most of the time in the optimization of membership functions not symmetrical T2-MFs are obtained. A disadvantage of our algorithm is that it only considers membership functions of triangular and trapezoidal form.

The procedure of the algorithm is summarized in three steps: calculate the slope values, calculate the degree values of the membership functions and send to the inference stage the membership degrees and the linguistic terms [7].

Fuzzification stage has inputs such as error($e=x_1$) and change of error($de=x_2$), each with three membership functions (Negative Big(NB="01"), Zero (Z="10"), Positive Big (PB="11")), two trapezoidal MFs and one triangular MF, the universe of discourse as the degree of membership are designed for 8 bits however, we can simply change one variable in the VHDL code to increase or decrease the number of bits. The Fuzzification stage has outputs as degree and linguistic terms for the error input (g_e1,g_e2,g_e3,e1,e2,e3) and the change of error input (g_de1,g_de2,g_de3,de1,de2,de3).

4.2 Inference

The inference stage receives the data sent from the fuzzification stage, which are labels and the membership degrees of each input are: g_e1, g_e2, g_e3, e1, e2, e3, g_de1, g_de2, g_de3, de1, de2, de3, so that multiplexes labels and evaluates the rule base and is illustrated in Table 1.

Table 1. Rule Matrix

e \ de	NB	Z	PB
NB	01	01	01
Z	10	10	10
PB	11	11	11

BD=01 H=10 BI=11

An example of rules using this codification is: If *e* is "PB" and *de* is "Z" then *C* (consequent) is BI. For each of these rules the max-min operation is calculated [9] of labels (c1, c2, c3, c4, c5, c6, c7, c8, c9) and the firing forces (gc1, gc2, gc3, gc4, gc5, gc6, gc7, gc8, gc9) are sent to the defuzzification stage.

4.3 Defuzzification

The Defuzzification stage is calculated using the Height´s method as shown in Equation 8 [10].

$$y_{(x)} = \frac{\sum_{m=1}^{n} C_m o_m}{\sum_{m=1}^{n} o_m} \tag{8}$$

where *C* is the consequent (firing forces) and *o* is the consequent tags (labels). Once the consequent is calculated using Equation 8 the defuzzification stage sends to the output the crisp value.

The three stages are targeted on a FPGA Xilinx Spartan 3AN XC3S700A device. Table 2 shows the device utilization summary for these stage, and we can see that after having synthesized in VHDL the AT2-FIS there is space is available on the FPGA.

Table 2. Device utilization summary for the Fuzzification (F), Inference (I) and Defuzzification (D) stages

Logic Utilization	Used			Available			Utilization (%)		
	F	I	D	F	I	D	F	I	D
No. of 4 Input LUTs	5234	181	104	15360	15360	15360	34	1	0
No. of Bonded IOBs	155	153	2057	173	173	15360	99	88	13

5 Optimization of the Average Approximation of an Interval Type-2 Fuzzy Inference System

We optimized the type-2 membership functions (T2-MFs) of the AT2-FIS with PSO. Figure 4 shows the triangular and trapezoidal T2-MFs that are used.

The design of the AT2-FIS only considers triangular and trapezoidal T2-MFs for each input and output, and Figure 5 shows the inputs and output design.

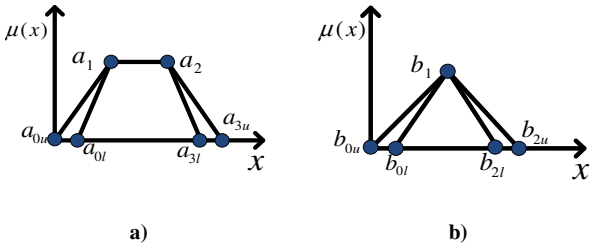


Fig. 4. Triangular and trapezoidal T2-MFs parameters

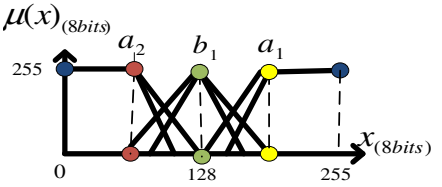


Fig. 5. Inputs and output of the AT2-FIS design

Figure 5 shows the design of the inputs and output T2-MFs of the AT2-FIS, where the universe of discourse and the degrees of membership are divided into 8 bits (or any number of bits). The parameters of these T2-MFs are moved according to the ranges given in Table 3.

Table 3. T2-MF boundary parameters for each input and output

Trapezoidal 1 T2-MF	Triangular T2-MF	Trapezoidal 2 T2-MF
$a_{0U} = a_{0L} = a_1 = 0$	$0 < b_{0U} < 128$	$128 < a_{0U} < 255$
$0 < a_2 < 128$	$b_{0U} < b_{0L} \leq 128$	$a_{0U} < a_{0L} < 255$
$a_2 < a_{3U} \leq 128$	$b_1 = 128$	$a_{0L} < a_1 \leq 255$
$a_2 < a_{3L} < a_{3L}$	$128 < b_{2U} < 255$	$a_2 = a_{3L} = a_{3U} = 255$
	$b_{2U} > b_{2L} \leq 255$	

where U corresponds to the ranges of the upper T2-MFs and L correspond to the ranges of the lower T2-MFs. We can see in Figure 5 that the PSO only needs to move 10 points (see Table 3) for each input and output instead of 17 points for each input and output of Figure 5. In reaching this conclusion, we conducted several experiments to test that with only the optimization of 10 points it was sufficient for a better response in the AT2-FIS in a lesser runtime.

After designing the AT2-FLC (see section 3) and taking into account their characteristics we arrived to the conclusion that the PSO method should be of the multiobjective type [3], because is based on evaluating three characteristics, minimum steady state error, minimum overshoot and minimum undershoot, and these three characteristics help us to determine the best AT2-FLC solution:

a) Minimum overshoot

$$if\ y_{(t)} > r_{(t)} \quad \rightarrow o_1 = \min(y_{(t)}) - r_{(t)} \tag{9}$$

b) Minimum undershoot

$$o_2 = \left|\min(y_{(t)}) - r_{(t)}\right| \tag{10}$$

c) Minimum output steady state error (sse)

$$sse = \sum_{t=201}^{1000} y_{(t)} - r_{(t)} \tag{11}$$

where $y_{(t)}$ is the output of the system and $r_{(t)}$ is reference. The three objective functions are evaluated for fitness evaluation.

For the optimization of the T2-MFs using PSO, we need to define the number of particles in the swarm, the calculation of the position and the initial and final velocity, given by Equations 2 and 3.

The PSO process starts by generating the initial swarm with 10 particles, and these particles are evaluated once for initial selection of the best global particle (*Pgbest*) and best local particle (*Plbest*). If a better particle is found, the T2-MF parameters are sent to the AT2-FLC into the FPGA, if a better particle is not found then each one is again updated by equations for the position and velocity of the particle. Then the T2-MF parameters are downloaded to AT2-FLC in the FPGA, if it meets the convergence criteria (iterations number) then the cycle ends, if not fulfilled the optimization cycle is evaluated the particle swarm by selecting the *Pgbest* and *Plbest* well continuing to end the cycle of optimization. Figure 6 shows the PSO process for the AT2-FLC.

The initial particles are created randomly respecting the ranges of the T2-MFs in Table 3. The particle swarm optimization fits the parameters of the T2-MFs in order to find the best AT2-FLC for ReSDCM using Equation 8, Equation 9 and Equation 10, which are evaluated in the simulation/implementation FPGA block of Figure 6, and for all experiments we used 10 particles.

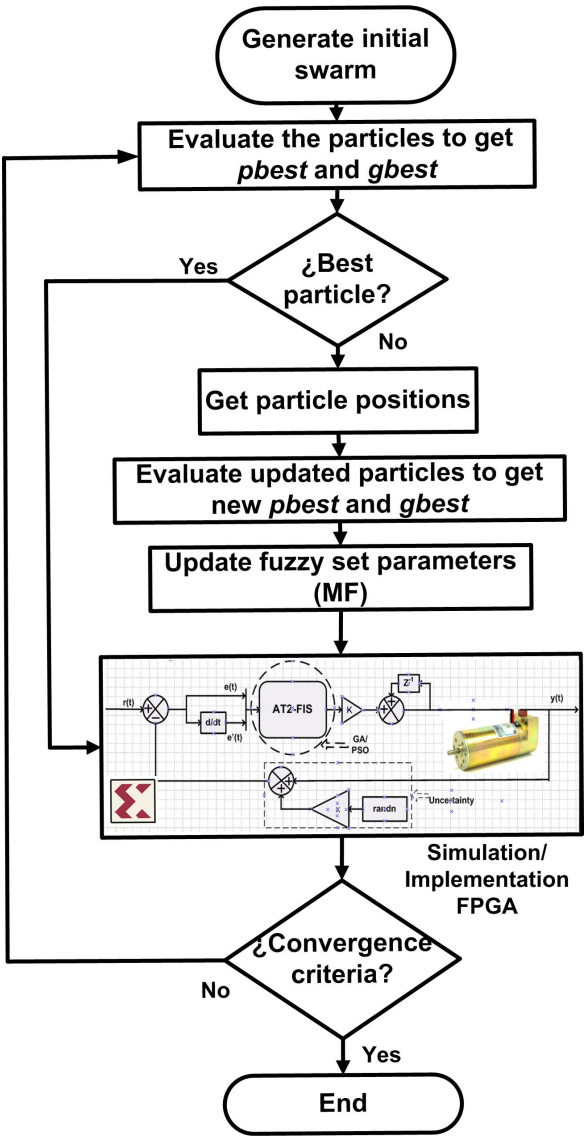


Fig. 6. PSO process for AT2-FLC

6 Results

To demonstrate the performance of the T2-MF optimization for the AT2-FLC for ReSDCM in FPGA implementation, we perform several experiments.

Table 4 shows some results obtained for the T2-MFs with the PSO method, each experiment is an AT2-FLC with different T2-MF parameters. Based on previous experience we change the C_1 and C_2 parameters, which are used to calculate the velocity of each particle of the swarm, the number of iterations varies between 30 and 40, the calculated error is the average error, the time shown is the runtime of the optimization.

Table 4. Results for the T2-MF with PSO for AT2-FLC using FPGA

No.	Iterations	C_1	C_2	Average Error	Time(min)
1	30	0.19	0.19	0.6744	94.6352
2	30	0.15	0.15	1.1352	171.9780
3	30	0.2	0.2	0.5955	92.3928
4	30	0.2	0.2	1.8481	100.2600
5	30	0.21	0.21	0.8050	92.3712
6	30	0.25	0.2	1.1345	92.5114
7	30	0.25	0.25	1.5162	100.1839
8	30	0.2	0.19	0.6786	92.2988
9	30	0.2	0.19	1.1669	92.5059
10	30	0.1	0.1	1.0780	72.7945
11	30	0.1	0.1	1.0224	173.1366
12	30	0.2	0.3	1.5601	98.2220
13	30	0.1	0.09	1.3637	104.2218
14	30	0.1	0.09	0.9950	161.7849
15	30	0.1	0.09	1.1246	88.7282
16	30	0.1	0.08	0.9102	104.0216
17	30	0.09	0.09	0.6164	115.9914
18	30	0.09	0.09	1.3634	172.9863
19	30	0.09	0.08	1.2134	132.1412

For this set of experiments the best AT2-FLC was obtained for experiment No. 3, because it has a lower average error. Figure 7 shows the speed of the DC motor (30 rpm) for the AT2-FLC of experiment No. 3.

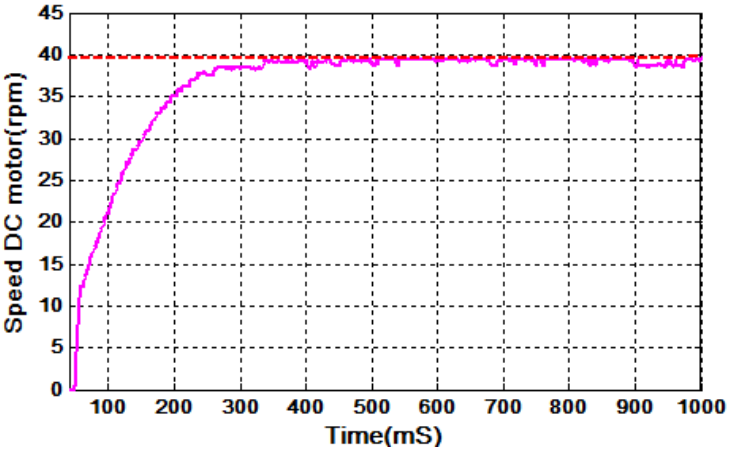


Fig. 7. Speed DC motor of the AT2-FLC

Figure 8 shows the T2-MFs of the inputs and output for experiment No. 3.

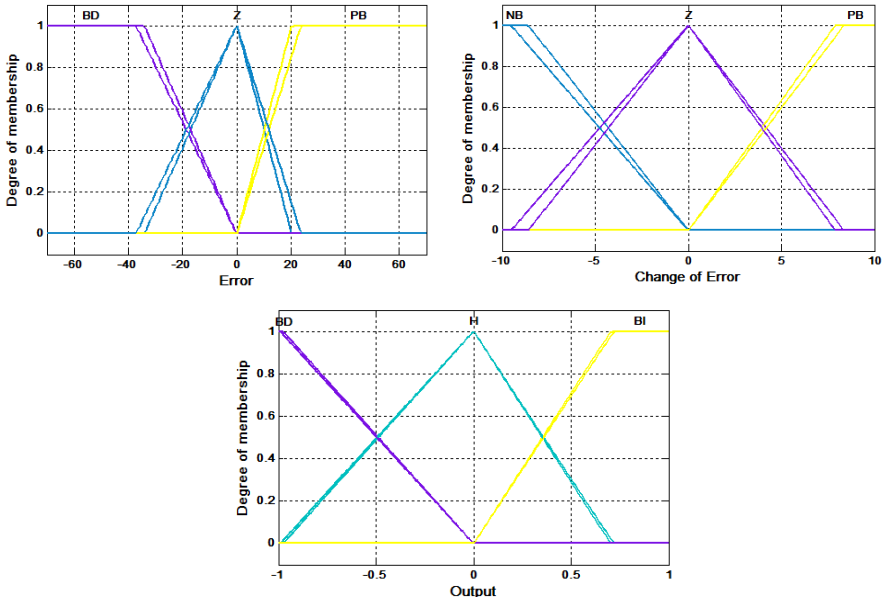


Fig. 8. T2-MFs for inputs and output of the AT2-FIS for ReSDCM

We present a comparison of the best AT2- FLC optimized with PSO (Experiment No.3). The main objective is to apply uncertainty (Equation 7) to AT2-FLC for ReSDCM, in this case we are making comparisons to a desired speed of 40 rpm.

Table 5 shows the comparison of the AT2-FLC (Experiment No.3) adding different levels of uncertainty.

Table 5. Comparison of the best AT2-FLC optimized with PSO

No.	Uncertainty level (x)	Average Error
1	0	0.5955
2	0.001	0.6940
3	0.005	0.7905
4	0.008	0.8386
5	0.05	0.7785
6	0.08	0.7972
7	0.1	0.9673
8	0.5	1.1724
9	0.8	1.3158
10	1	1.6142

The idea of applying uncertainty to the AT2-FLC is to check its robustness of this controller. Figure 9 shows the speed of DC motor for AT2-FLC optimized with PSO for different levels of uncertainty.

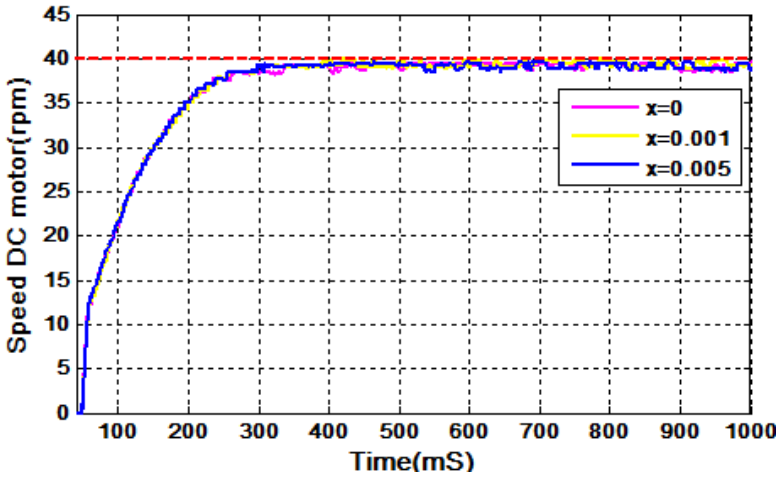


Fig. 9. AT2-FLC optimized with PSO at 40 rpm for different levels of uncertainty

Figure 10 shows a graphical comparison of the average errors of the AT2-FLC optimization using the PSO in 19 different experiments.

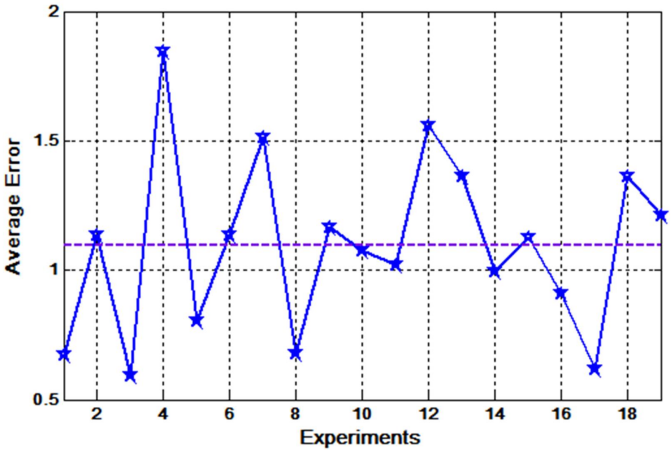


Fig. 10. Average errors for PSO method for AT2-FLC

In Figure 10, we calculated the mean of average error for all experiments, in this case the PSO method has an average error of 1.0948.

Figure 11 shows a graphical comparison of the runtime of the AT2-FLC optimization using the PSO in 19 different experiments.

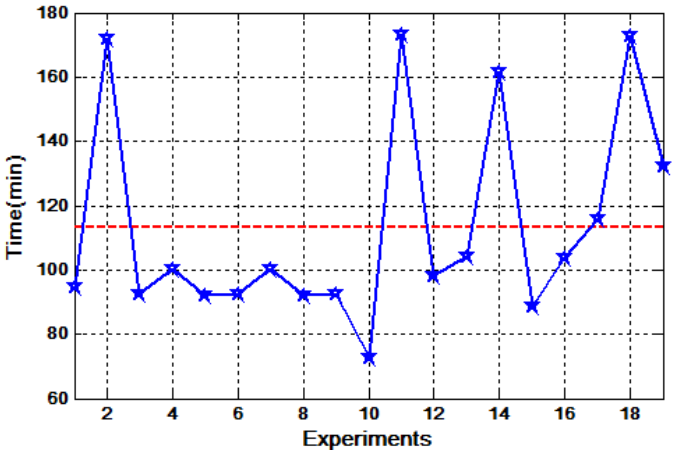


Fig. 11. Runtime for PSO method for AT2-FLC

Of the runtime of the PSO obtained in Figure 10, we calculated the mean of the runtime, in this case the PSO method has a runtime of 113.3244 min.

Experiments were conducted with different resolutions of the VHDL codification for AT2-FLC with PSO optimized, in this case we use 8, 10, 14, 16, 20 and 24 bits. The Table 6 shows the results obtained of the AT2-FLC with PSO optimization for ReSDCM at 30 rpm for different number of bits.

Table 6. Comparison of the best AT2-FLC optimized with PSO for different number of bits

Resolution (Bits)	Average Error
8	0.5955
10	0.3976
14	0.1917
16	0.1917
20	0.1917
24	0.1917

In Table 6 we can see that after 14 bits, the average error does not change, this is because the AT2-FLC performed operations with floating point and it is likely after a certain number of bits some data can not be considered.

Figure 12 shows the comparison of the speed of the DC motor for different numbers of bits.

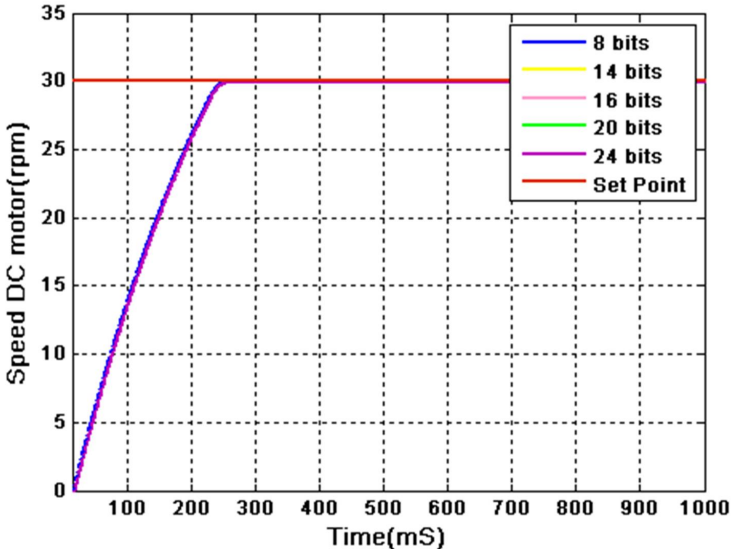


Fig. 12. Comparison of the speed DC motor for different number of bits

Figure 13a and Figure 13b show a zoom to observe the difference between speeds DC motor with different resolution for AT2-FLC at 30 rpm. In Figure 13b we notice the speed DC motor for 8 bits and 24 bits, other speeds are not appreciated because they have that the same average error as the speed of the DC motor for 24 bits.

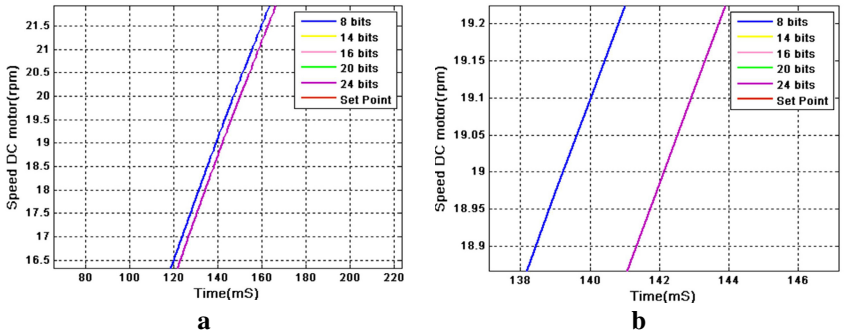


Fig. 13. Comparison of the speed DC motor for different number of bits

In Figure 13b we have a close-up to get a better vision of the behavior of the speeds, we see that the speed of the DC motor for 8 bits has a lower time delay compared to the others, however the others speeds of the DC motor have a lower average.

7 Conclusions

In this paper, an average approximation of an interval type-2 fuzzy system was proposed for hardware implementation. In this particular case, to be implemented into a FPGA.

For all the experiments, it was considered that the common goal of controlling the speed of the DC motor in a FPGA.

Our goal is to achieve an optimized AT2-FLC with a small runtime, for this reason the fuzzy rules are not changed and we propose an optimization for T2-MFs, where only some of these parameters are modified.

An AT2-FLC implementation based on a Xilinx Spartan 3AN FPGA was proposed. We have shown the device utilization for FPGA, these results are encouraging because they allows us to introduce of more T2-MFs and fuzzy rules, in other words, a more complex AT2-FLC to obtain a better result, but this would increase the runtime.

We used different numbers of bits for VHDL codification for AT2-FLC, reaching the conclusion for to our particular application, which is the regulation speed of DC motor, the use of 14 bits is the best option because the error decreases 32% compared with the 8 bits initially used in the implementation.

References

[1] Clerk, M.: Particle Swarm Optimization. ISTE Ltd. (2006)
[2] Klir, G.J., Mark, J.W.: Uncertainty Based Information: Elements of Generalized Information Theory. Physica-Verlag (1999)
[3] Man, K.F., Tang, K.S., Kwong, S.: Genetic algorithms. Springer (2000)

[4] Maldonado, Y., Castillo, O., Melin, P.: Design of a Type-2 Fuzzy Controller and its Comparison with Type-1 Fuzzy and PID Controllers for Velocity Regulation in a DC Motor. In: Proceedings of the World Conference on Soft Computing (WConSC 2011), pp. 1–6 (2011)
[5] Mendel, J.M.: Uncertainty Rule Based Fuzzy Logic Systems: Introduction and New Directions. Prentice-Hall, Upper-Saddle River (2001)
[6] Mendel, J.M.: Type-2 Fuzzy Sets and Systems: an Overview. IEEE Computational Intelligence Magazine 2, 20–29 (2007)
[7] Montiel, O., Sepúlveda, R., Maldonado, Y., Castillo, O.: Design and Simulation of the Type-2 Fuzzification Stage: Using Active Membership Functions. In: Castillo, O., Pedrycz, W., Kacprzyk, J. (eds.) Evolutionary Design of Intelligent Systems in Modeling, Simulation and Control. SCI, vol. 257, pp. 273–293. Springer, Heidelberg (2009)
[8] Sepulveda, R., Castillo, O., Melin, P., Montiel, O.: An Efficient Computational Method to Implement Type-2 Fuzzy Logic in Control Applications Analysis and Design of Intelligent Systems Using Soft Computing Techniques. In: Advances in Soft Computing, vol. 41, pp. 45–52. Springer (2007)
[9] Sepulveda, R., Montiel, O., Olivas, J.A., Castillo, O.: Methodology to Test and Validate a VHDL Inference Engine of a Type-2 FIS, Through the Xilinx System Generator. In: Evolutionary Design of Intelligent Systems in Modeling, Simulation and Control, pp. 295–308. Springer (2009)
[10] Zadeh, L.A.: Fuzzy Sets. Information and Control 8, 338–353 (1965)

Genetic Optimization of Membership Functions in Modular Fuzzy Controllers for Complex Problems

Leticia Cervantes and Oscar Castillo

Tijuana Institute of Technology
ocastillo@tectijuana.mx

Abstract. In this paper a method to design modular fuzzy controllers using genetic optimization is presented. The method is tested with a problem that requires 5 individual controllers. Simulation results with a genetic algorithm for optimizing membership functions of the 5 controllers are presented. Simulation results show that the proposed modular control approach offers advantages over existing control methods.

1 Introduction

This paper focuses on the field of fuzzy logic and control area, these areas can work together to solve various control problems. The problem of water level control for a 3 tank system is illustrated. This control is carried out by controlling 5 valves whose outputs are the inputs to the 3 tanks. The main idea in this paper is to apply a genetic algorithm to optimize the membership functions of the 5 controllers. Each controller has to open and close one of the valves. To control each of the valves we have 5 type-1 fuzzy systems and each fuzzy system has to control one valve of the 3 tanks. After that, the simulation is carry out using type-1 fuzzy systems, then genetic algorithms are used to optimize the 5 controllers. Finally results are presented and compared.

The rest of the paper is organized as follows: In section 2 some basic concepts to understand the work are presented, Section 3 shows a case study, problem description and results are presented and finally in Section 4 conclusion is shown.

2 Background and Basic Concepts

In this section some basic concepts needed for this work are presented.

2.1 Genetic Algorithm

Genetic algorithms (GAs) were proposed by John Holland in the 1960s and were developed by Holland and his students and colleagues at the University of

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 51–62.
springerlink.com

Michigan in the 1960s and the 1970s [2][3]. In contrast with evolution strategies and evolutionary programming, Holland's original goal was not to design algorithms to solve specific problems, but rather to formally study the phenomenon of adaptation as it occurs in nature and to develop ways in which the mechanisms of natural adaptation might be imported into computer systems [13][17]. Holland's 1975 book Adaptation in Natural and Artificial Systems presented the genetic algorithm as an abstraction of biological evolution and gave a theoretical framework for adaptation under the GA [4][5]. A GA allows a population composed of many individuals to evolve under specified selection rules to a state that maximizes the “fitness” [15]. Holland's GA is a method for moving from one population of "chromosomes" (e.g., strings of ones and zeros, or "bits") to a new population by using a kind of "natural selection" together with the genetics inspired operators of crossover, mutation, and inversion [16]. Each chromosome consists of "genes" (e.g., bits), each gene being an instance of a particular "allele" (e.g., 0 or 1) [12][10]. The selection operator chooses those chromosomes in the population that will be allowed to reproduce, and on average the fitter chromosomes produce more offspring than the less fit ones [26]. Crossover exchanges subparts of two chromosomes, roughly mimicking biological recombination between two single chromosome ("haploid") organisms; mutation randomly changes the allele values of some locations in the chromosome; and inversion reverses the order of a contiguous section of the chromosome, thus rearranging the order in which genes are arrayed. (Here, as in most of the GA literature, "crossover" and "recombination" will mean the same thing.) [7][14]. Some of the advantages of a GA include: Optimizes with continuous or discrete variables, doesn’t require derivative information, simultaneously searches from a wide sampling of the cost surface, deals with a large number of variables [11][27].

A typical algorithm might consist of the following:

1. Start with a randomly generated population of n l–bit chromosomes (candidate solutions to a problem).
2. Calculate the fitness $f(x)$ of each chromosome x in the population.
3. Repeat the following steps until n offspring have been created:
   - Select a pair of parent chromosomes from the current population, the probability of selection being an increasing function of fitness. Selection is done "with replacement," meaning that the same chromosome can be selected more than once to become a parent.
   - With probability P_c (the "crossover probability" or "crossover rate"), cross over the pair at a randomly chosen point (chosen with uniform probability) to form two offspring. If no crossover takes place, form two offspring that are exact copies of their respective parents. (Note that here the crossover rate is defined to be the probability that two parents will cross over in a single point. There are also "multipoint crossover" versions of the GA in which the crossover rate for a pair of parents is the number of points at which a crossover takes place.)
   - Mutate the two offspring at each locus with probability P_m (the mutation probability or mutation rate), and place the resulting chromosomes in

the new population. If n is odd, one new population member can be discarded at random.

- Replace the current population with the new population.

Go to step 2 [28][29].

2.2 Fuzzy System

The idea of fuzzy systems appeared very early in the literature of fuzzy sets; it was originated by Zadeh (1965). The concept of a fuzzy system is intimately related to that of a fuzzy set. En order to make our discussion self-contained, it will be helpful to begin with a brief summary of some of the basic definitions pertaining to such sets. Research on fuzzy systems seems to have developed in two main directions. The first is rather formal and considers fuzzy systems as a generalization of nondeterministic systems. These have been studied within the same conceptual framework as classical systems. This approach has given birth to a body of abstract results in such fields as minimal realization theory and formal automata theory, sometimes expressed in the setting of category theory. The system is considered over a given period during which inputs, outputs, and relations may change [26][11].

A system will be called fuzzy as soon as inputs or outputs are modeled as fuzzy sets or their interactions are represented by fuzzy relations. Usually, a system is also described in terms of state variables. In a fuzzy system a state can be a fuzzy set. However, the notion of a fuzzy state is quite ambiguous and needs to be clarified. Note that generally a fuzzy system is an approximate representation of a complex process that is not itself necessarily fuzzy [18][19]. According to Zadeh, the human ability to perceive complex phenomena stems from the use of names of fuzzy sets to summarize information [20]. The notion of probabilistic system corresponds to a different point of view: all the available information at any time is modeled by probability distributions, built from repeated experiments. A fuzzy system can be described either as a set of fuzzy logical rules or as a set of fuzzy equations [21][22]. Fuzzy logical rules must be understood as propositions associated with possibility distributions in the sense of l.E. For instance, “if last input is small, then if last output is large, then current output is medium”, where “small” is a fuzzy set on the universe of inputs, and “medium” and “large” are fuzzy sets on the universe of outputs [23][24] . Let u_t , y_t, and s_t denote respectively the input, output, and state of a system S at time t. U, Y, S are respectively the set of possible inputs, outputs, and states [25][30]. Such a system is said to be deterministic if it is characterized by state equations of the form:

$$s_{t+1} = \delta(u_t, s_t), \qquad y_t = \sigma(s_t), \qquad t \in \mathbb{N}. \tag{1}$$

s_0 is called the initial state; δ and Ω are functions from U X S and from S to S and Y, respectively. S is said to be nondeterministic if S_{t+1} and / or Y_t, are not uniquely determined by U_t and S_t [31][1]. Let S_{t+1} and Y_t be the sets of possible values of S_{t+1} and Y_t, respectively, given U_t, and S_t. S_{t+1} and Y_t, may be understood

as binary possibility distributions over *S* and *Y*, respectively. In some cases a fuzzy system is used to control complex problem to obtain better results [8][9][6].

3 Case Study

In this Section the problem description is presented and results are shown.

3.1 Problem Description

In this work the case study considers the problem of water level control for a 3 tanks system where the 3 tanks include valves that are opened or closed, these valves must be well controlled to give the desired level of water in each of the three tanks. The end tanks have a valve that fills and in the middle of the 3 tanks there are two valves that control the water level between tanks 1 and 2, and tanks 2 and 3. The water tank 3 has a valve to output more water flow, the case study model is made in simulink and has three inputs (tank 1, tank2 and tank3), and these entries correspond to the existing water levels in tank 1, tank2 and tank3. The outputs of the model made in simulink has five valves, which provide water (v1 and v2) valves that are interconnected tanks (v13 and v32) and finally the exit valve is responsible for the drainage of the three tanks (v20). The problem is shown in Figure 1.

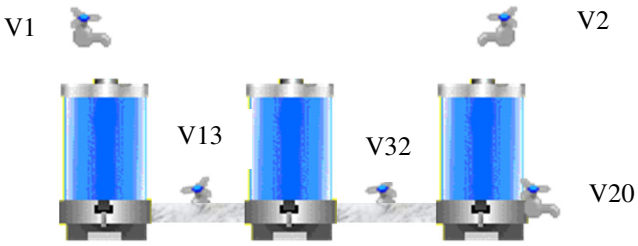


Fig. 1. Water control of 3 tanks

3.2 Type-1 Fuzzy System

For this case study was necessary to use fuzzy systems to realize the simulation, each fuzzy system has one or two inputs depend on the valve. Valves that are between 2 tanks using 2 inputs (tank1 and tank2 or tank2 and tank3). The outputs are the valves, in total 5 fuzzy system was use in this problem. The fuzzy systems are shown in Figures 2 to 6.

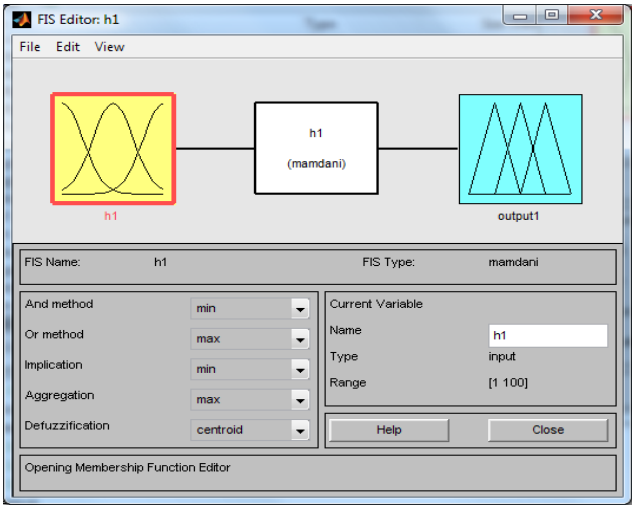


Fig. 2. Fuzzy system to control valve 1

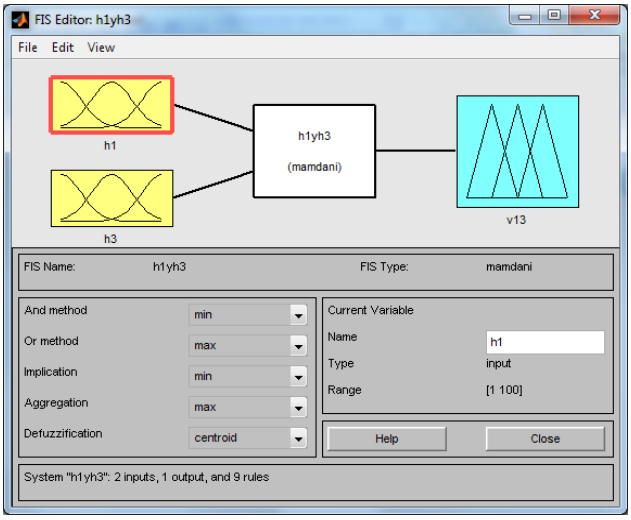


Fig. 3. Fuzzy system to control valve 13

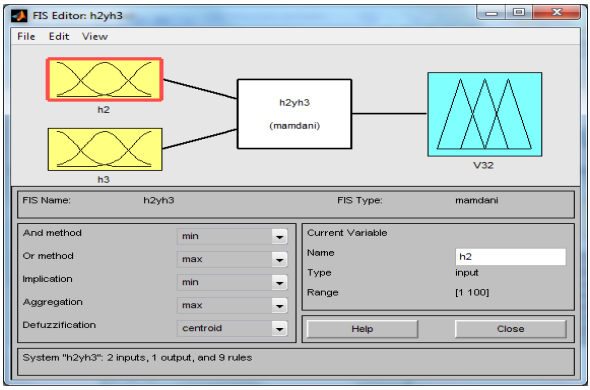


Fig. 4. Fuzzy system to control valve 32

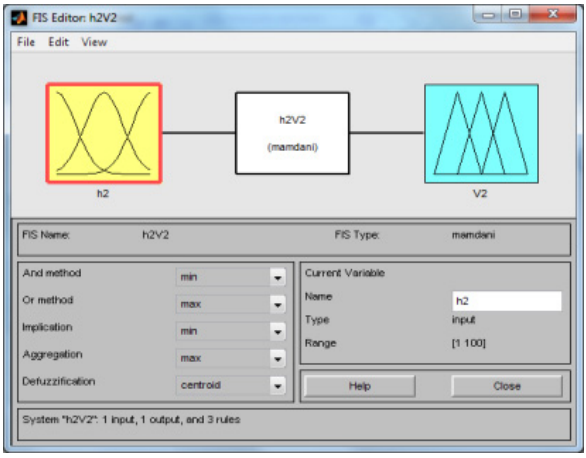


Fig. 5. Fuzzy system to control valve 2

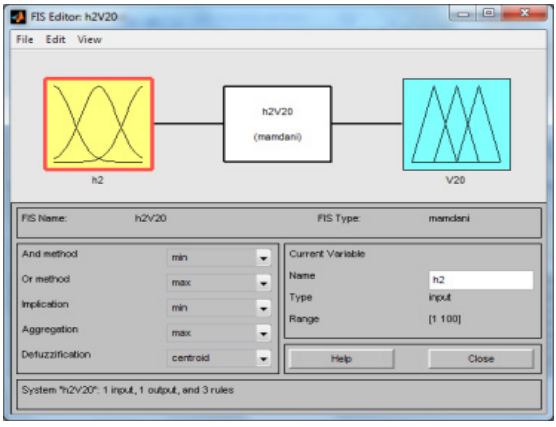


Fig. 6. Fuzzy system to control valve 20

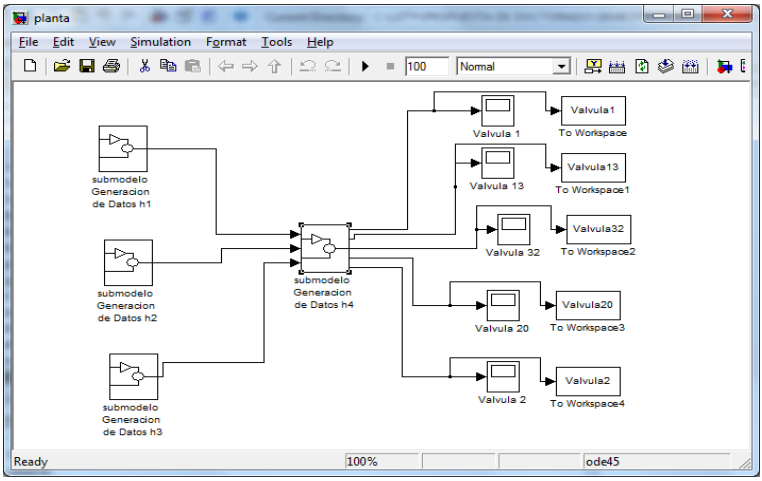


Fig. 7. Simulation plant

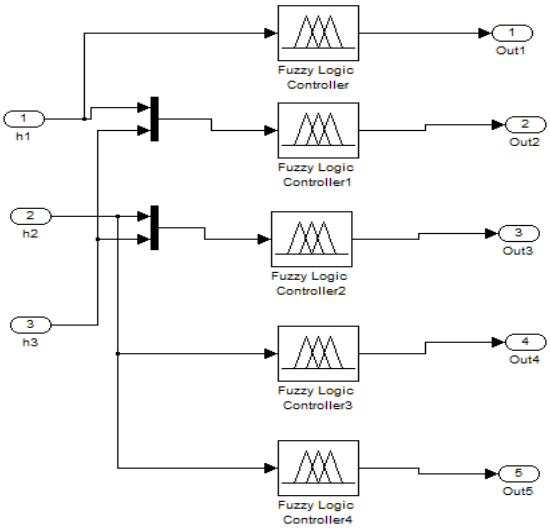


Fig. 8. Simulation plant showing inputs and outputs

Having created the fuzzy systems, the simulation was performed using the Matlab programming language. The simulation plant is shown in Figures 7 to 8.

The simulation was carried out using the fuzzy system shown before, the membership functions used in this case was triangular, Gaussian and trapezoidal. The results with type-1 fuzzy system are presented in Tables 1 to 3.

Table 1. Results for the simulation plant using triangular membership functions

Using Triangular Membership Function	Error
valve 1	0.9246
valve 13	0.9278
valve 2	0.9278
valve 20	0.9279
valve 32	0.8341

Table 2. Results for the simulation plant using Gaussian membership functions

Using Gaussian Membership Function	Error
valve 1	0.898
valve 13	0.8994
valve 2	0.8994
valve 20	0.8995
valve 32	0.8463

Table 3. Results for the simulation plant using trapezoidal membership functions

Using Trapezoidal Membership Function	Error
valve 1	0.9522
valve 13	0.9551
valve 2	0.9504
valve 20	0.9551
valve 32	0.8233

3.3 Genetic Algorithm

After obtaining the previous mentioned results, genetic algorithm optimization was performed. The genetic algorithm is used to optimize the membership functions of each fuzzy system (inputs and outputs). In the genetic algorithm the

membership functions of the 5 controllers were optimized, In the algorithm the error of each controllers is taken and finally the results of each controller were added, and final results is divided between the number of the controllers. The fitness function is shown in next equation:

$$f(y) = \frac{\left(\text{Error C1} = \sum_{i=1}^{n} \frac{\left| y^i_{REF_1} - y^i_{FS_1} \right|}{n} + \text{Error C2} = \sum_{i=1}^{n} \frac{\left| y^i_{REF_2} - y^i_{FS_2} \right|}{n} + \ldots \text{Error CN} = \sum_{i=1}^{n} \frac{\left| y^i_{REF_N} - y^i_{FS_N} \right|}{n} \right)}{N} \quad (2)$$

Where Y_{REF} is the reference, Y_{FS} is the output of the controller and **n** is the number of point used in comparison. Error C1 is the error of control 1 to N, and N in the number of the controllers.
The parameters used in the GA are:

- Generations: 20
- Individuals: 25
- Selection: Roulette
- Crossing point
- Discrete mutation Pm: 0.2

After the use of the genetic algorithm the results obtained in the simulation are shown in Table 4.

Table 4. Results for the simulation plant using triangular membership functions and genetic algorithm

Error using triangular membership functions and genetic algorithm				
Valve 13	Valve 1	Valve 20	Valve 2	Valve 32
0.109	0.1146	0.0939	0.2077	0.218
0.131	0.1228	0.1329	0.1861	0
0.119	0.1275	0.111	0.239	0
0.115	0.1116	0.1092	0.2216	0
0.109	0.0908	0.1191	0.214	0
0.109	0.1132	0.0954	0.1922	0
0.117	0.1225	0.1003	0.1853	0
0.107	0.1102	0.1146	0.1938	0

Table 4. (*continued*)

0.105	**0.0993**	**0.0851**	**0.2428**	**0**
0.125	0.1196	0.113	0.1433	0
0.123	0.1191	0.1394	0.246	0
0.115	0.1114	0.091	0.1539	0
0.117	0.1231	0.101	0.1818	0
0.107	0.1444	0.0661	0.1366	0
0.117	0.1225	0.1003	0.1853	0

The above table shows a lower error in comparison with only using a type-1 fuzzy system.

4 Conclusions

A benchmark problem was used to test the proposed approach and based on the obtained results we can say that to achieve control of the present problem, a genetic algorithm is a good alternative to obtain a good fuzzy controller.

When a complex control problem is at hand, we start working on the case study, once results are obtained with type-1 fuzzy systems is a good choice to use a genetic algorithm for optimizing membership functions of inputs and outputs of the controllers and to obtain better control as was the case in this control problem. In the moment when genetic algorithm was use results were better that type-1 fuzzy system, this is possible because in the moment that genetic algorithm is employed, it moves the parameters of the membership functions and the problem has more options to control de valves and the genetic algorithm is evaluated to obtain the best fuzzy system to control the open and close valves and this is why better results are obtained by optimizing the membership functions.

References

1. Castillo, O.: Design of Interval Type-2 Fuzzy Logic Controllers. Springer (2012)
2. Castillo, O., Martinez-Marroquin, R., Melin, P., Valdez, F., Soria, J.: Comparative study of bio-inspired algorithms applied to the optimization of type-1 and type-2 fuzzy controllers for an autonomous mobile robot. Inf. Sci. 192, 19–38 (2012)
3. Castillo, O., Melin, P.: A review on the design and optimization of interval type-2 fuzzy controllers. Appl. Soft Comput. 12(4), 1267–1278 (2012)
4. Castillo, O., Melin, P.: New fuzzy-fractal-genetic method for automated Mathematical Modelling and Simulation of Robotic Dynamic Systems. In: IEEE International Conference on Fuzzy Systems, vol. 2, pp. 1182–1187 (1998)

5. Castillo, O., Melin, P., Montiel, O., Sepúlveda, R.: Optimization of interval type-2 fuzzy logic controllers using evolutionary algorithms. Soft Comput. 15(6), 1145–1160 (2011)
6. Castillo, O., Kacprzyk, J., Pedrycz, W.: Soft Computing for Intelligent Control and Mobile Robotics. Springer (2011)
7. Cázarez, N., Aguilar, L., Castillo, O.: Fuzzy logic control with genetic membership function parameters optimization for the output regulation of a servomechanism with nonlinear backlash. Expert System Appl. 37(6), 4368–4378 (2010)
8. Cervantes, L., Castillo, O.: Design of a Fuzzy System for the Longitudinal Control of an F-14 Airplane. In: Castillo, O., Kacprzyk, J., Pedrycz, W. (eds.) Soft Computing for Intelligent Control and Mobile Robotics. SCI, vol. 318, pp. 213–224. Springer, Heidelberg (2010)
9. Cervantes, L., Castillo, O., Melin, P.: Intelligent Control of Nonlinear Dynamic Plants Using a Hierarchical Modular Approach and Type-2 Fuzzy Logic. In: Batyrshin, I., Sidorov, G. (eds.) MICAI 2011, Part II. LNCS, vol. 7095, pp. 1–12. Springer, Heidelberg (2011)
10. Coley, A.: An Introduction to Genetic Algorithms for Scientists and Engineers. World Scientific Book (1999)
11. Dubois, D., Prade, H.: Fuzzy sets and Systems: Theory and Applications (1980)
12. Gibbens, P., Boyle, D.: Introductory Flight Mechanics and Performance. University of Sydney, Australia Paper (1999)
13. Haupt, R., Haupt, S.: Practical Genetic Algorithm. Wiley Interscience (2004)
14. Hidalgo, D., Melin, P., Castillo, O.: An optimization method for designing type-2 fuzzy inference systems based on the footprint of uncertainty using genetic algorithms. Expert Syst. Appl. 39(4), 4590–4598 (2012)
15. Martinez-Soto, R., Castillo, O., Aguilar, L.: Optimization of interval type-2 fuzzy logic controllers for a perturbed autonomous wheeled mobile robot using genetic algorithms. Inf. Sci. 179(13), 2158–2174 (2009)
16. Melin, P., Castillo, O.: A new method for adaptive model-based control of non-linear plants using type-2 fuzzy logic and neural networks. In: IEEE International Conference on Fuzzy Systems, vol. 1, pp. 420–425 (2003)
17. Mitchell, M.: An Introduction to Genetic Algorithms. Massachusetts Institute of Technology Book (1999)
18. Rachman, E., Jaam, J., Hasnah, A.: Non-linear simulation of controller for longitudinal control augmentation system of F-16 using numerical approach. Information Sciences Journal 164(1-4), 47–60 (2004)
19. Reiner, J., Balas, G., Garrard, W.: Flight control design using robust dynamic inversion and time-scale separation. Automatic Journal 32(11), 1493–1504 (1996)
20. Sanchez, E., Becerra, H., Velez, C.: Combining fuzzy, PID and regulation control for an autonomous mini-helicopter. Journal of Information Sciences 177(10), 1999–2022 (2007)
21. Sefer, K., Omer, C., Okyay, K.: Adaptive neuro-fuzzy inference system based autonomous flight control of unmanned air vehicles. Expert Systems with Applications Journal 37(2), 1229–1234 (2010)
22. Song, Y., Wang, H.: Design of Flight Control System for a Small Unmanned Tilt Rotor Aircraft. Chinese Journal of Aeronautics 22(3), 250–256 (2009)
23. Walker, D.J.: Multivariable control of the longitudinal and lateral dynamics of a fly by wire helicopter. Control Engineering Practice 11(7), 781–795 (2003)
24. Wu, D.: A Brief Tutorial on Interval Type-2 Fuzzy Sets and Systems (July 22, 2010)

25. Wu, D., Mendel, J.: On the Continuity of Type-1 and Interval Type-2 Fuzzy Logic Systems. IEEE T. Fuzzy Systems 19(1), 179–192 (2011)
26. Zadeh, L.: Fuzzy Sets and Fuzzy Information Granulation Theory. Beijing Normal University Press Baijing (2000)
27. Zadeh, L.: Fuzzy Sets. Information and Control 8(3), 338–353 (1965)
28. Zadeh, L.: Shadows of Fuzzy Sets. Probl. Peredachi Inf. 2(1), 37–44 (1966)
29. Zadeh, L.: Fuzzy Logic. Neural Networks and Soft Computing Commun. ACM 37(3), 77–84 (1994)
30. Zadeh, L.A.: Some reflections on soft computing, granular computing and their roles in the conception, design and utilization of information/intelligent systems. Soft Comput. 2, 23–25 (1998)
31. Zadeh, L.A.: Outline of a new approach to the analysis of complex systems and decision processes. IEEE Trans. Syst. Man Cybern. SMC-3, 28–44 (1973)

Designing Systematic Stable Fuzzy Logic Controllers by Fuzzy Lyapunov Synthesis

María Concepción Ibarra, Oscar Castillo, and Jose Soria

Tijuana Institute of Technology
Tijuana, México
ocastillo@hafsamx.org

Abstract. Fuzzy logic handles information imprecision using intermediate expressions to define assessments. Fuzzy Systems are intelligent models whose main application has been in Control Engineering applications. Stability is one of the most important issues of control systems. This determines the system to respond in an acceptable way. This work is based on the fuzzy Lyapunov synthesis in the design of fuzzy controllers, to verify the system's stability. The stability will be studied on Mamdani and Sugeno fuzzy systems .The case study presented is a system of a cylindrical tank of water, where we aim to maintain a certain level of water, which is regulated through the controls applied to the water outlet valve of the tank. The method is also tested using an inverted pendulum, which is an unstable system, which can fall at any time unless an appropriate force is applied control.

1 Introduction

One of the main areas of application of fuzzy logic has undoubtedly been the automatic process control, mainly due to the special feature of fuzzy systems that operate in the same numerical and linguistic framework.

Fuzzy systems have shown ability to resolve problems on several application domains. At present time, there is a growing interest to improve the fuzzy systems with learning and adaptation capacities.

Fuzzy systems have been successfully applied to classification problems, control and in a considerable amount of applications. On most of the cases, the key to success has been the ability of the fuzzy systems of incorporate human expert's knowledge.

One of the main problems that man has come across to in the study of the theory of dynamic systems control, is the stability problem. Throughout the years several criterions have been developed to evaluate the stability in the fuzzy controllers.

An effective method to create stable fuzzy controllers is the use of fuzzy Lyapunov synthesis method, where we can design the linguistic rules bases of such controllers.

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 63–79.
springerlink.com

With time, some methods were formalized, such as the fuzzy Lyapunov synthesis [1] to measure the stability of the only control systems that were applied until the end of the 90s, which are the Type-1 fuzzy control systems.

In this work, the proposed method of fuzzy Lyapunov synthesis is described, with emphasis on its application to the systematic design of fuzzy controllers, where the objective is to achieve the stability in such controllers using this method. Where is shown that as the fuzzy Lyapunov synthesis it's a valid tool for evaluate the stability in Type-1 fuzzy control systems.

In this work the results for the design of FLC (fuzzy logic controller-FLS) applying fuzzy Lyapunov synthesis are presented, a concept that is based in the computation with words paradigm [6], with the purpose of provide evidence of the systems strength.

Also, this work we present a dynamic model for the study cases, such method will guarantee the stability when applied to the water tank case and the inverted pendulum.

2 Type-1 Fuzzy Sets and Systems

The Fuzzy sets are defined based on the operating characteristics of systems. A fuzzy set in the universe U is characterized by the membership function A (x) which takes the interval [0, 1], unlike classical sets take the value zero or one {0, 1}.

The formal definition of a fuzzy set and membership function is as follows:

"If X is a collection of objects denoted generically by x, then a fuzzy set in X is defined as a set of ordered pairs"

The fuzzy set can be represented by:

$$A = \{(\mu_A(x), x) / x \varepsilon U\} \quad (1)$$

Where μA (x) is the degree of membership. The membership functions for the fuzzy set A.

A membership function with parameters p (x) the element x is a follows:

$$\mu_A(x) = \mu_A(p_1(x), p_2(x), \ldots, p_n(x)) \quad (2)$$

An enumeration of pairs defined on discrete elements of the set is as follows:

$$A = \sum_{x \in U} \mu_A(x)/X \quad (3)$$

Where ∑ is not a sum, but an aggregation of pairs, and $\mu_A(x)/X$ does not represent any ratio, but a couple (possible/ cell).

Each fuzzy system is associated with a set of rules with regard to IF-THEN linguistic interpretations and can be expressed as follows:

$$R^m: \text{If } u_1 \text{ is } A_1^m \text{ and } \ldots u_p \text{ is } A_p^m. \text{ Then v es } B^m \quad (4)$$

With m = 1,2, ... M

And where A^m and B^m are fuzzy sets in U ⊂ R (real numbers) and V ⊂ R respectively, u= $(u_1, u_2, \ldots u_n)$

$\in U_1 x U_2 x \ldots x U_n and v \in V.$ $and x = x_{1,} x_{2,} \ldots, x_n \in Ue$ and $\in V$ are the specific numerical values of u and v, also respectively.

A Mamdani fuzzy system consists of 4 basic elements: the fuzzifier, the rule base, the knowledge base and inference system defuzzifier. While in the Sugeno fuzzy system the rule base operates differently than the Mamdani systems because the consequent of these rules is no longer a linguistic label but is a function of input that has the system at any given time.

3 Overview of the Problem

3.1 Description of Then of the Water Tank

Let us consider the problem of designing a stable fuzzy controller (FLC) for a cylindrical water tank, based on a Takagi-Sugeno controller.

The tank has an inlet and outlet pipe. It can change the valve controlling the water that is entering, but the flow going out depends on the diameter of the outlet pipe (which is constant) and the pressure in the tank (which varies with the level of water). The system has many nonlinear characteristics.

A controller for the water level in the tank needs to know the current water level and needs to be able to calibrate the valve. The input to the controller is the error in the water level (the desired water level minus the current water level) and its output is the velocity at which the valve opens or closes.

3.1.1 Dynamic Model

The dynamic behavior of the system is governed by the following differential equation:

$$\dot{x} = q(t) - p(x)u \tag{5}$$

where:

$\dot{x}$: Change in the amount of water in the tank.
q (t): is the flow of water entering the tank.
u: is the variable that controls both opening the drain valve.
p (x)u: is the flow of water leaving the tank.

The aim is to design the rule base for constructing a fuzzy controller:

$$u = u(x; p_s, q_s) \tag{6}$$

x: is the quantity of water in the tank
p_s: is the nominal output steady flow of water.
q_s: is the nominal steady flow of water intrusion.

Capable of regulating the amount of water in the tank x (t) to a desired nominal amount x_s:

$$x(t) \rightarrow x_s \tag{7}$$

The system has three modes of operation: x is low, x is normal and x is high.

3.1.2 Stability Analysis

To consider fuzzy Lyapunov synthesis, we assume the following:

- We assume that the functional relationship (5) is known.
- But p (x) is not known explicitly and the only knowledge we have of p (x) is: p (x) ≥0 for all x.
- The value p_s= p (x_s) is known.

Theorem 1 (Asymptotic stability [3])
An equilibrium point x = 0 is asymptotically stable at $t = t_0$ if

1. x = 0 is stable, and
2. x= 0 is locally attractive; i.e., there exists $\delta(t0)$ such that

$$\|x(t0)\| < \delta \Rightarrow \lim_{t\to\infty} x(t) = 0. \tag{8}$$

Consider the nonlinear system with an equilibrium point at the origin, f(0)=(0) then the origin is asymptotically stable if there exists a scalar Lyapunov function V(x) with continuous partial derivatives such that

- V(x) is positive definite
- $\dot{V}$ (x) is negative definite

To determine the rules of control in each of the three modes, the fuzzy controller design proceeds as follows.

Let us introduce the Lyapunov function candidate.

$$V = \frac{1}{2}(x - x_s)^2 \tag{9}$$

Differentiating V we have:

$$\dot{V} = (x - x_s)\dot{x} = (x - x_s)(q(t) - p(x)u) \tag{10}$$

Negative equation for each mode of operation and stability to the tank system, so we need to be met:

$$\begin{aligned} \dot{V} &< 0 \\ \dot{V} &\leq 0 \end{aligned} \tag{11}$$

If *x* is low, then $(x - x_s)$< 0, therefore negative for $\dot{V}$ we need (q (t)> p (x) u) is fulfilled.

But we know that q(t) and p (x) are non-negative, so we set, u = 0.

If x is high, then $(x - x_s)$>0, thus making $\dot{V}$ negative needs that $q(t) < p(x)u$ is met and set u = u_{max} where $u_{max}$is the openness of the outlet valve.

For the last case when x is normal, when designing the Takagi-Sugeno fuzzy controller, each rule is given by the following fuzzy rule as follows $u = k_1 x + k_2$ for some constant k1, k2. Substituting $u = k_1 x + k_2$ in the fuzzy control system we obtain:

$$\dot{V} = (x - x_s)(q(t) - p(x)(k_1 x + k_2))$$
$$= (x - x_s)(q(t) - p(x)(k_1(x - x_s)k_1 x_s + k_2))$$
$$= k_1 p(x)(x - x_2)^2 + (x - x_s)\,(q(t) - p(x)(k_1 x_s + k_2)) \quad (12)$$

The first term in the case of $(k_1 p(x)(x - x_2)^2)$ is not positive for any k_1>0. Therefore, to make negative $\dot{V}$ is needed to clear the second term that is $q(t) - p(x)(k_1 x_s + k_2) = 0$, or it could be $k_2 = \frac{q(t)}{p(x)} - k_1 x_s$. Then $q(t) - p(x)$are unknown, we approach using q_s and p_s therefore we obtain $k_2 = \frac{q_s}{p_s} - k_1 x_s$

Then we have that when x is normal, $u = k_1 x + k_2 = k_1 x + \frac{q_s}{p_s} - k_1 x_s$for some constant $k_1 > 0$.

Recall that the outlet valve of the tank water is not always negative, that is, always leaving at least a minimum amount of water, $k_1$therefore be fulfilled: $k_1 = (0 - x_s) + \frac{q_s}{p_s} \geq 0$, or:

$$k_1 \leq \frac{q_s}{p_s x_s} \quad (13)$$

In summary, using Fuzzy Lyapunov synthesis we obtain the following rules of Takagi-Sugeno control for the water tank system. Where now the output linguistic variables are as shown below:

- If x is low then $u = 0$
- If x normal then $u = (x - x_s) + \frac{q_s}{p_s}$
- If x high then u = u

Using Fuzzy Lyapunov synthesis and fuzzy control rules Takagi-Sugeno type obtained for the System Water Tank, help us reach the Mamdani type rules where the problem statement in terms of the input variable would be virtually same, that is, the input variables are given in linguistic variables and x is the amount of water in the tank which as noted may be low, normal or high and now add a further input variable, which will flow_current call and this variable can be negative or positive, this depends on the nominal steady flow denoted by $q_s$input. Now as for the output variables there is a difference with the Takagi-Sugeno type controller where the output variables are mathematical functions such as: If x is normal then $u = k_1(x - x_s) + \frac{q_s}{p_s}$

The Conditions can be converted to fuzzy rules as follows:

- Si x is low then $u = 0$
- Si x is normal then $u = (x - x_s) + \frac{q_s}{p_s}$
- Si x is high then $u = u$

(14)

- IF nivel is low THEN valve is close_fast
- IF nivel is normal THEN valve is no_change
- IF nivel is high THEN valve is open_fast

(15)

3.2 Description of the Inverted Pendulum

The Inverted Pendulum plant consists of a cart and a pendulum. The regulator's objective is to move the cart to its commanding position, without causing the pendulum to tip over.

3.2.1 Dynamic Model

The inverted pendulum control has a huge variety of problems that have made it one of the concrete systems for testing control laws discussed in more recent times. In the inverted pendulum control there are basically two problems: the problem of local stability around the equilibrium position, which is analogous to the problem of the juggler who intends to keep a stick in the tip of a finger, and the problem of lifting the pendulum from its rest position to the position where it is kept straight upwards.

Let us consider the problem of designing a stable fuzzy controller for the well-known inverted pendulum problem. The state variables are as follows [9]:

$x_1 =$ θ- The angle of the pendulum, and
$x_2 = \dot{\theta}$ - Angular velocity

The system dynamic equations, which are assumed unknown, are shown below:

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= f(x_1, x_2) + g(x_1, x_2)u \end{aligned} \tag{16}$$

Where:

$$f(x_1, x_2) = \frac{9.8 \sin x_1 - \frac{mlx_2^2 \cos x_1 \sin x_1}{m_c+m}}{l\left(\frac{4}{3} - \frac{m\cos^2 x_1}{m_c+m}\right)} \tag{17}$$

m_c: is the mass of the carriage
M: is the mass of the rod
l: the length of the rod
U: is the applied force (control).

3.2.2 Stability Analysis

To apply fuzzy Lyapunov synthesis, we assume the following:

- The system has two degrees of freedom, θ and $\dot{\theta}$ for us $x_1$and $x_2$called respectively. Therefore $\dot{x}_1 = x_2$.
- $\dot{x}_2$ is proportional to the control signal u, that is, when u increases (decreases) $\dot{x}_2$increases (decreases).

Theorem 1 (Asymptotic Stability [3]). An equilibrium point x = 0 is asymptotically stable at $t = t_0$ if

1 x = 0 is stable, and
2 x = 0 is locally attractive; i.e., there exists δ(t0) such that

$$\|x(t0)\| < \delta \Rightarrow \lim_{t\to\infty} x(t) = 0 \quad (18)$$

Consider the nonlinear system with an equilibrium point at the origin, f(0)=0 then the origin is asymptotically stable if there exists a Lyapunov function V (x) with continuous derivatives such that:

- V(x) is positive definite
- $\dot{V}$ (x) is negative definite

The fuzzy controller design proceeds as follows.

Let us introduce the Lyapunov function candidate

$$V(x_{1,}x_2) = \frac{1}{2}(x_1^2 + x_2^2) \quad (19)$$

which is positive-definite and radially unbounded function. The time derivative of$V(x_{1,}x_2)$ results in:

$$\dot{V} = x_1\dot{x}_1 + x_2\dot{x}_2 = x_1x_2 + x_2\dot{x}_2 \quad (20)$$

To guarantee stability of the equilibrium point we require that:

$$x_1x_2 + x_2\dot{x}_2 < 0 \quad (21)$$

We can now derive sufficient conditions so that inequality (21) holds: If and have opposite signs, then and (21) will hold if; ifx_1 and $x_2$are both positive, then $x_1x_2 < 0$ (21) will hold if $\dot{x}_2 = 0$; $x_1$if $x_2$and are both negative, then (21) will hold if $\dot{x}_2 < -x_1$; if x_1 and x_2 are both negative, then (21) will hold if $\dot{x}_2 < -x_1$.

Can clearly see that V is positive definite.

With this knowledge about the system, we can now derive sufficient conditions to ensure that (20) is fulfilled, these conditions can be viewed as follows:

- If x_1 and $x_2$have opposite signs, then x_1 x_2 and (21) is satisfied if $\dot{x}_2 = 0$
- If x_1 and x_2 and are both positive, (21) is satisfied if $\dot{x}_2 = -x_1$
- If x_1 and x_2 and are both negative, (21) is satisfied if $\dot{x}_2 =>\ -x_1$

Conditions can be transferred from conditions for them to be converted to fuzzy rules created, are as follows:

- IF $x_1$is positive AND x_2 is positive, THEN $\dot{x}_2$ is negative big
- IF $x_1$is negative AND $x_2$is negative, THEN $\dot{x}_2$ is positive big
- IF $x_1$is positive AND$x_2$is negative, THEN $\dot{x}_2$is zero
- IF $x_1$is negative AND $x_2$is positive, THEN $\dot{x}_2$iszero

It is important to note that the partitions or fuzzy granulation x_1, $x_2$and *u* elegantly derive the expression (19), because $V = x_2(x_1 + \dot{x}_2)$, and as we need $\dot{V}$ is negative, it is natural to examine the signs of $x_1$and x_2, therefore, the obvious fuzzy partition is positive, negative. The partition for x_2 or *u* called large negative, zero, positive big is obtained in a similar way when we give linguistic values positive or negative for $x_1$and $x_1$in (20).

To ensure that $\dot{x}_1 < -x_1$ y $\dot{x}_2 > -x_1$ is satisfied even if we do not know the exact magnitude of x_1,only that it is positive or negative, we must give $\dot{x}_2$ or u large negative values and large positive.

Obviously, we can also start a predefined partition given to the linguistic variables and then try to understand each value in the expression for $\dot{V}$ you're using and from this find the rules, which is a somewhat more complex task.

Either way, whatever is done first, so far we have shown that the Fuzzy Lyapunov synthesis [5] transforms the classical Lyapunov approach from the world of conventional mathematics to the world of fuzzy system or the Computing with Words paradigm [10].

4 Results for Case 1

We show in this section the simulations of the dynamic model of the plant the water tank.

A controller for the water level in a tank has to know the current water level has to be able to set the valve.

The controller input is the water level error (desired water level minus the actual water level), and its output is the speed at which the valve opens or closes.

We can change the valve controlling the water flow, but the output rate depends on the diameter of the outlet pipe (which is constant) and the pressure in the tank (which varies with the level of water). The system has some very nonlinear characteristics.

4.1 Mamdani Fuzzy Controller

Simulation results for the Mamdani fuzzy controller are shown in Figures 1 and 2.

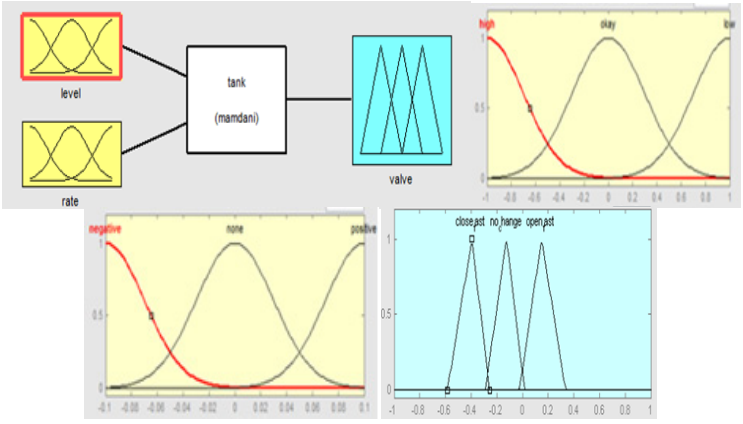


Fig. 1. Set membership functions for Type-1 Mamdani if the water tank

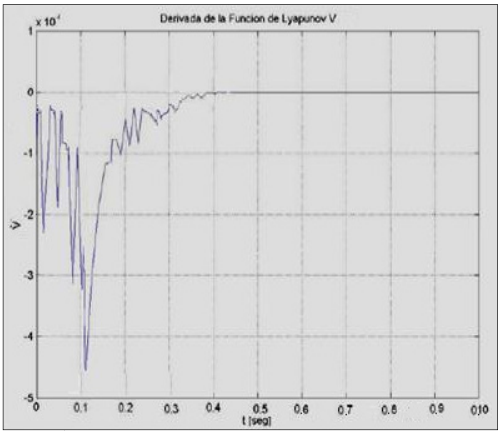


Fig. 2. V̇ for the plant with the Mamdani Fuzzy Controller is stabilized in time0.5 s

4.2 Sugeno Fuzzy Controller

Simulation results for the Sugeno fuzzy controller are shown in Figures 3 and 4.

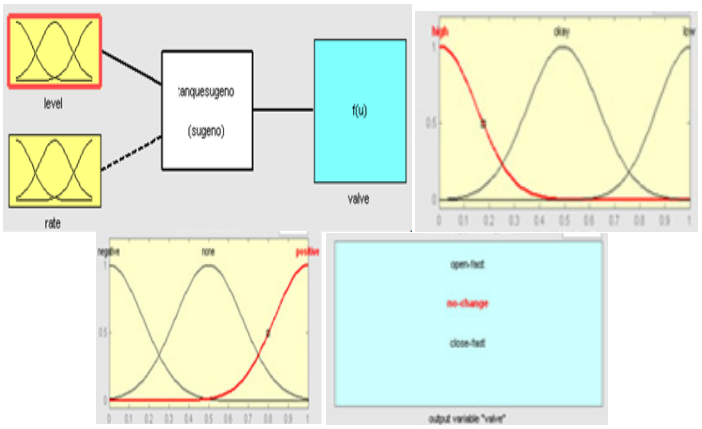


Fig. 3. Set membership functions for Type-1 Sugeno the case of water tank

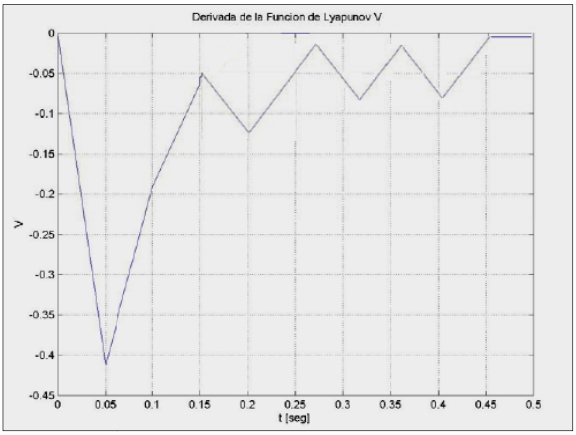


Fig. 4. $\dot{V}$ for the plant with the Sugeno Fuzzy Controller is stabilized in time 0.45 s

We show in table 1 a comparison of the results of bath controllers for case 1.

Table 1. Case 1 Comparison

Fuzzy Controller	Time (s) that stabilizes Comparative simulations of energy	Error
Mamdani	0.5s	0.0358
Sugeno	0.45s	0.0348

5 Results for Case 2

The Inverted Pendulum consists of a cart and a pendulum.

The regulator's objective is to move the carriage to its commanding position, without causing the pendulum to tip over.

5.1 Mamdani Fuzzy Controller

Simulation results for the Mamdani fuzzy controller are shown in Figures 5 and 6.

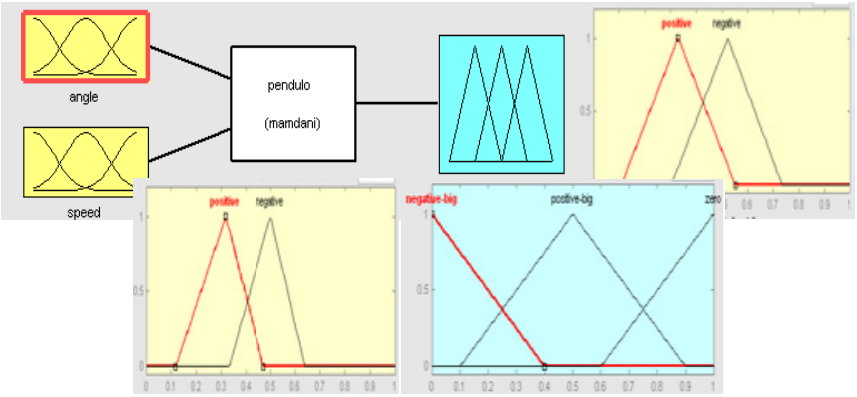


Fig. 5. Set of membership functions for type-1 Mamdani inverted pendulum case

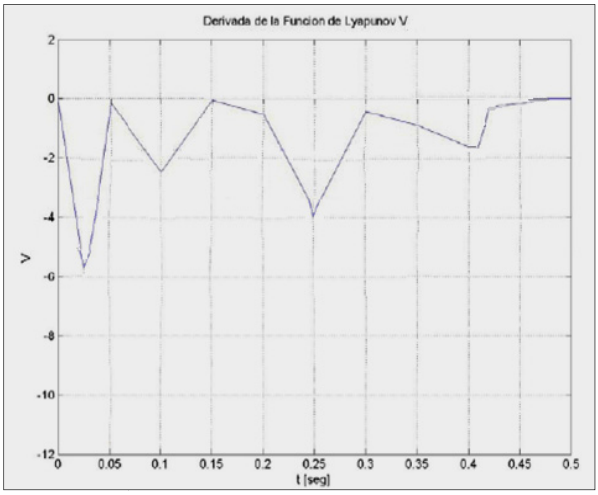


Fig. 6. $\dot{V}$ for the plant with the Mamdani Fuzzy Controller with xd = 0.5 rad, stabilizes in time 0.45s

5.2 Sugeno Fuzzy Controller

Simulation results for the Sugeno fuzzy controller are shown in Figures 7 and 8.

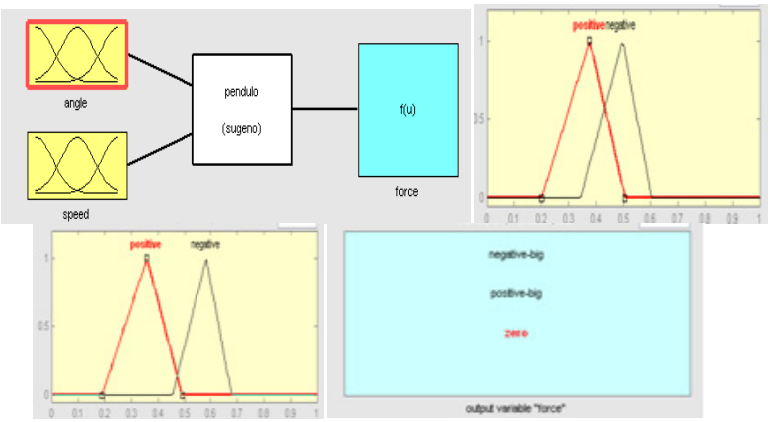


Fig. 7. Set of membership functions for type-1 Sugeno the case of the inverted pendulum

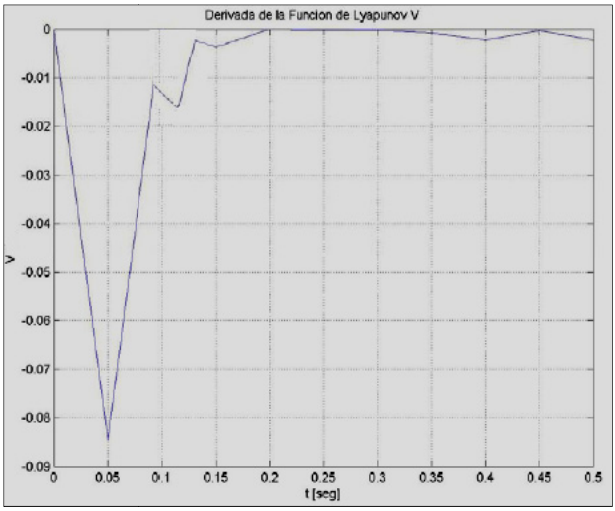


Fig. 8. $\dot{V}$ for the plant with the Fuzzy Controller with $x_d = 0.5$ rad, stabilizes in time 0.2 s

We show in Table 2 a comparison of the results of bath controllers for case 2.

Table 2. Case 2 Comparison

Fuzzy Controller	Time (s) that stabilizes Comparative simulations of energy	Error
Mamdani	0.5s	0.0458
Sugeno	0.45s	0.0356

6 Results of the Statistical Test

In each case, a procedure was developed, which was performed statistical tests "Z score", to test the validity of stability for cases where the Mamdani and Sugeno fuzzy controllers, which allowed the claim about a population parameter, this method is called a hypothesis test for the sample.

6.1 Statistical Test Results for Case 1 Mamdani

Shows descriptive statistics for hypothesis testing, the method of Lyapunov Mamdani and without the method respectively, for the water tank system.

Table 3. Statistical test for Mamdani Fuzzy System for case 1

Fuzzy System	*Sample*	*Null hypothesis* H_0	*Null alternative* H_1	*Significance level* $\propto$	*Average*	*Standard deviation*
Mamdani Lyapunov (n_1)	*30*	$\mu_1 \leq \mu_2$	$\mu_1 > \mu_2$	**0.05**	*0.2795*	*0.415*
Mamdani (n_2)	*30*				*-2.342*	**2.333**

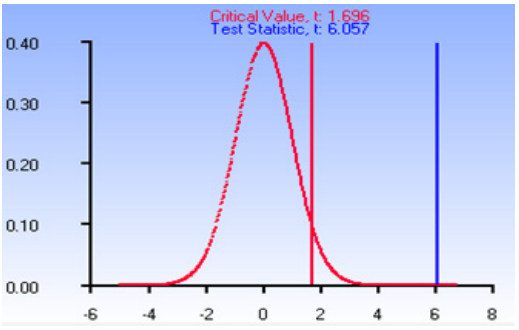


Fig. 9. Graphic of Mamdani hypothesis testing case 1

States that the test of null hypothesis is rejected with a confidence interval of 95%, the value of Z = 6.0571, the alternative hypothesis accepted, which gives us sufficient statistical information to determine that the method of Mamdani

Lyapunov is statistically greater than Mamdani, which there is evidence of the method.

6.2 Statistical Test Results Case 1 Sugeno

This section shows descriptive statistics for hypothesis testing with the method of Lyapunov for Sugeno and without the method respectively, for the water tank system.

Table 4. Statistical test for the Sugeno Fuzzy System for case 1

Fuzzy System	*Sample*	*Null hypothesis* H_0	*Null alternative* H_1	*Significance level* $\propto$	*Average*	*Standard deviation*
Sugeno Lyapunov (n_1)	*30*	$\mu_1 \leq \mu_2$	$\mu_1 > \mu_2$	**0.05**	*0.2106*	*0.375*
Sugeno (n_2)	*30*				*-2.145*	**2.1375**

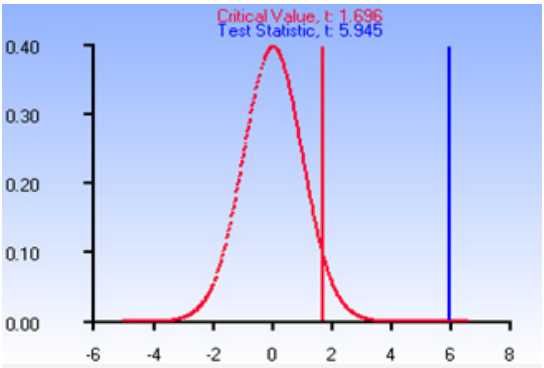


Fig. 10. Graphic of Sugeno hypothesis testing case 1

States that the test of null hypothesis is rejected with a confidence interval of 95%, the value of Z = 5.9453, the alternative hypothesis is accepted, which gives us sufficient statistical information to determine that the Sugeno method of Lyapunov is greater statistically Sugeno, which there is evidence of the method.

6.3 Statistical Test Results for Case 2 Mamdani

Shows descriptive statistics for the hypothesis test with the Mamdani Lyapunov method without the method respectively, for the inverted pendulum system.

Table 5. Statistical test for Mamdani Fuzzy System for case 2

Fuzzy System	*Sample*	*Null hypothesis* H_0	*Null alternative* H_1	*Significance level* $\propto$	*Average*	*Standard deviation*
Mamdani Lyapunov (n_1)	*30*	$\mu_1 \leq \mu_2$	$\mu_1 > \mu_2$	**0.05**	*0.144*	*0.270*
Mamdani (n_2)	*30*				*-2.815*	**2.8052**

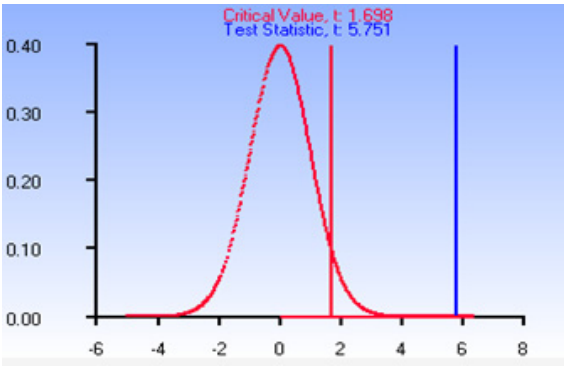


Fig. 11. Plot of Mamdani hypothesis testing for case 2

States that the test of null hypothesis is rejected with a confidence interval of 95%, the value of Z = 5.751, the alternative hypothesis is accepted, which gives us sufficient statistical information to determine that the Mamdani method of Lyapunov is greater statistically Mamdani, which there is evidence of the method.

6.4 Statistical Test Results Case 2 Sugeno

Shows descriptive statistics for hypothesis testing with the method of Lyapunov Sugeno and without the method respectively, for the inverted pendulum system.

Table 5. Statistical test for the Sugeno Fuzzy System for case2

Fuzzy System	*Sample*	*Null hypothesis* H_0	*Null alternative* H_1	*Significance level* $\propto$	*Average*	*Standard deviation*
Sugeno Lyapunov (n_1)	*30*	$\mu_1 \leq \mu_2$	$\mu_1 > \mu_2$	**0.05**	*0.181*	*0.346*
Sugeno (n_2)	*30*				*-2.612*	**2.5047**

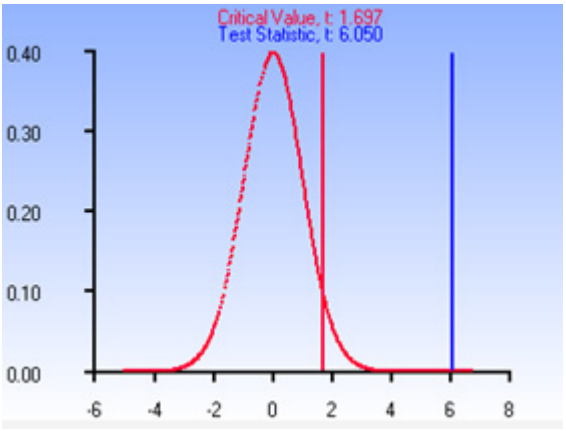


Fig. 12. Plot of Sugeno hypothesis testing case 2

States that the test of null hypothesis is rejected with a confidence interval of 95%, the value of Z = 6.0502, provides statistical information sufficient to determine that the Sugeno method of Lyapunov is greater statistically Sugeno, which evidence of the methods.

7 Conclusions

The main objective of this research was to propose a systematic methodology to design stable fuzzy controllers to solve different cases.

With the proposed method stability of control was achieved in two cases based on simulation results.

The problems for the two cases are resolved as expected; this statement is consistent with simulations, where fuzzy controllers are designed following the method of fuzzy Lyapunov synthesis to achieve the solution to the problem.

References

[1] Cazares Castro Nohe, R.: Estabilidad en sistemas de control difuso ti-po-2. Instituto Tecnológico de Tijuana, México (2005)
[2] Feng, G., Sun, D., IEEE: Generalized H2 Controller Synthesis of Fuzzy Dynamic Systems Based on Piecewise Lyapunov Functions (2002)
[3] Lee, H.: On methods for improving performance of PI-type fuzzy logic controllers. IEEE Transactions on Fuzzy Systems 1, 298–301 (1993)
[4] Margaliot, M., Langholz, G.: A new approaches to Fuzzy Modeling and Control: Design and Analysis. World Scientific, Singapore (2000)
[5] Margaliot, M., Langholz, G.: A new approach to the design of fuzzy control rules. In: Proceedings of the International Conference on Fuzzy Logic and Applications (Fuzzy 1997), Israel, pp. 248–254 (1997)
[6] Margaliot, M., Langholz, G.: Adaptive fuzzy controller design via fuzzy Lyapunov synthesis. In: Proceedings of the IEEE International Conference on Fuzzy Systems (FUZZY-IEEE 1998), Alaska, EEUU (1998)
[7] Mamdani, E.H., Assilian, S.: An experiment in linguistic synthesis with fuzzy logic controller. International Journal on Man-Machine Studies 7, 1–13 (1975)
[8] Cázarez-Castro, N.R., Aguilar, L.T., Castillo, O.: HIS: 216-221, HybridGenetic-FuzzyOptimization of a Type-2 FuzzyLogicController. Instituto Tecnológico de Tijuana, México (2008)
[9] Zadeh, L.: Fuzzy Logic = Computing With Words. IEEE Transactions on Fuzzy Systems 4(2), 103–111 (1996)
[10] Zadeh, L.: Fuzzy logic. IEEE Computer (1998)
[11] Zadeh, L.A.: The concept of a linguistic variable and its application to approximate reasoning – I. Information Sciences 8, 199–249 (1975)
[12] Zadeh, L.: Fuzzy Logic = Computing With Words. IEEE Transactions on Fuzzy Systems 4(2), 103–111 (1996)

Design of Fuzzy Control Systems with Different PSO Variants

Resffa Fierro and Oscar Castillo

Tijuana Institute of Technology

Abstract. This paper describes the metaheuristic of Optimization by Swarm of Particles (PSO-Particle Swarm Optimization) and its variants (Clamping speed, inertia and constriction coefficient) as an optimization strategy to design the membership functions of Benchmark Control Cases (Tank water and Inverted Pendulum) Each of the variants have their own advantages within the algorithm because they allow the exploration and exploitation in different ways and this allows us to find the optimum.

1 Introduction

At present time and within computer science, optimization is a key issue specifically in real-world problems. In analyzing these problems there is the difficulty of ensuring a good solution in a reasonably short time.

There are a variety of methods used to solve optimization problems. In particular, it is interesting to find heuristics that are approximate methods of solution, using an iterative process to guide the search for solutions, thus combining different concepts derived from fields such as Artificial Intelligence, Biological Evolution and Collective Intelligence.

This work is no exception in that we use the metaheuristic of Optimization by Swarm of Particles (PSO - Particle Swarm Optimization) developed by Kennedy and Eberhart in 1995 [4], where this is inspired by the behavior of flocks of birds, of fish and bees. This algorithm will be considered along with by three of its variants which are clamping speed, inertia and constriction.

In the literature it is has been said that since this metaheuristic appeared in 1995 to the present it has had a major impact in the optimization of complex problems.

2 Metaheuristic Optimization by Swarms of Particles

Basically the PSO metaheuristic, is based on an iterative algorithm where a population of individuals called Swarm and each of the individuals are called particles, fly over a search space to find optimal solutions. Where each particle is a possible solution to a particular problem. More precisely we have particles flying in a multidimensional space where the position of each particle is adjusted according to its

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 81–88.
springerlink.com

own experience and that of their neighbors. It is a technique inspired by the flight patterns of birds; this simulates the movements of a flock trying to find food. Developed initially by Kennedy and Eberhart in 1995 [4].

We have that x_i (t) is the position of the particle in the search space in a time t, unless otherwise indicated, t denotes the time steps. The position of the particle is the change in velocity v_i (t), to the current position, i.e.:

$$x_i(t+1)=x_i(t)+v_i(t+1) \quad (1)$$

$_i$: index of particles
t: time index

x_i: position of the particle

v_i: velocity of the particle

To calculate the velocity of the particle the following equation is used:

$$v(t+1)=v_{ij}(t) + c_1 r_{1j}(t)[y_{ij}(t)-x_{ij}(t)] + c_2 r_{2j}(t)[\hat{y}_j(t)-x_{ij}(t)] \quad (2)$$

Where we have the following parameters forming Equation (2):

i: index of particles
t: time index

v_i: velocity of the particle

x_i: position of the particle

y_{ij}: best position found by particle i (personal best)

$\hat{y}_j$: best position found by the swarm (best overall)

c_1,c_2: constant acceleration

r_1,r_2: random numbers in the interval [0,1] applied to the particle.

Table 1 shows the classical PSO algorithm and Figure 1 illustrates the particle movement based on the algorithm of Table 1.

3 Variants of PSO

In this section several variants of PSO are presented.

3.1 Clamping Speed

What determines if an optimization algorithm is efficient and accurate is the exploration and exploitation. Exploration is the search capability to explore different regions and find a good optimum. Exploitation, on the other hand, is the ability to concentrate the search in a certain area to find a candidate solution.

Table 1. Classical PSO Algorithm

Create and Initialize the cloud
t= 0
Cloud ← Initializing particle cloud
While not at the stop condition do
t=t+1
For i = 1 to size (Swarm) to
Evaluate each particle x_i of the Swarm
If fitness xi is better than fitness_mejorpos$_i$ then
mejorpos$_i$ ← x_i
fitness _mejorposi ← fitness_ x_i
end If
If fitness _mejorpos$_i$ is better than fitness _mejorpos$_i$ then
mejorpos ← mejorpos$_i$
fitness _mejorpos ← fitness _mejorpos$_i$
end If
For i = 1 to size (Swarm) to
Calculate the speed of v_i x_i based on the values (2)
Calculate the new position of x_i, its current value and v_i (1)
F in for
end while
Output: Return the best solution found

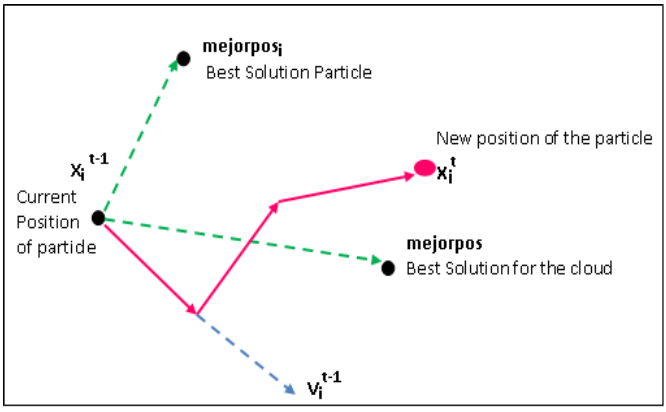


Fig. 1. Particle movement taking into account the algorithm of Table 1

In this case the following equation is used to calculate the speed of clamping where we have:

$$v_{ij}(t+1)= \quad v'_{ij}(t+1) \text{ if } v'_{ij}(t+1) < V_{max,\,j} \tag{3}$$

$$V_{max,\,j} \text{ if } v'_{ij}(t+1) \geq V_{max,\,j}$$

$$V_{max,\,j} = \boldsymbol{\delta}(x_{max,j} - x_{min,j})$$

The value of $V_{max,\,j}$ is very important because this allows the search in more detail; large values allow global exploration, while smaller values allow local working.

3.2 Inertia Factor

PSO calculates the new position of each particle as a function of the flight, updating its position after having calculated the new position. This inertia factor was introduced to reduce the influence that the search direction brings the particle. The equation for calculating the new speed of the particle is as follows, but is already accompanied by w that represents the inertia:

$$v_{ij}(t+1)= \mathbf{w} \cdot v_{ij}(t)+ c_1 r_{1j}(t)[y_{ij}(t)-x_{ij}(t)]+c_2\, r_{2j}(t)[\hat{y}_j(t)-x_{ij}(t)] \tag{4}$$

Now the new speed ($v_{ij}(t+1)$) is determined by taking into account the following parameters:

"w". The inertia weight or inertia factor is a value, which regulates the influence of the previous velocity of the particle v (t) in calculating the new speed v (t + 1), by way of regulating the flight of the particle making a balance between exploitation and exploration of the search space. The inertia factor under certain conditions promotes the convergence of the cloud, i.e. all the particles approach the leader of the cloud.

Now v(t), Represents the current speed of the particle that was previously calculated by the same equation, refers to the direction of flight having the particle.

The Acceleration coefficients $c_1 * r_{1j}$ () and $c_2 * r_{2j}$ () and r represents a random real number uniformly distributed between 0.0 and 1.0. The coefficient c1 regulates the influence of cognitive knowledge of the particle and the coefficient c2 regulates the influence of knowledge social.

c_1 regulates the influence of the best position reached by the particle ($\mathbf{y}_{ij}$) to guide your new address and c_2 regulates the influence of the cluster leader ($_j$) in the search direction of the particle.

$\mathbf{y}_{ij}$. Represents the memory of the particle, its best position achieved so for in that generation.

$\hat{\mathbf{y}}_j$. Represents the best position of the swarm, i.e. the leader.

$\boldsymbol{x}$. Represents the current position of the particle is taken as reference for calculating the new speed.

v (t+1) is the speed of the current particle will determine the new search direction of the particle in generation t + 1. After calculating the current speed v (t +1) updates the particle position using equation (1).

3.3 Constriction Coefficient

Proposed by Clerc, where this variant affects the rate and calculated update and the equation is modified as follows:

$$v_{ij}(t+1)=\mathbf{X}([v_{ij}(t)+\Phi_1(y_{ij}(t)-x_{ij}(t))+\Phi_2(\hat{y}_j(t)-x_{ij}(t))]) \quad (5)$$

The constraint factor X is calculated using acceleration factors and ϕ_2 ϕ_1.

Where

$$x = \frac{2k}{|2-\phi - \phi(\phi-4)|} \quad (6)$$

With

$$\Phi = \Phi_1 + \Phi_2$$
$$\Phi_1 = c_1 r_1$$
$$\Phi_2 = c_2 r_2$$

Using the constraints where $\phi \geq 4$ and k ε [0,1]. For equation (6)

4 Statement of the Problem

The problem is to optimize the membership functions of fuzzy controllers for Benchmark cases starting with the water tank where the 3 variants are used to find out which works best to find results. Below is the proposed architecture.

The proposed architecture is illustrated in Figure 2.

It is known in advance by reviewing the literature and previous works that to develop fuzzy controllers and obtain good results optimizing membership functions is not easy it takes time and good design that is given to this. That's why this work intends to make the optimization of membership functions with variants of the PSO and find out how efficient and fast can be to find good results.

Here are some tests that are being conducted using two variants of the PSO (Factor of inertia and constriction coefficient).

In 5 tests that were performed where these results were obtained.

Table 2 shows experiments with the two variants of PSO.

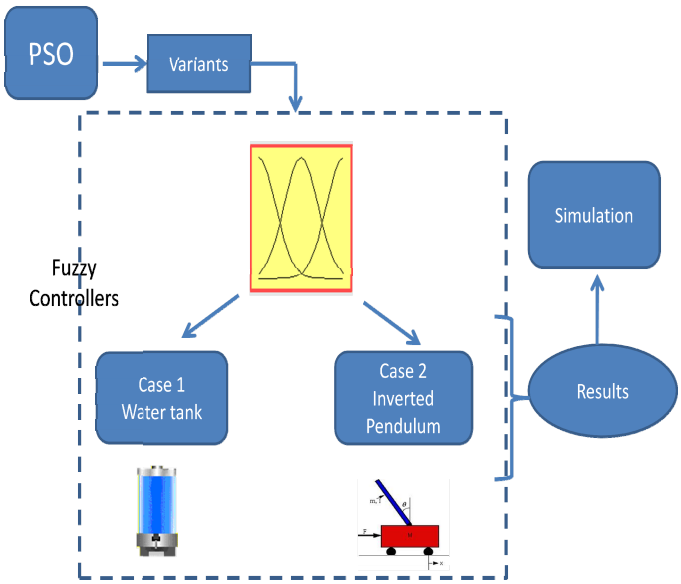


Fig. 2. Architecture of PSO optimization for fuzzy control

Table 2. Experiments with two variantes of PSO

Experiment	Iteration	Size Swarm	Inertia	Constriction	Average error
1	10	20	1.0	0.2	0.42465
2	10	20	1.0	0.2	0.026217
3	10	20	1.0	0.2	0.011472
4	10	20	1.0	0.2	2.521
5	10	20	1.0	0.2	0.011315

Being worked by making further experiments to obtain better results, conducting experiments with each of the variables.

As shown in Table 2 the best result was obtained in experiments is the number 5 which was performed with 10 iterations and 20 particles, using two variants of PSO, obtaining an error of 0.011315. Figure 3 shows the performance of PSO.

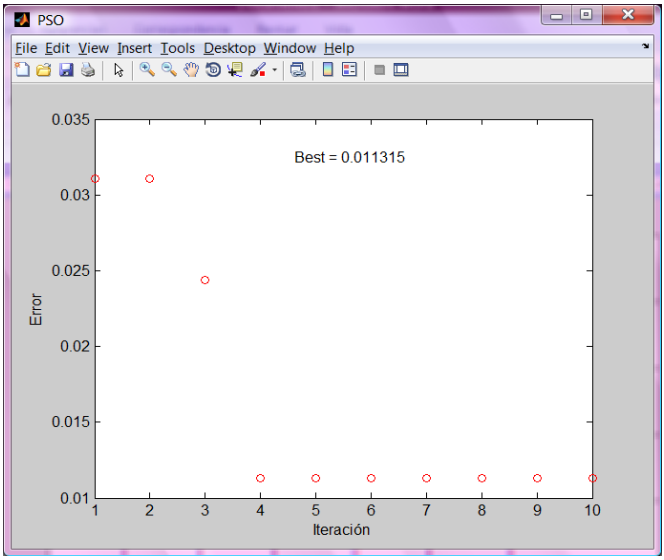


Fig. 3. The following figure shows the performance of PSO

Ahead shown in Figure 4, we have the resulting simulation using the two variants of PSO, we see that almost made it entirely to get the reference.

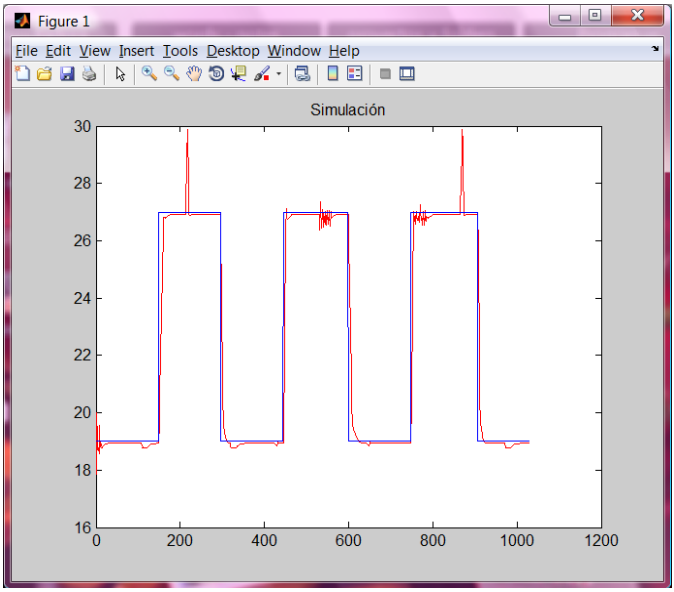


Fig. 4. Benchmark simulation of the water tank

5 Conclusions

In this paper the design of fuzzy control systems has been illustrated with PSO variants to verify their efficiency and accuracy, although we continue currently working on achieving better results. It is noted that the simulation results show the potential use of these optimization methods for this type of control problems.

References

[1] Andries, P., Engelbrecht, A.: Fundamentals of Computational Swarm Intelligence, pp. 85–129. Wiley, England (2005)
[2] Birge, B.: PSOt: A Particle Swarm Optimization Toolbox for Matlab. In: IEEE Swarm Intelligence Symposium Proceedings, April 24-26 (2003)
[3] Del Valle, Y., Kumar, G.V., Mohagheghi, S., Hernandez, J.C., Harley, R.G.: Particle Swarm Optimization: Basic Concepts. Variants and Applications in Power Systems (April 2008)
[4] Kennedy, J., Eberhart, R.: Particle Swarm Optimization. In: IEEE Int´l. Conf. on Neural Networks, Perth, Australia, pp. 1942–1948. IEEE Service Center, Piscataway (1995)
[5] Kennedy, J., Shi, Y.: Swarm Intelligence. Academic Press, Inc. (2001)
[6] Kennedy, J., Shi, Y.: Swarm Intelligence. Morgan Kaufmann Publishers, San Francisco (2001)
[7] Kennedy, J.: The Particle Swarm: Social Adaptation of Knowledge. In: IEEE International Conference on Evolutionary Computation, pp. 303–308 (1997)
[8] Zadeh, L.A.: Fuzzy Logic. IEEE Computer, pp. 83–92 (1965)

Methodology to Design Fuzzy Logic Controller for Soft-Core Embedded into FPGA

Roberto Sepúlveda, Oscar Montiel-Ross, and Jorge Quiñones

Instituto Politécnico Nacional - CITEDI. Av. del Parque 1310, Tijuana, B., C., México
{r.sepulveda,o.montiel}@ieee.org, jorge@citedi.mx

Abstract. In this paper, the methodology for the design of fuzzy controllers for softcore processors, such as the Xilinx Microblaze embedded in the VIRTEX5 FPGA, is proposed to regulate the angular position of the axes of an experimental platform. The platform uses servomotors to control the rotational movements of the X-Y-Z axes, this with respect to the earth horizon. The angular position is feedback using three inclinometers sensors based on MEMS technology with SPI interface. The desired position is regulated using three independent fuzzy PD+I controllers, which use the error and change of error as input signals. The proposed methodology consists in the design and evaluation of the fuzzy controllers using the Fuzzy Logic Toolbox of Matlab.

Keywords: Soft-core, Fuzzy controllers, FPGA.

1 Introduction

Many of the chemical processes, electrical, mechanical, etc., require electronic control systems to regulate their behavior, algorithms are designed to obtain control of the process or "plant" [1]. To carry out the control, there is a lot of techniques and strategies, one of them is based on the natural language of humans, this discipline is called "fuzzy control" which is considered a rule-based control usually raised by the knowledge of an expert. To carry out the fuzzy control it is necessary to use processing systems for the necessary calculations involved in this process, for this reason in subsequent sections of this work, we propose an alternative fuzzy controller design implemented in programmable logic devices [2], and FPGAs particular devices.

Given its multidisciplinary nature, a fuzzy inference system (FIS), have the advantage of using human knowledge, which is crucial in control problems, where it is difficult to construct accurate mathematical models [1]. A FIS is composed of a fuzzification interface, a rule base, database and decision-making unit, and a defuzzification stage, as it is shown in Fig. 1.

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 89–96.
springerlink.com

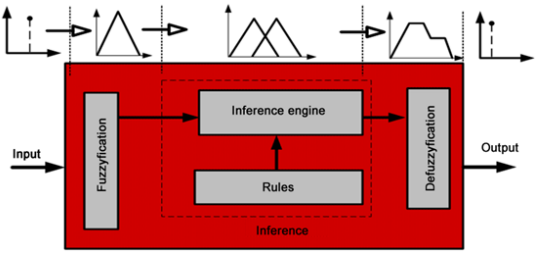


Fig. 1. Fuzzy Inference System

The purpose of a FIS is to model a specific system with the incorporation of human experience in the form of fuzzy rules, and based on these rules, to take a decision. For example:

$$If\ x\ is\ a\ and\ y\ is\ b\ then\ z\ is\ c.$$

The processors are classified into two categories "Hard" and "Soft", this classification refers to the flexibility and configurability of the architecture. The "Hard" processor is dedicated and physically embedded in silicon, such as we see in conventional processors computers, microcontroller, DSP, they have a fixed architecture that cannot be changed or altered. The "soft processor", Fig. 2, uses as a basis the existing programmable logic resources for the FPGA to implement the logic of a specific processor architecture, as well as a customizable software, with the advantage of being able to add modules such as IIC, Ethernet, PCI Express, accountants, USB, memory, coprocessors, etc., many of them are encapsulated in IP cores, within these processors are the Microblaze from Xilinx [3], Altera Nios II, Lattice LatticeMico32, PowerPC440 IBM, etc. .

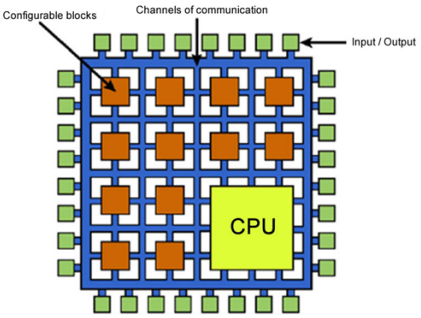


Fig. 2. A soft Processor

2 Fuzzy Controller Design for an FPGA

This section presents the methodology used for the development of digital controllers embedded in FPGA technology [4, 6]. It is proposed an experimental platform

in order to control the angle of inclination of its axes, this control platform development is based on a soft processor, it is interconnected to sensors to measure real-time readings of its position [7, 8]. For the development of FPGA soft processors, Xilinx has incorporated into its software a development suite Xilinx Platform Studio tool and SDK, needed to create the hardware profile that incorporates the Microblaze and PowerPC cores [3].

For fuzzy control applications, the control strategy is based on rules usually raised by the knowledge from experts, this is crucial in applications where it is difficult to construct accurate mathematical models [8-12]. To get an overview of the system being controlled in Fig. 3 a scheme of the experimental platform is shown. To solve the problem of angular position control, through the axes of the platform, a PD + I fuzzy controller (Proportional-Integral Derivative more action) is proposed, it has two inputs and one output [6]. The application aims to regulate the angle of inclination of the platform to a desired value using a control system, with a Microblaze soft processor with 32-bit architecture embedded in an FPGA.

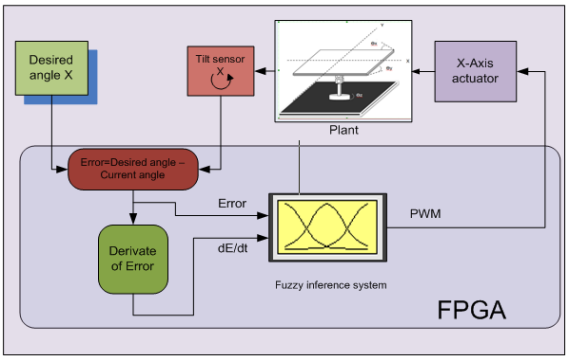


Fig. 3. Scheme of the experimental platform

The fuzzy inference system uses two input variables, one variable is the error, which is calculated by:

$$e(t) = r(t) - y(t) \tag{1}$$

where:

$e(t)$ =Error.
$r(t)$ =Set Point.
$y(t)$ =Current value.

The second variable in our FIS is the derivative of the error and it is calculated by the equation:

$$\Delta e(t) = e(t) - e(t-1) \tag{2}$$

where:

$\Delta e(t) =$ Derivate of error.
$e(t) =$ Current error.
$y(t-1) =$ Previous value.

The output variable will control the actions, and it is directly related to the response to exercise servomotors exert the rotational motion of the platform in a shaft. For the analysis, we propose a proportional controller with incremental output differential, since the output response will be added depending on the error in an attempt to approximate the desired point until the establishment. The first step is to create a fuzzy inference system with fuzzy logic toolbox of Matlab.

The universe of discourse of the linguistic variables "error" and "change of error" is in the range of $[-360°, 360°]$, these values were chosen objectively because the maximum error of measurement is $360°$. The distribution of the triangular membership functions of the variables "error" and "change of error", are denoted MN (Very negative), N (Negative), C (Zero), P (Positive) MP (very positive), which are illustrated in Figures 4 and 5.

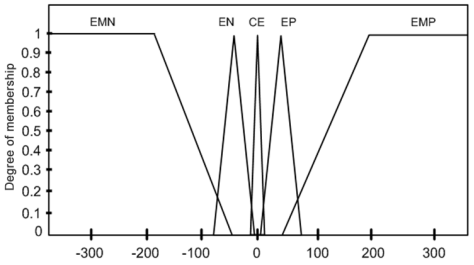


Fig. 4. Distribution of membership functions of the input variable "Error"

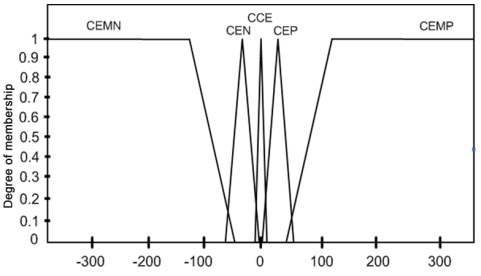


Fig. 5. Distribution of membership functions of the input variable "Change of error"

Five membership functions comprise the variable "output", they are denoted as DMN (Decrease Very Negative), DN (Decrease Negative), CI (Zero Increase), IP (Increase Positive) IMP (Increase Very Positive). The output provides the control

action to regulate the movement of the motors, which exert an incremental positive or negative rotational movement. The proposed ranges, shown in Fig. 6, were selected taking into account the operating parameters of the motors placed on the platform.

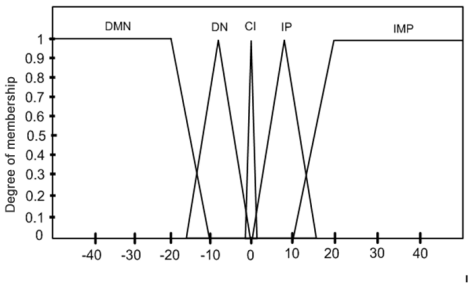


Fig. 6. Distribution of membership functions of the output variable "Output"

For the operation of the fuzzy controller, the Mamdani method was used [1], and we have twenty-five rules because we have five linguistic terms in each of the two variables (5x5), Fig. 7 shows the matrix rules.

Δe / e	CEMN	CEN	CCE	CEP	CEMP
EMN	DMN	DMN	DMN	DN	DN
EN	DMN	DN	DN	CI	IP
CE	DMN	DN	CI	IP	IMP
EP	DN	CI	IP	IP	IMP
EMP	IP	IP	IMP	IMP	IMP

Fig. 7. Matrix system of inference rules

Once designed the fuzzy inference system, the implementation must take a computer processing system, for our purposes the application of control tools were created to carry out the implementation of embedded soft processor in a Virtex 5 FPGA, through the ML507 development team of the company Xilinx Inc., then to evaluate the real-time operation of the fuzzy controller for the experimental platform. Matlab provides an independent tool for modeling, simulating or evaluating fuzzy inference systems (FIS), it consists of a library written in ANSI C language, it is located in the subdirectory named facilities "fis.c".

3 Results

The tests conducted are based primarily on embedding algorithms PD + I fuzzy controller in an FPGA development system manufactured by Xilinx Inc. ML507,

installed in the testbed. The Microblaze performance, shown in Fig. 8, met the desired requirements for testing the fuzzy controller, obtaining a sampling rate of 13 ms, to evaluate all functions related to the process of fuzzification inference and defuzzification at a frequency of 50 MHz clock.

C function	Runtime (ms)
returnFismatrix	97
ErrorAnteriorEjeX	12
fisBuildFisNode	64
EvalRules	13

Fig. 8. Performance of Microblaze in C functions

To check the operation of the PD + I controller, the controller performs the experiment with a desired angular position for the X axis equal to 45±, in the graph of Visual Basic, shown in Fig. 9, the measurement is obtained in real time using an inclinometer through serial communication and as can be seen, the platform reaches the target with an acceptable response of 0.4 seconds.

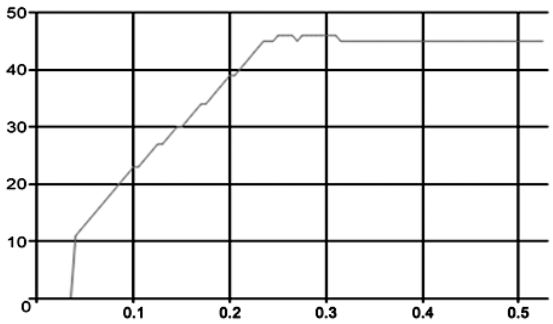


Fig. 9. Transient response of the controller

The final position of the platform can be physically seen in Fig. 10, note that the controller's performance met an acceptable performance and reflects the expected behavior based on the proposed rules, also notes that it arrives at the platform quickly, without causing pronounced oscillatory movements.

Another test was to enter a disturbance at the moment that the platform reaches the desired value. This was obtained by applying a force on its own axis with the intention of destabilizing it and checking the effectiveness of the logic implemented in that situation. The response to a disturbance applied to the platform, can be seen in Fig. 11, and it is observed that the platform is able to respond appropriately, it quickly returns to the desired value without unstable.

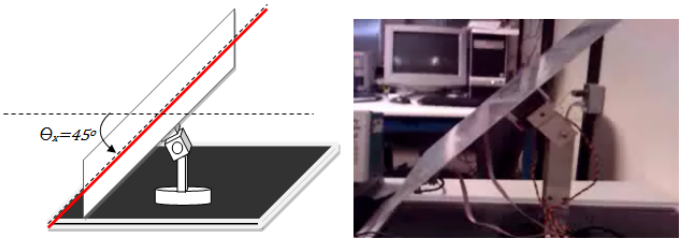


Fig. 10. The Experimental platform

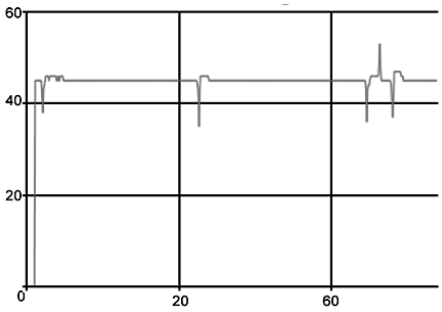


Fig. 11. Controller's response to a disturbance

To analyze the controller' s performance, the experimental platform control surface is shown in Fig. 12, which is represented by a three-dimensional graphical output response, as a function of the linguistic variables error and change of error.

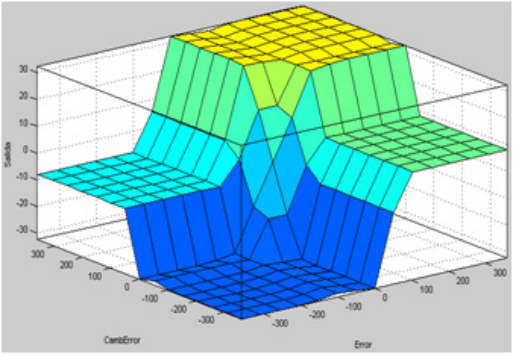


Fig. 12. Control surface

4 Conclusions

The proposed methodology reduces the design time for the creation of a fuzzy inference system using FPGA embedded soft processors, since we took advantage of creating the FIS with the Matlab toolbox, with the benefits that it provides, like graphical interface and simulink. Once the system was designed and tested, it can be exported to custom hardware, for example, the FPGA for rapid real time prototype.

Acknowledgment. The authors thank Instituto Politécnico Nacional, Comision de Operación y Fomento de Actividades Académicas del Instituto Politécnico Nacional, and CONACYT for supporting our research activities.

References

1. Mamdani, E.H., Assilan, S.: An experimental in linguistic synthesis with a fuzzy logic controller. International Journal of Man-Machines Studies, 1–13 (1975)
2. Togai, H., Watanabe, H.: Expert System on a chip: an engine for real time approximate reasoning. In: Proceeding of the ACM SIGART International Symposium on Methodologies for Intelligent Systems, pp. 147–154. ACM Press, New York (1986)
3. Xilinx Corporation, EDK OS and Libraries Reference Manual, Xilinx, pp. 100–120 (2008)
4. Zeidman, B.: Designing with FPGAS and CPLDS, pp. 30–80. Elsevier (2003)
5. Jim, L.: Embedded Control Systems in C/C++, pp. 1–192. CMP Books, Berkley (2004)
6. Montiel, O., Sepúlveda, R., Olivas, J., Castillo, O.: Development of an embedded simple tuned fuzzy controller. In: IEEE International Conference on Fuzzy Systems, FUZZ-IEEE 2008 (IEEE World Congress on Computational Intelligence), pp. 1–7 (2008)
7. Dimu, A., Khor, J.: Neural and Fuzzy Logic Control of Drives and Power Systems. Newnes, 237–160 (2002)
8. Precup, R.-E., Hellendoorn, H.: A survey on industrial applications of fuzzy control. Computers in Industry 62(3), 213–226 (2011)
9. Miyamoto, S.: Automatic train operation by predictive fuzzy control. In: Sugeno (ed.) Industrial Applications of Fuzzy Control, Amsterdam, pp. 1–18 (1985)
10. Altrock, C.V.: Fuzzy logic technologies in automotive engineering. In: Embedded Systems Conference, vol. 2, pp. 407–422 (1994)
11. Cherry, A.S., Jones, R.P.: Fuzzy logic control of an automotive suspension systems. In: Proc. Inst. Elect. Eng. D: Control Theory Application, vol. 142(2), pp. 149–160 (1995)
12. Lennon, W.K., Passino, K.M.: Intelligent Control for Brake Systems. IEEE Trans. on Control Systems Technology 7(2), 188–202 (1999)

Optimization of Membership Functions for Type-1 and Type 2 Fuzzy Controllers of an Autonomous Mobile Robot Using PSO

San Juana Aguas-Marmolejo and Oscar Castillo

Tijuana Institute of Technology, Tijuana, Mexico
ocastillo@tectijuana.mx

Abstract. This paper describes the application of the optimization algorithm based on particle swarms known by its acronym as PSO, used to adjust the parameters of membership functions of a fuzzy logic controller (FLC) to find the optimal intelligent control for a wheeled autonomous mobile robot. Results of several simulations show that the PSO is able to optimize the type-1 and type 2 FLCs for this specific application.

1 Introduction

Over the years, science and technology has advanced by leaps and bounds. Sometimes faster sometimes slower, but always moving forward. It is with this development that new branches and areas of application have emerged. One of them, and for this research, the most important, is the branch of control applied to autonomous mobile robots.

There are many techniques and control algorithms for these robots, each with its own characteristics, each of them and they offer advantages and disadvantages that allows the scientist the option to use the one you serve for your home work or research. Given the specialization of new mechanisms good control techniques are required that are able to offer a satisfactory job in performing this task.

One such technique is called "Fuzzy Logic" (Fuzzy, Fuzzy), which is considered one of the best. This is its versatility, flexibility and simplicity. This technique became known in the years 60 (in 1965 by Lotfi A. Zadeh, professor at the University of California Berkeley) for the article "Fuzzy Sets" Information and Control.

Fuzzy logic is one of the most used methods of computational intelligence and the better future, this is possible thanks to the efficiency and simplicity of fuzzy systems because they use linguistic terms similar to those that humans use.

The complexity in the development of these fuzzy systems can be found in deciding what the best parameters of membership functions, the number of rules, or even the best detail that could give us the best solution to the problem we want solve.

A solution to the problem mentioned is the application of bio-inspired algorithms for optimization of fuzzy systems. Optimization algorithms can be a useful tool because of its ability to solve nonlinear problems either limited or even NP-hard problems. Among the most widely used optimization methods are:

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 97–104.
springerlink.com

genetic algorithms (GA), Ant Colony Optimization (ACO), particle swarm optimization (PSO), etc. [1].

This paper describes the application of particle swarm algorithm (PSO) as a method of optimizing the parameters of membership functions of the proposed fuzzy logic controller (FLC) in order to find the best intelligent controller for an autonomous mobile robot wheel.

2 Optimization Algorithm Using Particle Swarm Optimization (PSO)

Optimization by clouds or swarms of particles (PSO) [2] [3] is a relatively new technique that is slowly taking rise and recognition as an effective and efficient algorithm. While PSO algorithm shares similarities with other evolutionary computation techniques while also differs in certain respects and needs no evolution operators such as crossover and mutation.

PSO emulates the swarm behavior of insects, a herd of grazing animals, a cloud of birds, a host of fish in these swarms or clouds that made the search for food in a collaborative manner. Each member of a swarm adapts its search patterns, learning from his own experience and experiences of other members, i.e. taking into account their cognitive beliefs and social beliefs.

These phenomena are discussed in the algorithm and the mathematical models are built on the method for updating the positions of each particle.

In the PSO algorithm, a member in the swarm, called a particle, represents a possible solution is a point in the search space. The global optimum is considered as the location of food, the problem would be implemented as the optimal solution found. Each particle has a fitness value and a speed to adjust the flying direction according to the best.

The general formula (equation 1) for determining the motion of particles which are presented below, is shown in two parts the cognitive and social part of which are crucial to identify the type of algorithm that is being implemented in our case we used the Full GBEST ie both C1 and C2 must have values greater than 0 but less than 4, respectively [4].

$$V_{id} = V_{id}\,(t+1) + C_1 r_{id}(t)\,[Y_{id}(t) - X_{id}(t)] + C_2 r_2(t)\,[Y_{id}(t) - X_{id}(t)] \quad (1)$$

There is another formula (equation 2) that is critical for the update of each particle, this assesses the current position of the particle and the previous position to choose which is the most appropriate to find more quickly the result this position is recalculated at each new iterations that is the algorithm.

$$X_i(t+1) = X_i(t) + V_i\,(t+1) \quad (2)$$

3 Design of the Fuzzy Controller

As determined by one of the most used and effective techniques to solve control problems is to use fuzzy systems to meet the control objective, so it is necessary to design a fuzzy controller for the actual speed of the mobile robot.

In this research work we use a type inference systems using Takagi-Sugeno type fuzzy system in order both 1 and 2, the use of linguistic variables in the input and output mathematical functions.

The errors of the linear and angular velocities (respectively) were taken as input variables, while the left and right pairs are taken as outputs. The membership functions used in the entry are trapezoidal for negative (N) and positive (P), and a triangle was used to zero (C) linguistic terms [5]. Figure 1 shows the input and output variables used, these are used for both types of fuzzy logic (1 and 2).

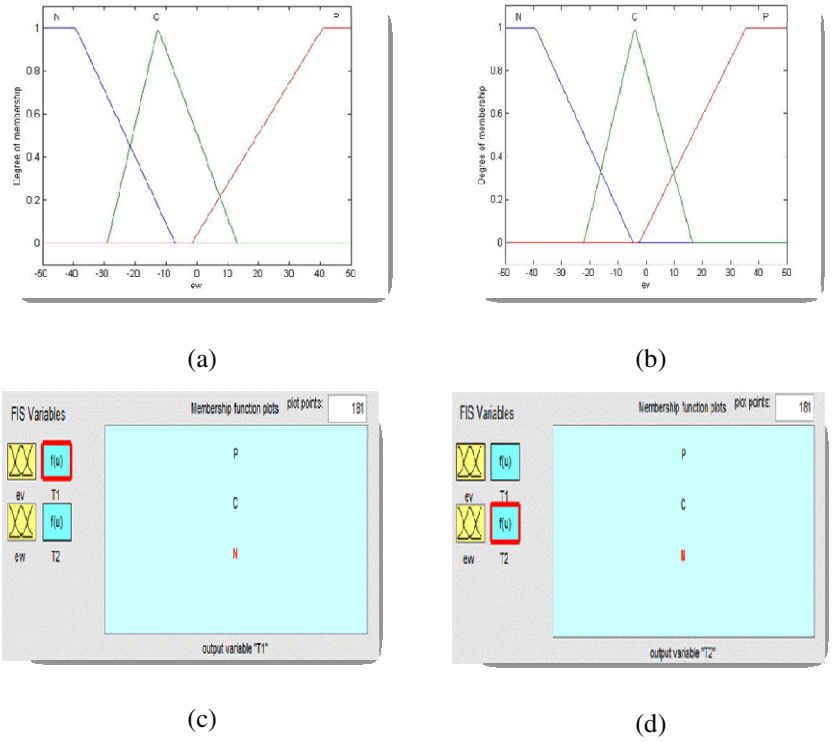


Fig. 1. Variable input / output: (a) error of the linear velocity (ev). (b) Angular velocity error (ew). (c) right torque output (τ 1). (d) left torque (τ 2).

The FLC has 9 rules [6], which are adapted to the style of Takagi-Sugeno controller, so the output has a single point, so the results are constants values (P, C, N), which obtained through a procedure using a weighted average defuzzification by.

ϖ R1: If Vangular is C VLineal is C, then τ 1 is C and τ2 is C
ϖ R2: If Vangular is C VLineal is P, then τ 1 is C and τ 2 is P
ϖ R3: If Vangular is C and VLineal is N, then τ 1 is τ 2 is N
ϖ R4: If Vangular P and VLineal it is C, then τ 1 is P and τ 2 is C
ϖ R5: If Vangular is P and VLineal is P, then τ 1 is P and τ 2 is P
ϖ R6: If VLineal is P and Vangular is N, then τ1 is P and τ 2 is N

ϖ R7: If VLineal is N and Vangular is C, then τ 1 is N and τ2 is C
ϖ R8: If VLineal is N and Vangular is P, then τ 1 is N and τ 2 is P
ϖ R9: If Vangular is N and VLineal is N, then τ 1 is N and τ 2 is N

The linguistic terms of input variables are shown in the first row and column of Table 1, the rest of the content corresponds to the linguistic terms of output variables.

Table 1. FLC Rules

ev/ew	N	C	P
N	N/N	N/C	N/P
C	C/N	C/C	C/P
P	P/N	P/C	P/P

Table 2. Results for PSO using the constriction coefficient for type-1

Experiment No.	Iterations	Swarm	Coefficient Constriction	Average error	Runtime
1	50	50	1.0	0.0606	00:20:05
2	100	200	1.0	0.2670	02:44:53
3	100	100	1.0	0.0301	01:49:55
4	100	150	1.0	0.0315	01:54:31
5	100	50	1.0	0.0266	03:16:34
6	100	57	1.0	0.0211	00:41:38
7	100	300	1.0	0.0276	04:04:54
8	100	80	1.0	0.0527	01:03:45
9	150	80	1.0	0.0260	01:35:17
10	300	150	1.0	0.0307	04:30:16
11	500	200	1.0	0.0529	39:56:23
12	200	90	1.0	0.0345	01:54:59
13	150	100	1.0	0.0496	01:36:37
14	100	53	1.0	0.0230	00:39:29

4 Results of the Simulation

This section presents the results of proposed controller to stabilize the autonomous mobile robot. The evaluation is done through computer simulation done in MATLAB ® and Simulink ® 2007b.

Table 2 shows the results of the FLC, obtained by varying the values of maximum iterations and the number of particles, where the highlighted row shows the best result obtained with the method. Figure 2 shows the behavior of the optimization method.

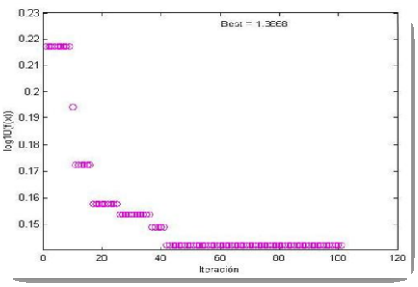


Fig. 2. Evolution of PSO for the optimization of FLC

Figure 3 shows the membership functions of the FLC obtained by the PSO algorithm, and achieved the desired path and the degree of error was obtained.

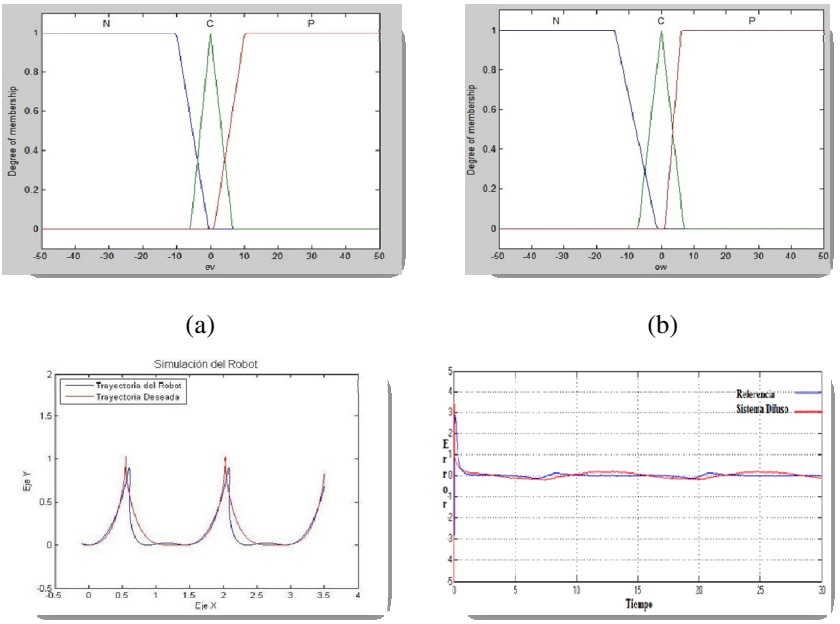


Fig. 3. (a) Linear velocity (ev), (b) Angular velocity (ew), (c) shows the desired path and the trajectory obtained, (d) plots representing the degree of error in the simulation

As we can see, the above results are acceptable for type-1 FLC obtaining a final result of 0.0211 using 57 particles and 100 iterations in a time of 47 minutes and 38 seconds for this experiment, but as previously mentioned were also simulations with type-2, taking into account the same parameters and conditions of Takagi-Sugeno controller, the results of these simulations are presented below in Table 3 [7].

Table 3. Results of PSO using the constriction coefficient for type-2

Experiment No.	Iterations	Swarm	Coefficient Constriction	Average error	Runtime
1	100	150	1.0	0.0659	02:20:31
2	100	200	1.0	0.0675	02:56:21
3	100	250	1.0	0.0666	03:05:31
4	200	150	1.0	0.0663	03:55:06
5	200	200	1.0	0.0651	04:07:14
6	200	250	1.0	0.0642	04:20:10
7	200	300	1.0	0.0536	04:54:43
8	200	350	1.0	0.0554	05:12:09
9	250	350	1.0	0.0600	05:38:20
10	300	300	1.0	0.0531	06:12:57
11	300	350	1.0	0.0503	07:30:28
12	300	380	1.0	0.0500	07:55:08
13	350	400	1.0	0.0501	08:10:15
14	350	450	1.0	0.0503	08:59:01
15	400	300	1.0	0.0502	12:31:11

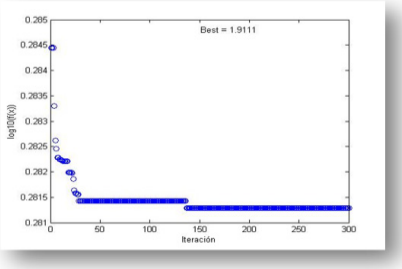


Fig. 4. Plot of convergence of the PSO algorithm

Figure 5 shows the membership functions of the FLC obtained by the PSO algorithm, and achieved the desired path and the degree of error was obtained.

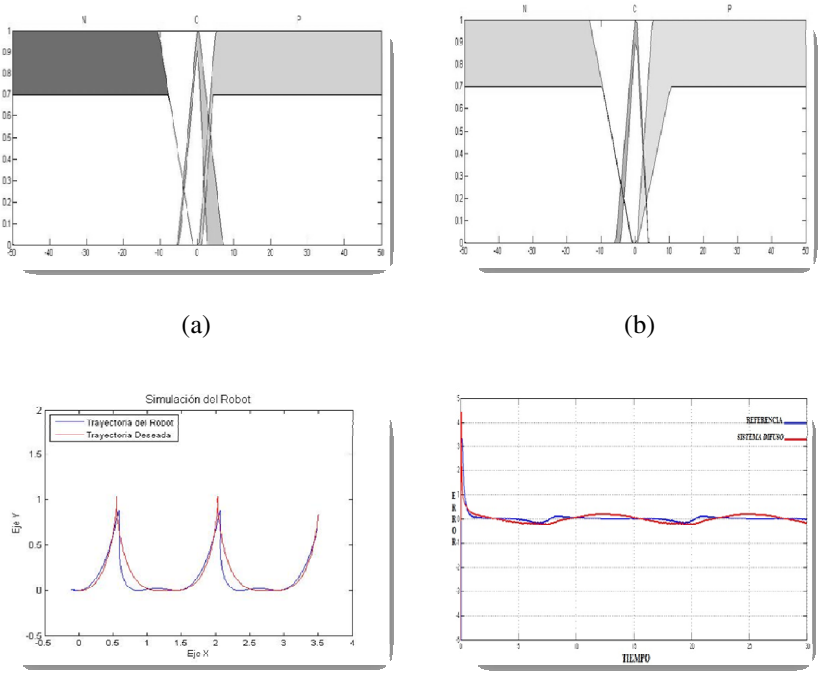


Fig. 5. (a) Linear velocity (*ev*), (b) Angular velocity (*ew*), (c) shows the desired path and the trajectory obtained, (d) plots representing the degree of error in the simulation

The above results are acceptable for type-2 FLC to obtain a final result of 0.0500 using 380 particles and 300 iterations in a time of 7 hours 55 minutes and 08 seconds for this experiment.

5 Conclusions

With the results of the experiments shown in table 4, we can determine that for this particular problem was much better optimized with fuzzy logic controller type-1.

The trajectory tracking controller is designed based on the dynamics and kinematics of mobile autonomous robot through the application of PSO for the optimization of membership functions of fuzzy controller both type-1 and type-2 with the good results obtained after simulations.

Table 4. Comparison of results of PSO algorithm type-1 and type-2

	Iterations	Swarm	Average error	Run-time
PSO with type-1 fuzzy logic	100	57	0.0211	00:41:38
PSO with type-2 fuzzy logic	300	380	0.0500	07:55:08

References

[1] Wales, J., Sanger, L.: Enciclopedia libre, Wikipedia. [Fecha de consulta: 17 Enero 2011], Disponible en, http://es.wikipedia.org/wiki/NP-hard

[2] Eberhart, R.C., Kennedy, J.: A new optimizer using particle swarm theory. In: Proc. 6th Int. Symp. Micromachine Human Sci., Nagoya, Japan, pp. 39–43 (1995)

[3] Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: Proc. IEEE Int. Conf. Neural Networks, pp. 1942–1948 (1995)

[4] Engelbrecht, A.: Fundamentals of Computational Swarm Intelligence, pp. 65–70. Wiley, England (2005)

[5] Martínez, R., Castillo, O., Aguilar, L.: Intelligent control for a perturbed autonomous wheeled mobile robot using type-2 fuzzy logic and genetic algorithms. JAMRIS 2(1), 1–11 (2008)

[6] Astudillo, L., Castillo, O., Aguilar, L.T., Martínez, R.: Hybrid Control for an Autonomous Wheeled Mobile Robot Under Perturbed Torques. In: Melin, P., Castillo, O., Aguilar, L.T., Kacprzyk, J., Pedrycz, W. (eds.) IFSA 2007. LNCS (LNAI), vol. 4529, pp. 594–603. Springer, Heidelberg (2007)

[7] Martínez, R., Castillo, O., Aguilar, L.: Optimization of interval type-2 fuzzy logic controllers for a perturbed autonomous wheeled mobile robot using genetic algorithms. Information Sciences 179, 2158–2174 (2009)

Part II

Hybrid Intelligent Systems for Pattern Recognition and Time Series Prediction

Multi-Objective Hierarchical Genetic Algorithm for Modular Neural Network Optimization Using a Granular Approach

Daniela Sánchez and Patricia Melin

Tijuana Institute of Technology
danielasanchez.itt@hotmail.com, pmelin@tectijuana.mx

Abstract. In this paper we propose a multi-objective hierarchical genetic algorithm (MOHGA) for modular neural network optimization. A granular approach is used due to the fact that the dataset is divided into granules or sub modules. The main objective of this method is to know the optimal number of sub modules or granules, but also allow the optimization of the number of hidden layers, number of neurons per hidden layer, error goal and learning algorithms per module. The proposed MOHGA is based on the Micro genetic algorithm and was tested for a pattern recognition application. Simulation results show that the proposed modular neural network approach offers advantages over existing neural network models.

1 Introduction

Hybrid intelligent systems are computational systems that integrate different intelligent techniques. Examples of these techniques are modular neural networks (MNN) and genetic algorithms (GA). Hybrid intelligent systems are now being used to support complex problem solving and decision making in a wide variety of tasks. Hybrid intelligent systems allow the representation and manipulation of different types and forms of data and knowledge, which may come from various sources. In this paper these techniques are combined using a granular approach. It was decided to apply the proposed method to pattern recognition to test the approach with complex problems.

There are many works that combine different techniques and they have demonstrated that the integration of different intelligent techniques provide good results, such as in [11][15][16][17][18][19][20][23].

This paper is organized as follows: Section 2 contains the basic concepts used in this research work, section 3 contains the general architecture of the proposed method, section 4 presents experimental results and in section 5, the conclusions of this work are presented.

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 107–120.
springerlink.com

2 Basic Concepts

In this section we present a brief overview of the basic concepts used in this research work.

2.1 Modular Neural Networks

Neural networks (NNs) can be used to extract patterns and detect trends that are too complex to be noticed by either humans or other computer techniques [24]. The modular neural networks (MNNs) are comprised of modules. The idea on which this kind of learning structure is based on the divide-and-conquer paradigm: the problem should be divided into smaller sub problems that are solved by experts (modules) and their partial solutions should be integrated to produce a final solution [3][13][24]. A module can be a sub-structure or a learning subprocedure of the whole network [2].

The results of the different applications involving Modular Neural Networks (MNNs) lead to the general evidence that the use of modular neural networks implies a significant learning improvement comparatively to a single NN and especially to the backpropagation NN. Each neural network works independently in its own domain. Each of the neural networks is build and trained for a specific task [14].

2.2 Multi-Objective Hierarchical Genetic Algorithm

A Genetic algorithm (GA) is an optimization and search technique based on the principles of genetics and natural selection [10][21][25]. GAs are nondeterministic methods that employ crossover and mutation operators for deriving offspring. GAs work by maintaining a constant-sized population of candidate solutions known as individuals (chromosomes) [7][12][22].

Introduced in [26], a Hierarchical genetic algorithm (HGA) is a type of genetic algorithm. Its structure is more flexible than the conventional GA. The basic idea under hierarchical genetic algorithm is that for some complex systems, which cannot be easily represented, this type of GA can be a better choice. The complicated chromosomes may provide a good new way to solve the problem [27][28].

Multi-objective optimization (MO) seeks to optimize the components of a vector-valued cost function. Unlike single objective optimization, the solution to this problem is not a single point, but a family of points known as the Pareto-optimal set. Each point in this surface is optimal in the sense that no improvement can be achieved in one cost vector component that does not lead to degradation in at least one of the remaining components [9].

There are three general approaches to multi-objective optimization. The first is to combine the individual objective functions into a single composite function (Aggregating functions). The second is to use Population-based approaches and

the third is to use Pareto-based approaches. A Pareto optimal set is a set of solutions that are non dominated with respect to each other. Pareto optimal sets can be of varied sizes, but the size of the Pareto set increases with the increase in the number of objectives [1].

In this work the multi-objective genetic algorithm is based on a Micro genetic algorithm, proposed in [5][6]. Two main characteristics of this kind of genetic algorithm are that it works with a small population and has a re initialization process.

2.3 Granular Computing

Granular computing is often defined as an umbrella term to cover many theories, methodologies, techniques, and tools that make use of granules in complex problem solving. Granular computing is a new term for the problem solving paradigm and may be viewed more on a philosophical rather than technical level [29][30][31][32].

Granular computing has begun to play important roles in bioinformatics, e-Business, security, machine learning, data mining, high-performance computing and wireless mobile computing in terms of efficiency, effectiveness, robustness and uncertainty [4][34][35].

A granule may be interpreted as one of the numerous small particles forming a larger unit. The philosophy of thinking in terms of levels of granularity, and its implementation in more concrete models, would result in disciplined procedures that help to avoid errors and to save time for solving a wide range of complex problems. At least three basic properties of granules are needed: internal properties reflecting the interaction of elements inside a granule, external properties revealing its interaction with other granules and, contextual properties showing the relative existence of a granule in a particular environment [33].

3 General Architecture of the Proposed Method

The proposed method combines modular neural networks (MNN) and fuzzy logic as response integrators. In particular, it can be used for pattern recognition. This proposed method is able to use some data sets, for example to use "N" biometric measures to identify someone and the data of each biometric measure would be divided into different numbers of sub modules. The general architecture of the proposed method is shown in Figure 1. For joining the different responses of each biometric measure fuzzy integration is used. The proposed method also performs the optimization of the modular neural networks (as number of layers, goal error, number of neurons, etc.) and the different parameters of the fuzzy integrator.

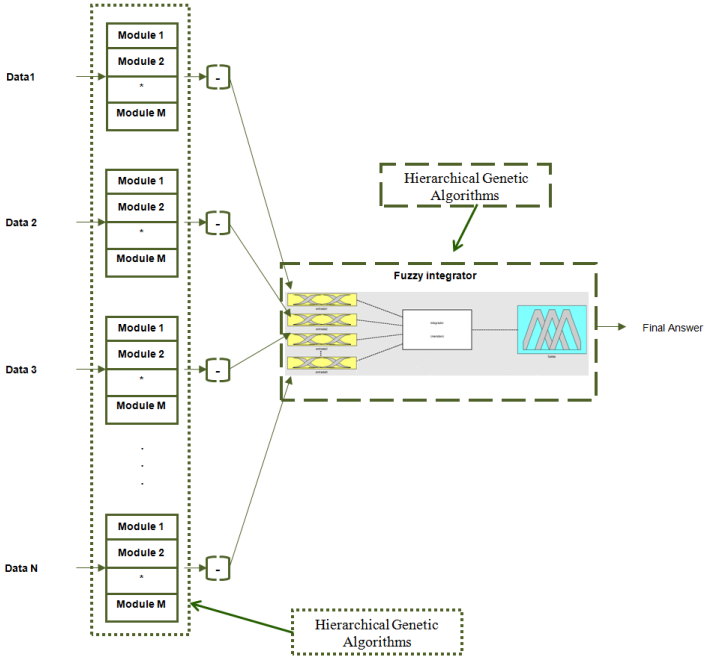


Fig. 1. The general architecture of the proposed method

3.1 General Architecture of the Proposed Method for the Modular Neural Network

The proposed method for MNN consists in changing the number of modules and the data per module, for example in the case of human recognition, it means that there will be different number of persons in each sub module. The number of sub modules can be established by a genetic algorithm, but at this moment the number is established randomly. The architecture of the proposed method for the modular neural network is shown in Figure 2.

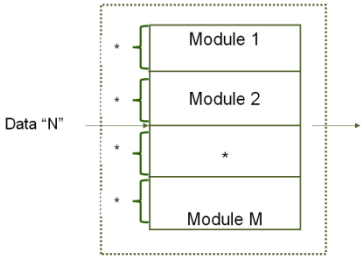


Fig. 2. The architecture of proposed method for the modular neural network

This method also chooses randomly which images will be used for training, but first the percentage of images for training is established (at this moment that percentage is defined randomly).

3.2 *Description of the Multi-Objective Hierarchical Genetic Algorithm for MNN Optimization*

With the purpose of knowing the optimal number of modules and the percentage of data for training, it is proposed the use of a genetic algorithm that allows the optimization of these parameters and others as the number of hidden layers, number of neurons per hidden layer, error goal and learning algorithms per module.

Figure 3 shows the chromosome, which was proposed for optimization of the neural networks.

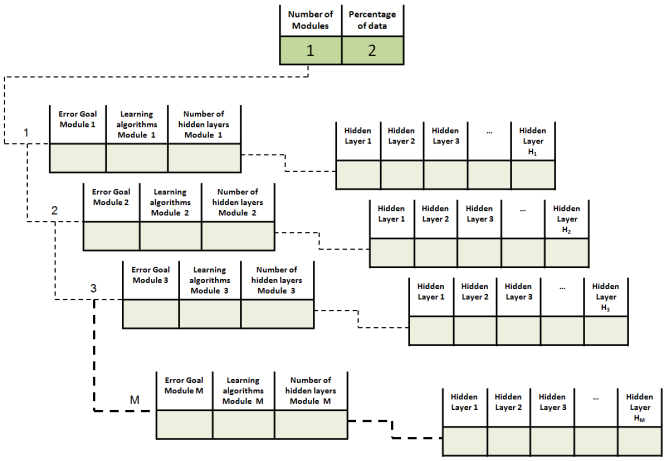


Fig. 3. The chromosome of the multi-objective hierarchical genetic algorithm for the MNN

The way in which the multi-objective hierarchical genetic algorithm works is illustrated in Figure 4 and described in more detail below.

First, a random population is generated. This random population is divided in two parts: a non-replaceable and replaceable portion. The non-replaceable portion never changes during the evolution, this helps to provide diversity. The replaceable portion experiences changes after certain condition is satisfied, this condition is called nominal convergence.

The working population at the beginning is taken (with a certain probability) from both portions of the main population. During each cycle, the MOHGA uses conventional genetic operators.

The external memory is initially empty, in each cycle the non-dominated vectors found are saved in that memory, logically a comparison is performed between the new vectors found and vectors already stored.

The MOHGA has two kinds of convergence. The first is the usually used (for example when it has the maximum number of cycle or generations, or when the value desired of one objective function is obtained). The second is called Nominal Convergence, in this case is established each 5 generations, here two non dominated vectors are taken of the external memory and these are compared with two vectors of the Replaceable portion, if the two vectors taken of the replaceable portion are dominated by the others, those vector are replaceable for the two vectors of the external memory, then the working population is reinitialized.

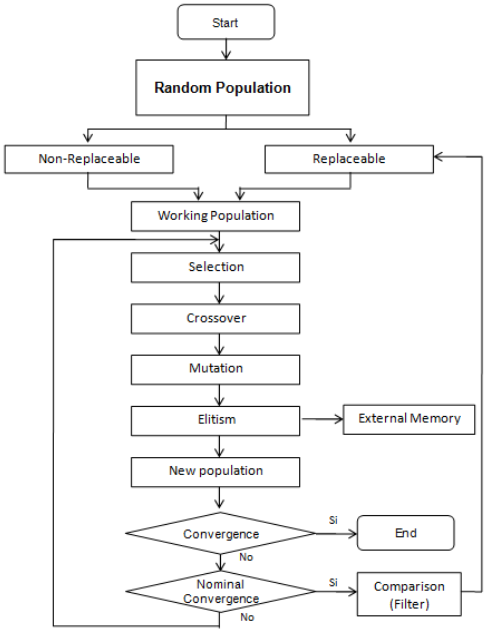


Fig. 4. Diagram that illustrates the way in which the multi-objective hierarchical genetic algorithm works

3.3 Objective Functions

In order to not only get the network that provides us with the lowest error of recognition another objective function is set, and so not only obtain the best network with the lowest error of recognition , but also obtain a modular neural network that uses the lowest percentage of data for the training phase. The objective functions are defined below:

$$\text{Min } f_1 = \sum_{i=1}^{m}\left(\left(\sum_{j=1}^{n_m} X_j\right)/n_m\right) \tag{1}$$

$$\text{Min } f_2 = \textbf{percentage of data} \tag{2}$$

3.4 Databases

The databases used in this work are described below in more detail.

3.4.1 Ear Database

We used a database of the University of Science and Technology of Beijing [8]. The database consists of 77 people, which contain 4 images per person (one ear), the image dimensions are 300 x 400 pixels, the format is BMP.

The persons are students and teachers from the department of Information Engineering. Two images with angle variation and one with illumination variation are used. Figure 5 shows an example of the pre-processing applied to each image in the ear.

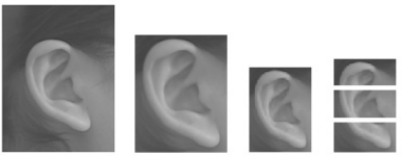


Fig. 5. Sample pre-processing done to the images of ear

3.4.2 Voice Database

In the case of voice, the database consists of 10 voice samples (of 77 persons), WAV format. The persons are students from the Tijuana Institute of Technology. The word that they said in Spanish was "ACCESAR". To preprocess the voice the Mel Frequency Cepstral Coefficients were used.

4 Experimental Results

In this section the results obtained in this work are presented. It was decided to use the database already described above. For the integration of responses the winner takes all method was used.

4.1 Non Optimized

In this test the images percentage and the images, which would be used for training, were established randomly. The non optimized results of the modular neural network are shown below.

4.1.1 Non Optimized Results of Ear

The best 5 results for the ear are shown in Table 1. In this test, it can be noticed that when the number of data per module is varied the rate of recognition varies.

It can be noticed that in the training # 4, that when the images 2, 3, and 4 are used a rate of recognition of 100% is obtained.

Table 1. The best results for the ear (Non Optimized)

Training	Images for training	Persons per module	Recognition Rate
1	(1,3 and 4)	Module # 1 (1 to 6) Module # 2 (7 to 14) Module # 3 (15 to 77)	67.53% (52/77)
2	(2 and 4)	Module # 1 (1 to 38) Module # 2 (39 to 70) Module # 3 (71 to 77)	77.92% (120/154)
3	(1 and3)	Module # 1 (1 to 9) Module # 2 (10 to 44) Module # 3 (45 to 77)	83.11% (128/154)
4	(2, 3 and 4)	Module # 1 (1 to 40) Module # 2 (41 to 50) Module # 3 (51 to 77)	100% (77/77)
5	(2 and 3)	Module # 1 (1 to 23) Module # 2 (24 to 47) Module # 3 (48 to 77)	93.50% (144/154)

4.1.2 Non Optimized Results of Voice

The best 5 results for the voice are shown in Table 2. In this test, we can notice that when the number of data per module is varied the rate of recognition varies. It can be noticed that in the training # 3, that when the voices 1, 3, 5, 7, 8 and 10 are used, a rate of recognition of 96.75% using 8 sub modules is obtained.

Table 2. The best results for voice (Non Optimized)

Training	Voices for training	Persons per module	Recognition Rate
1	53% (1,3,6,7 and 9)	Module # 1 (1 to 22) Module # 2 (23 to 57) Module # 3 (58 to 77)	278/385 72.20%
2	48% (1,2,5,6 and 7)	Module # 1 (1 to 39) Module # 2 (40 to 68) Module # 3 (69 to 77)	260/385 67.53%
3	35% (2,5,8 and 9)	Module # 1 (1 to 36) Module # 2 (37 to 68) Module # 3 (69 to 77)	401/462 86.79%
4	46% (3,5,6,7 and 10)	Module # 1 (1 to 40) Module # 2 (41 to 67) Module # 3 (68 to 77)	347/385 90.12%
5	59% (1,3,5,7,8 and 10)	Module # 1 (1 to 7) Module # 2 (8 to 39) Module # 3 (40 to 77)	298/308 96.75%

4.2 Optimized Results

These tests make use of the multi-objective hierarchical genetic algorithm, this MOHGA allows the optimization of parameters of the modular neural network, such as number of sub modules, percentage of data for training, number of hidden layers, number of neurons per hidden layer, error goal and learning algorithms per module.

4.2.1 Optimized Results of the Ear

The main parameters used in this evolution are shown in Table 3 and the Pareto optimal set found for the ear are shown in Figure 6.

Table 3. Main parameters of the MOHGA

Memory Size	Non-Replaceable Memory	Replaceable Memory	Working Memory	Pareto Optimal	Duration
50	25	25	5	7	12:48:08

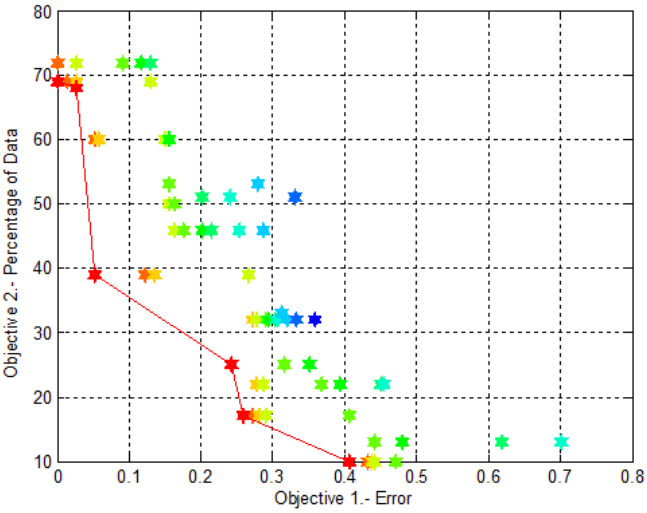


Fig. 6. Pareto optimal set for the evolution of the ear

The solutions found in the Pareto optimal set are shown in Table 4, and the best architecture is shown in Table 5.

Table 4. The best results for the ear (Pareto optimal set)

Solution	Num. of Modules	% Of data	Total rec.	Error
1	5	69%	100%	0
2	6	68%	97.40%	0.0260
3	5	39%	94.80%	0.0519
4	5	25%	75.75%	0.2424
5	9	17%	74.02%	0.2597
6	9	17%	74.02%	0.2597
7	5	10%	59.30%	0.4069

The solutions that have a recognition rate greater than 97% are taken, and of the resulting set, the solution with lower percentage of data is the best for us.

Table 5. The best result of the ear (Optimized)

Num. of Mod.	% and images	Num. Hidden layers and Num. of neurons	Persons per module	Rec. Rate	Error
5	69% (2,3 and 4)	*3(173,135,44)* *2(153,120)* *4(72,184,96,116)* *2(197,166)* *3(164,22,94)*	Module # 1 (1 to 6) Module # 2 (7 to 13) Module # 3 (14 to 27) Module # 4 (28 to 53) Module # 5(54 to 77)	77/77 100%	0

4.2.2 Optimized Results of the Voice

The main parameters used in this evolution are shown in Table 6 and the Pareto Optimal set found for the voice are shown in Figure 7.

Table 6. Main parameters of the MOHGA

Memory Size	Non-Replaceable Memory	Replaceable Memory	Working Memory	Pareto Optimal	Duration
50	25	25	5	9	01:51:12

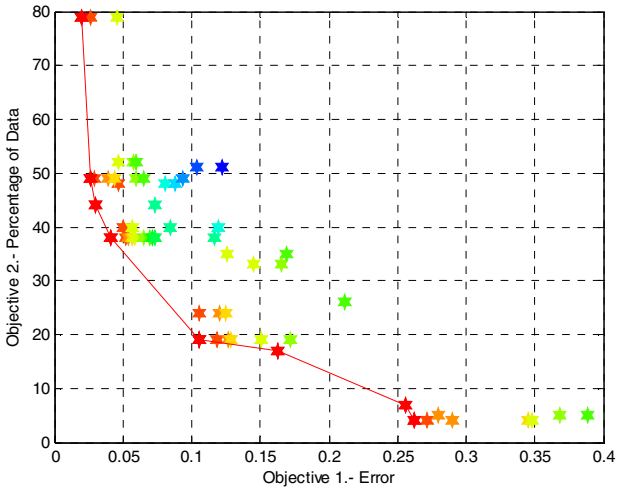


Fig. 7. Pareto optimal set of the evolution of voice

The solutions found in the Pareto optimal set are shown in Table 7, and the best architecture is shown in Table 8.

Table 7. The best results for voice (Pareto Optimal)

Solution	Num. of Modules	% Of data	Total rec.	Error
1	5	79%	98.05%	0.0195
2	9	49%	97.40%	0.0260
3	9	44%	96.96%	0.0303
4	5	38%	95.88%	0.0411
5	10	19%	89.44%	0.1055
6	10	19%	89.44%	0.1055
7	7	17%	83.76%	0.1623
8	7	7%	74.45%	0.2554
9	6	4%	73.73%	0.2626

The solutions that have a recognition rate greater than 97% are taken, and of the resulting set, the solution with lower percentage of data is the best for us.

Table 8. The best result of the voice (Optimized)

Num. of Mod.	% and voices	Num. Hidden layers and Num. of neurons	Persons per module	Rec. Rate	Error
9	49% (1,3,8,9 and 10)	*4 (57,144,128,83)* *4 (156,189,158,193)* *5(123,105,169,110, 105)* *1(89)* *3(78,143,62)* *2(101,38)* *4(22,60,91,173)* *4(81,128,139,118)* *4(145,28,187,32)*	Module # 1(1 to 14) Module # 2(15 to 35) Module # 3(36 to 46) Module # 4(47 to 50) Module # 5(51 to 53) Module # 6(54 to 55) Module # 7(56 to 64) Module # 8(65 to 72) Module # 9(73 to 77)	375/385 97.40%	0.0260

5 Conclusions

A new method for combining modular neural networks with a granular approach was proposed. The main goal of this work was providing the modular neural networks with the following characteristics: allow changing the number of modules, data per module, and percentage of data for training, all of that with the goal of obtaining a better rate of recognition.

A multi-objective hierarchical genetic algorithm was developed for optimization of some parameters of this model of modular neural networks, those parameters are the number of modules, percentage of data for training, goal error per module, number of hidden layers per module and their respective neurons. This MOHGA is able to obtain the best modular neural network with the lowest error of recognition and that uses the lowest percentage of data for the training phase.

In this work when the tests with the ear are compared, a significant difference does not exist, because the database has few images per person, but if we compare the non optimized versus the optimized test in the case of the voice, we can notice that with less data a good recognition rate is obtained.

References

1. Abraham, A., Jain, L., Goldberg, R.: Evolutionary Multiobjective Optimization, Softcover reprint of hardcover 1st ed. Springer (2005)
2. Auda, G., Kamel, M.S.: Modular Neural Networks A Survey. Int. J. Neural Syst. 9(2), 129–151 (1999)
3. Azamm, F.: Biologically Inspired Modular Neural Networks. PhD thesis. Virginia Polytechnic Institute and State University, Blacksburg, Virginia (2000)
4. Bargiela, A., Pedrycz, W.: The roots of granular computing. In: IEEE International Conference on Granular Computing (GrC), pp. 806–809 (2006)

5. Coello Coello, C.A., Lamont, G.B., van Veldhuizen, D.A.: Evolutionary Algorithms for Solving Multi-Objective Problems, 2nd edn. Springer (2007)
6. Coello Coello, C.A., Toscano Pulido, G.: A Micro-Genetic Algorithm for Multiobjective Optimization. In: Zitzler, E., Deb, K., Thiele, L., Coello Coello, C.A., Corne, D.W. (eds.) EMO 2001. LNCS, vol. 1993, pp. 126–140. Springer, Heidelberg (2001)
7. Coley, A.: An Introduction to Genetic Algorithms for Scientists and Engineers. Wspc, Har/Dskt edition (1999)
8. Database Ear Recognition Laboratory from the University of Science & Technology Beijing (USTB), http://www.ustb.edu.cn/resb/en/index.htmlasp (accessed September 21, 2009)
9. Fonseca, C.M., Fleming, P.J.: Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization. In: ICGA 1993, pp. 416–423 (1993)
10. Haupt, R., Haupt, S.: Practical Genetic Algorithms, 2nd edn., pp. 42–43. Wiley Interscience (2004)
11. Hidalgo, D., Castillo, O., Melin, P.: Type-1 and type-2 fuzzy inference systems as integration methods in modular neural networks for multimodal biometry and its optimization with genetic algorithms. Inf. Sci. 179(13), 2123–2145 (2009)
12. Huang, J., Wechsler, H.: Eye Location Using Genetic Algorithm. In: Second International Conference on Audio and Video-Based Biometric Person Authentication, pp. 130–135 (1999)
13. Khan, A., Bandopadhyaya, T., Sharma, S.: Classification of Stocks Using Self Organizing Map. International Journal of Soft Computing Applications 4, 19–24 (2009)
14. Melin, P., Castillo, O.: Hybrid Intelligent Systems for Pattern Recognition Using Soft Computing: An Evolutionary Approach for Neural Networks and Fuzzy Systems, 1st edn., pp. 119–122. Springer (2005)
15. Melin, P., Kacprzyk, J., Pedrycz, W.: Bio-inspired Hybrid Intelligent Systems for Image Analysis and Pattern Recognition. Springer (2009)
16. Melin, P., Mendoza, O., Castillo, O.: An improved method for edge detection based on interval type-2 fuzzy logic. Expert Syst. Appl. 37(12), 8527–8535 (2010)
17. Melin, P., Mendoza, O., Castillo, O.: Face Recognition with an Improved Interval Type-2 Fuzzy Logic Sugeno Integral and Modular Neural Networks. IEEE Transactions on Systems, Man, and Cybernetics, Part A 41(5), 1001–1012 (2011)
18. Melin, P., Sánchez, D., Castillo, O.: Genetic optimization of modular neural networks with fuzzy response integration for human recognition. Information Sciences 197, 1–19 (2012)
19. Mendoza, O., Melin, P., Castillo, O.: Interval type-2 fuzzy logic and modular neural networks for face recognition applications. Appl. Soft Comput. 9(4), 1377–1387 (2009)
20. Mendoza, O., Melin, P., Licea, G.: A hybrid approach for image recognition combining type-2 fuzzy logic, modular neural networks and the Sugeno integral. Inf. Sci. 179(13), 2078–2101 (2009)
21. Mitchell, M.: An Introduction to Genetic Algorithms, 3rd edn. A Bradford Book (1998)
22. Nawa, N., Takeshi, F., Hashiyama, T., Uchikawa, Y.: A study on the discovery of relevant fuzzy rules using pseudobacterial genetic algorithm. IEEE Transactions on Industrial Electronics 46(6), 1080–1089 (1999)

23. Sánchez, D., Melin, P.: Modular Neural Network with Fuzzy Integration and Its Optimization Using Genetic Algorithms for Human Recognition Based on Iris, Ear and Voice Biometrics. In: Soft Computing for Recognition Based on Biometrics, 1st edn. SCI, pp. 85–102. Springer (2010)
24. Santos, J.M., Alexandre, L.A., Marques de Sá, J.: Modular Neural Network Task Decomposition Via Entropic Clustering. ISDA (1), 62–67 (2006)
25. Segovia, J., Szczepaniak, P.S., Niedzwiedzinski, M.: E-Commerce and Intelligent Methods, 1st edn., p. 181. Physica-Verlag (2002)
26. Tang, K.S., Man, K.F., Kwong, S., Liu, Z.F.: Minimal Fuzzy Memberships and Rule Using Hierarchical Genetic Algorithms. IEEE Trans. Ind. Electron. 45(1), 162–169 (1998)
27. Wang, C., Soh, Y.C., Wang, H., Wang, H.: A Hierarchical Genetic Algorithm for Path Planning in a Static Environment with Obstacles. In: Canadian Conference on Electrical and Computer Engineering, IEEE CCECE 2002, vol. 3, pp. 1652–1657 (2002)
28. Worapradya, K., Pratishthananda, S.: Fuzzy supervisory PI controller using hierarchical genetic algorithms. In: 5th Asian Control Conference, vol. 3, pp. 1523–1528 (2004)
29. Yao, J.T.: A ten-year review of granular computing. In: Proceedings of the 3rd IEEE International Conference on Granular Computing (GrC), pp. 734–739 (2007)
30. Yao, J.T.: Information granulation and granular relationships. In: Proceedings of 2005 IEEE Conference on Granular Computing (GrC), pp. 326–329 (2005)
31. Yao, Y.Y.: A Partition Model of Granular Computing. In: Peters, J.F., Skowron, A., Grzymała-Busse, J.W., Kostek, B.z., Świniarski, R.W., Szczuka, M.S. (eds.) Transactions on Rough Sets I. LNCS, vol. 3100, pp. 232–253. Springer, Heidelberg (2004)
32. Yao, Y.Y.: Granular computing: basic issues and possible solutions. In: Proceedings of the 5th Joint Conferences on Information Sciences, pp. 186–189 (2000)
33. Yao, Y.Y.: On Modeling Data Mining with Granular Computing. In: 25th International Computer Software and Applications Conference (COMPSAC), pp. 638–649 (2001)
34. Yao, Y.Y.: Perspectives of granular computing. In: IEEE International Conference on Granular Computing (GrC), pp. 85–90 (2005)
35. Yu, F., Pedrycz, W.: The design of fuzzy information granules: Tradeoffs between specificity and experimental evidence. Applied Soft Computing 9(1), 264–27 (2009)

An Analysis on the Intrinsic Implementation of the Principle of Justifiable Granularity in Clustering Algorithms

Mauricio A. Sanchez[1], Oscar Castillo[2], and Juan R. Castro[1]

[1] Universidad Autonoma de Baja California, Tijuana, Mexico
{mauricio.sanchez,jrcastro}@uabc.edu.mx
[2] Tijuana Institute of Technology, Tijuana, Mexico
ocastillo@tectijuana.mx

Abstract. The initial process for the granulation of information is the clustering of data, once the relationships between this data have been found these become clusters, each cluster represents a coarse granule, whereas each data point represents a fine granule. All clustering algorithms find these relationships by different means, yet the notion of the principle of justifiable granularity is not considered by any of them, since it is a recent idea in the area of Granular Computing. This paper describes a first approach in the analysis of the relationship between the size of the clusters found and their intrinsic implementation of the principle of justifiable granularity. An analysis is done with two datasets, simplefit and iris, and two clustering algorithms, subtractive and granular gravitational.

1 Introduction

Granular computing is an area which has been gaining support since its initial conception[1],[2],[3]. Focusing on how information is treated and represented, it describes how information should efficiently relate to each other, defining the size of each granule and confining the cardinality of data into a meaningful information granule[3]. This area has expanded into different interpretations, since it is more of a theory in general than a defined methodology of treating information, yet they all share the same objective, to obtain meaningful granules. Information granules can also be represented in a number of forms, fuzzy logic[4], rough sets[5], etc.

The process of obtaining information granules is first preceded by the action of finding relations between all data; this process is usually done by a clustering algorithm. Clustering algorithms are defined as algorithms which find relationships between data, there are multiple methodologies in which such relationships are found, there are categorized as centroid based[6], density based[7], hierarchical based[8], among others. Each one of these obtaining similar results, yet at the same time, finding different results, this is, and there is a difference in performance on each type of algorithm. The end result of such algorithms are usually

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 121–134.
springerlink.com

cluster centers as well as areas of influence, in the specifics of centroid based clustering algorithms, they find cluster centers between the universe of data and radial areas of influence, which can be easily mapped into fuzzy Gaussian membership functions[9]. This paper focuses on this type of clustering algorithms.

Although clustering algorithms obtain acceptable results in the relationships found[10], they do so in a manner that does not take into account if they adhere to the basic theory of granular computing or not, since many algorithms precede the existence of the area of granular computing. Yet clustering algorithms and granular computing are intertwined in such a way that you cannot remove one from the other, because finding relationships in data is essential to obtaining information granules.

One step in the correct direction of uniting clustering algorithms and granular computing is the implementation of the principle of justifiable granularity[11]; this principle is a first attempt to describe, in more detail, how an information granule is in fact meaningful, and not redundant or too specific. Most, if not all, clustering algorithms do not take into account if they create meaningful granules or not, they only concentrate on the end result, and not if the chosen granules are optimized and/or meaningful.

This paper is organized into multiple chapters which introduce concepts on clustering algorithms, describe de concept of the principle of justifiable granularity and finally a discussion is given into how some clustering algorithms intrinsically implement such principle.

2 Clustering Algorithms

Clustering algorithms have the main objective of finding hidden relationships between the data inside of a specific information universe. In the following sections two clustering algorithms will be described, only until how granules are found, since some continue onto the process of creating a fuzzy system from the clusters which are found, this is to focus on the discussion and analysis on how the intrinsically impellent the principle of justifiable granularity.

2.1 Subtractive Algorithm

This algorithm is density based, which means that its end results are calculated by analyzing the density of data points inside a given radius, this is done iteratively point by point until an objective function is within a specified tolerance[12].

The following, describes the main calculations done by this algorithm:

1. A measure of the potential (1) of each data point is first calculated, taking into account the value of the given radius (2).

$$P_i = \sum_{j=1}^{n} e^{-\alpha \|x_i - x_j\|^2} \tag{1}$$

$$\alpha = \frac{4}{r_a^2} \tag{2}$$

2. The highest potential is selected (2) and accordingly reduces the potential on the rest of the point, calculated with the support of the given radius (4)

$$P_i \Leftarrow P_i - P_{1^e}^{*\,-\beta\|x_i - x_1\|^2} \tag{3}$$

$$\beta = \frac{4}{r_b^2} \tag{4}$$

3. This is repeated until the finalization condition (5) is met

$$P_k^* < \varepsilon P_1^* \tag{5}$$

4. All sigmas, or radial area of influence, are then calculated (6)

$$sigmas_i = \frac{r\big(\max(X) - \min(X)\big)}{\sqrt{8}} \tag{6}$$

Due to the nature of this algorithm, and the need to know the cardinality of each found cluster in respect to the information universe, as to apply the principle of justifiable granularity, a manner to find such cardinality (7) was implemented into the algorithm, this was done by calculating the distances between each found cluster and all data points, and the closest data points to each clusters were added to their respected cardinality (8), where the distance is calculated with the Euclidean distance measurement (9).

$$card\{c_i \in X_j\} \tag{7}$$

$$X_j = \text{data points closer to } \mathrm{c_i} \tag{8}$$

$$d_i = \sqrt{c_i - x_j};\quad i! = j \tag{9}$$

2.2 Granular Gravitational Algorithm

This is a hybrid centroid-density based algorithm, meaning all the calculations are done based on point density and point distance, this is considering that Newton´s Law of Universal Gravitation[13] is utilized to carry out the main cluster calculations[14].

The following describes the main calculations done by this algorithm:

1. All gravitational forces (10) in the system are calculated

$$F_{ij} = G\frac{m_i m_j}{d(x_i, x_j)^2} \tag{10}$$

2. The sum force (11) for all data points is then sorted in descending order

$$sumF = \sum_{n=0}^{i} F_j \tag{11}$$

3. All points with strong gravitational force and within a given radius are joined (12), joining the point with lesser mass to the point with more mass

$$x_i \cup x_j\,; \quad m_i \cup m_j \tag{12}$$

4. This is repeated until a balance in the gravitational forces in the system is achieved
5. All sigmas are then calculated based on the strongest force exerted by any found cluster unto the rest of the information universe

3 Principle of Justifiable Granularity

This principle is "concerned with the formation of a meaningful information granule based on available experimental evidence"[Witold,2011], meaning that finding clusters and assigning sigmas that give acceptable results is no longer relevant, and the correct size and cardinality of each cluster is.

3.1 Basic Theory

The principle of justifiable granularity is concerned with obtaining the adequate size of each information granule which was found for that specific information universe. In this theory, there are two main rules that must be followed:

1. The numeric evidence of a specific information granule must be as high as possible, this means that the cardinality of information within a granule must be high
2. The information granule should be as specific as possible, meaning that vagueness is reduced, relying only on a very strict cardinality

This completely contradicts itself, as shown in Fig.1, since in one side you must elevate the cardinality yet on the other side you must reduce it. This is not a problem in the final implementation, since a balance is found which conforms to both rules.

Fig. 1. Visual difference between both rules: a) High numerical evidence with low specificity, and b) Low numerical evidence with high specificity

Considering that the size of the granule is to be found, this size is separated into two segments, segment **a** and segment **b**, as shown in Fig 2, in other words, the left side of the granule and the right side of the granule, since the Median of the data points, *Med(D)*, may not always be in the center and the data points may not necessarily be normalized in a symmetric position, the lengths of both **a** and **b** will most of the time not be the same.

Fig. 2. Granule showing the difference between the lengths of a and b, both starting at the Median of the data points

The cardinality of each information granule (1) is important, since it is the basis for finding the lengths required to obtain a meaningful granule.

$$card\{x_k \in \Omega\} \tag{13}$$

This cardinality is separated into two distinct sets, which conform to the specific calculation of the length for both a and b. Whereas the cardinality for calculating b (14) would only be defined by the data points larger than Med(D).

$$card\{x_K \in \Omega, x_k > Med(D)\} \tag{14}$$

Considering the contradicting nature of the two rules of the principle of justifiable granularity, to obtain the best balance between both, an optimization must be done to find the best possible solution. This is obtained by maximizing an objective function (15), with reference to a user criterion α (16) which controls the balance between both rules. Varying this user criterion affects how specific, or general, a granule can become.

$$V(b^*) = max_{b>Med(D)}[\mathrm{V(b)}] \tag{15}$$

$$\alpha \in [0, \alpha_{max}] \tag{16}$$

This optimization is done with an integration of the probability density function from *Med(D)* to all prototypes of b, multiplied by a scaling factor which integrates the user criterion for specificity (17).

$$V(b) = e^{(-\alpha|b-Med(D)|)} \int_{Med(D)}^{b} p(x)\,\mathrm{d}x \tag{17}$$

3.2 The Specificity Criterion

As briefly described in the previous section, this α criterion controls how specific or general the information granule will become, while still maintaining a balance between both rules, which, in nature, contradict each other.

As a simple implementation it is easy to start the value at 0 to represent the most general possible granule, yet it is somewhat trivial to know the maximum value α can take as to have a very specific granule.

A form to find an approximate optimum maximum value for the criterion is best described in Fig. 3, where its behavior is of an inverse natural logarithm plot. α_{max}, is an approximate value at best, since after a threshold is surpassed, no matter the increment of value, how it affects the scaling factor of the length of the information granule is negligent.

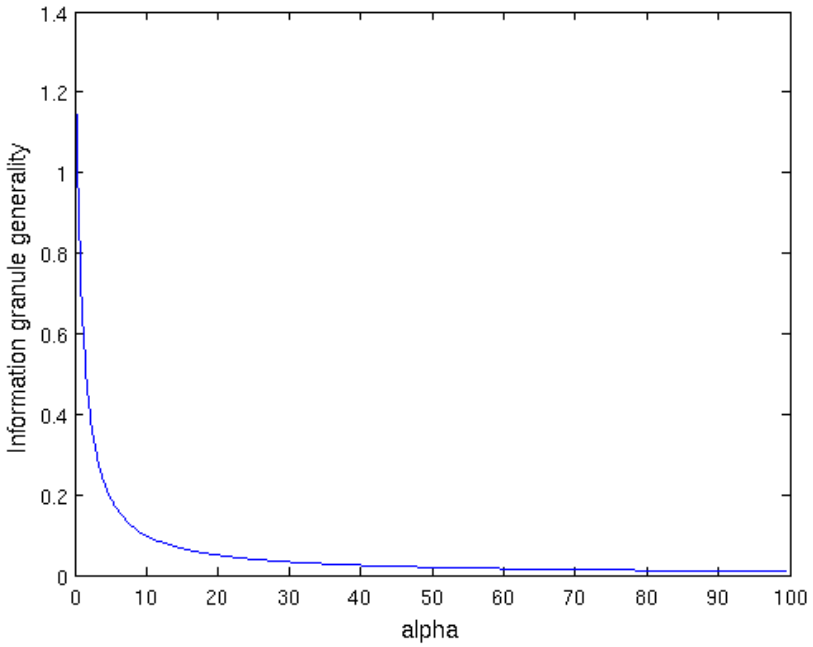


Fig. 3. Behavior of α in respect to the generality of the information granule

The approximate value of α_{max} is somewhere in the area where the behavior plot stops to descend fast and starts to stabilize, as shown in Fig. 4.

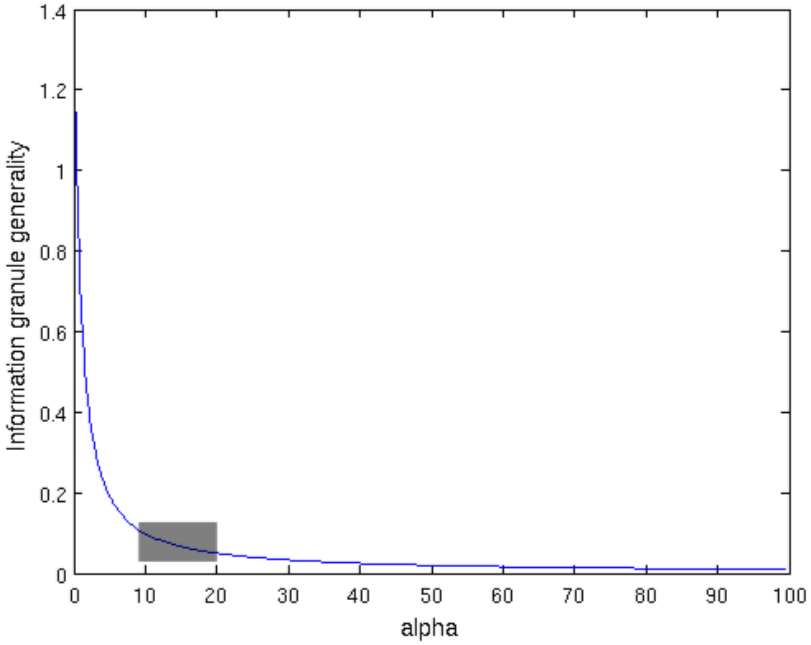


Fig. 4. Interval where the approximate value of α_{max} is found

A heuristic is offered which can find this approximate value of α_{max}, which is described as the natural logarithm of the cardinality of the chosen side (18) to find its length divided by the length of the closest point (x_1) to *Med(D)* (19).

$$\partial = count(card\{c_i \in \Omega > Med(D)\}) \tag{18}$$

$$\alpha_{max} = \frac{\log(\partial)}{|Med(D) - x_1|} \tag{19}$$

On other terms of application, and considering that choosing an α close to 0 will scale the information granule to a non specific granule, with a length near 0, the middle point between [0, α_{max}] or $\alpha_{max}/2$, will not necessarily impose a length of exactly half the size, since the behavior, as already shown, is non-linear, and it will greatly vary from the numerical evidence contained within each specific information granule.

4 Algorithm Analysis

The analysis which will be shown is of a comparison on the variable level, which is how the principle of justifiable granularity is applied. Comparing the length of each sigma in respect to the lengths found by the principle of justifiable granularity,

considering the user criterion which varies the end results, they will be of different values, only to demonstrate how they vary with each specific cardinality. Another consideration in the results is that both lengths a and b should have different values for their respective α, yet the same value was chosen to demonstrate another difference in how the lengths are different even if on the same granule, yet diversified depending on the cardinality of the left or right side of the *Med(D)*.

Originally, the value of *Med(D)* should be calculated depending on the data contained within each information granule, but since the comparison is done using existing clustering algorithms, they already obtain cluster centers, which in this case, represent the value of *Med(D)*, and for the sake of a fair comparison, without altering the results given by these algorithms, *Med(D)* will not be calculated and will be obtained directly from these algorithms, taken from their found centers.

The sigmas which are obtained by these algorithms will be compared to the calculated lengths by the principle of justifiable granularity.

For a more equal analysis of both algorithms, their results which will be analyzed will be done using their obtained clusters when they identify 100% the datasets.

4.1 Simplefit Dataset

This dataset is a very simple data fitting benchmark with one input and one output, this dataset can be seen in Fig. 5.

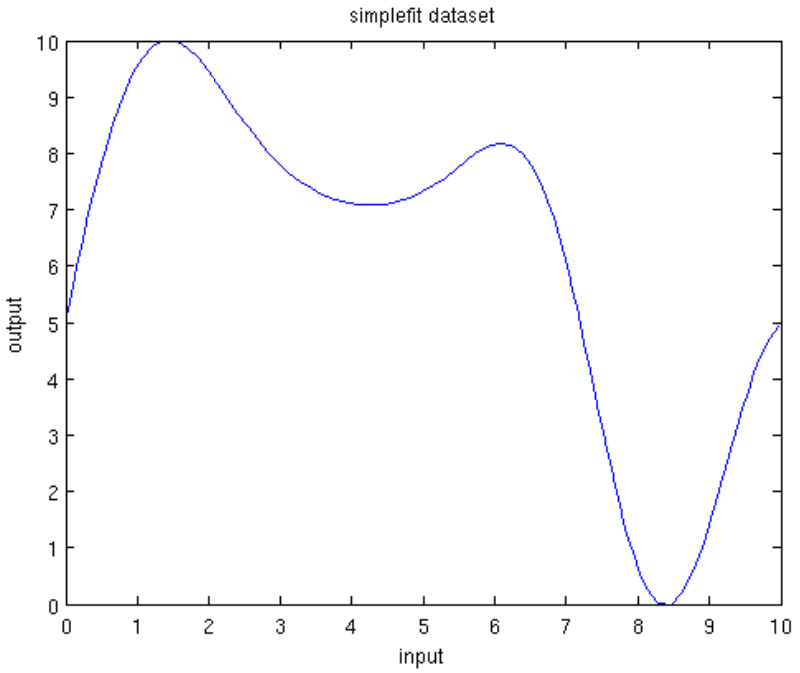


Fig. 5. Visual representation of the simplefit dataset with one input and one output

Both algorithms which are being analyzed will be compared utilizing $\alpha=0$ on both lengths of the granule, a and b. Fig. 6 shows the subtractive algorithm´s sigmas, represented by straight lines, against the lengths found by the principle of justifiable in this case, the granule is none-specific and completely generalized, yet in comparison with the sigmas from the subtractive algorithm, half reach far beyond the cardinality of each granule.

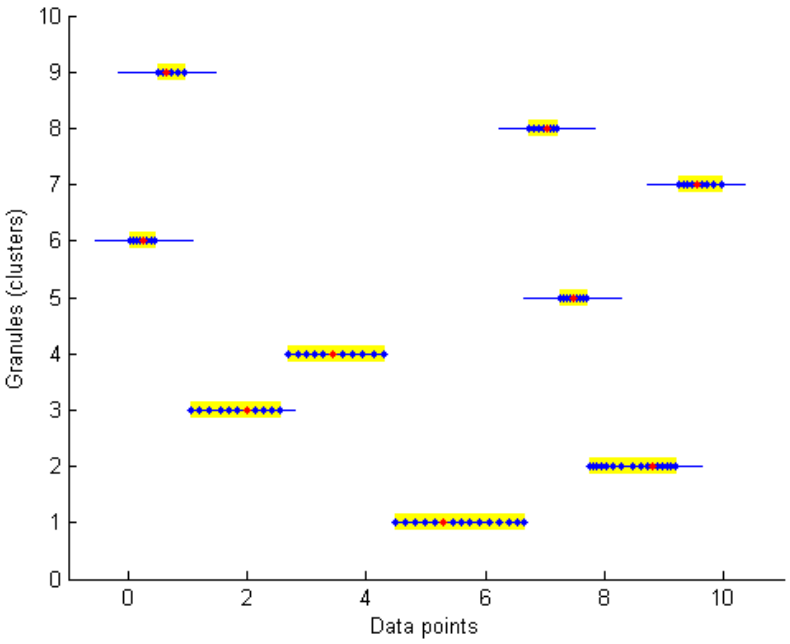


Fig. 6. Results for subtractive algorithm of the simple fit dataset under, comparing the sigmas (lines) found by the algorithm against the lengths suggested by the principle of justifiable granularity (shaded area).

The granular gravitational algorithm, shown in Fig. 7, shows a very different behavior in its sigmas, compared to the subtractive algorithm´s sigmas, since they adapt much more to the cardinality of each granule, but in comparison with the lengths given by the principle of justifiable granularity, they are still very generalized and reach beyond the its cardinality in the same way the subtractive algorithm does.

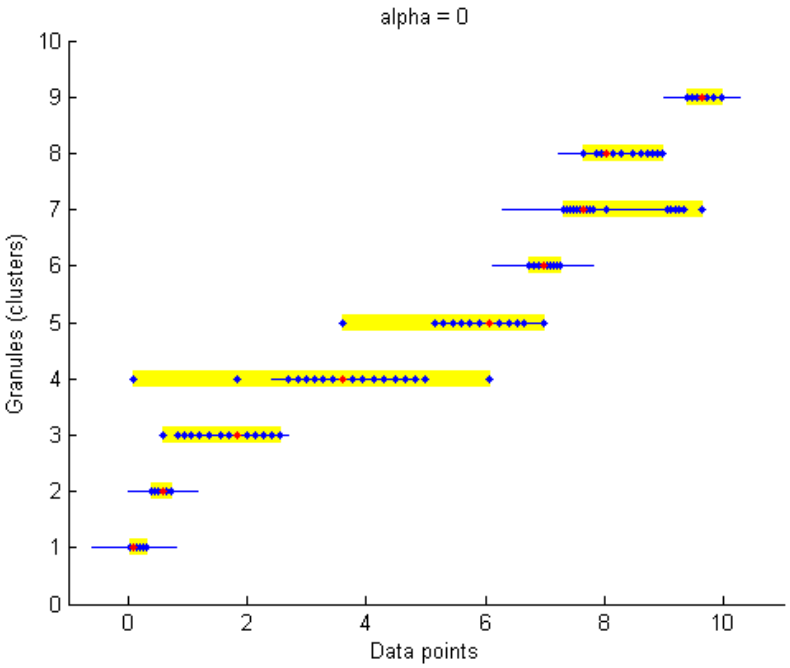


Fig. 7. Results for granular gravitational algorithm of the simple fit dataset under, comparing the sigmas (lines) found by the algorithm against the lengths suggested by the principle of justifiable granularity (shaded area)

4.2 Iris Dataset

The iris dataset is a benchmark dataset in remarks to clustering and classification, with four inputs and three classes, and a non-linear solution, this makes for a very interesting dataset to test.

First to be analyzed, is the comparison for the subtractive algorithm, specifically for the input variable of the petal length, comparing the sigmas obtained by the clustering algorithm against the lengths obtained by applying the principle of justifiable granularity with an $\alpha=5$. As shown in Fig. 8, the calculated sigmas by the subtractive algorithm still overreach, in most cases, far beyond the cardinality of each cluster, yet in some cases some data points are not inside that area of influence. For the lengths found by the principle of justifiable granularity, since the criterions was chosen as a higher number as to suggest more specificity and less numerical evidence, they reach far less that the subtractive algorithm´s sigmas. This level of adaptation in the length of each granule is not possible in the subtractive algorithm.

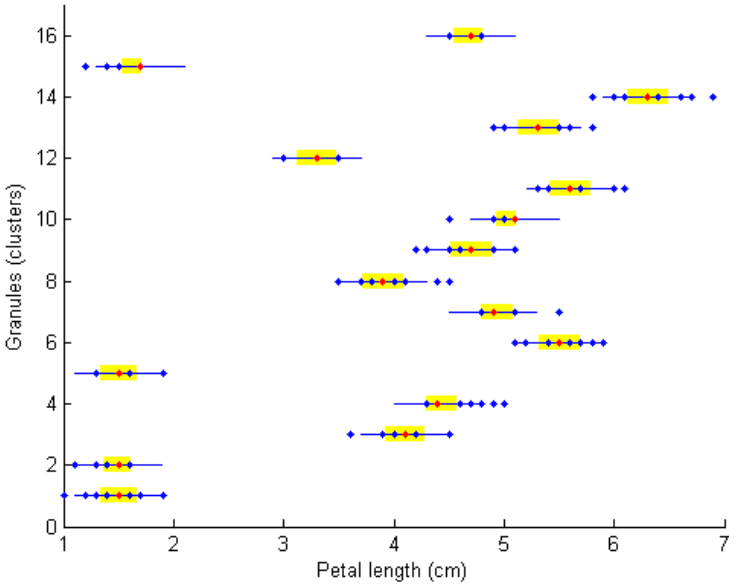


Fig. 8. Clusters found by the subtractive algorithm for the variable ´petal length´, comparing the sigmas found by the algorithm (straight lines), against the lengths found by the principle of justifiable granularity (shaded areas)

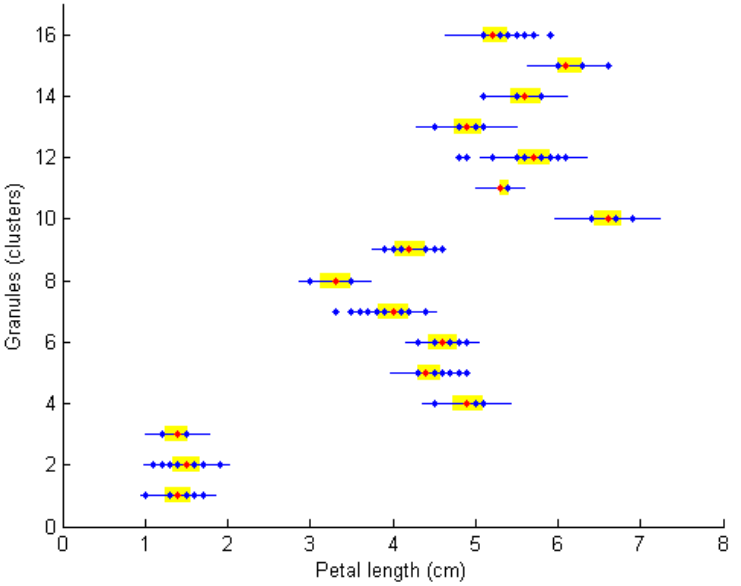


Fig. 9. Clusters found by the granular gravitational algorithm for the variable ´petal length´, comparing the sigmas found by the algorithm (straight lines), against the lengths found by the principle of justifiable granularity (shaded areas)

As for the granular gravitational algorithm´s results, the same variable was chosen with the same value for the specificity criterion. As shown in Fig. 9, the sigmas found by the clustering algorithm adapt more to the cardinality, yet this adaptation is not very noticeable. Comparing with the lengths found by applying the principle of justifiable granularity, the length of the granules are much smaller, caused by choosing a specificity criterion which in turn limits the numerical evidence which supports the information granule.

In the next section, a much more detailed analysis regarding the cardinality, sigmas and results obtained by applying of the principle of justifiable granularity will be given.

4.3 Discussion

First of all, comparing both algorithm´s general performance, they obtain exactly the same number of clusters when they 100% identify these datasets, meaning their performance is very similar. The main difference in their results is that the subtractive algorithm´s sigmas are constraint to the same length of each cluster on the variable level, and the granular gravitational algorithm adapts its sigmas to global cardinality, meaning that the clusters at the variable level will not always be optimal.

Discussing now the intrinsic application of the principle of justifiable granularity in both algorithms, we first analyze the results obtained the simple fit dataset. Since this comparison was done with a specificity criterion of 0, a full coverage of the lengths were expected, the subtractive algorithm had mixed results in this case, since some of its clusters were perfectly represented yet other clusters were very over-represented, having their length reach far beyond the cardinality; and the granular gravitational, in this case, did adapt more the cardinality of each cluster, in some cases ignoring isolated data points inside its own cardinality, yet in other cases its reach went far beyond its data limits.

Analyzing these results, we can assume that, in general, the subtractive algorithm obtains sigmas that are more specific in nature and the granular gravitational algorithm obtains sigmas that are less specific. Even with these differences, they both have very mixed results in the specificity length of their sigmas.

Directing attention to the iris dataset, these results show a different facet of the justifiability of granules, since the criterion which was chosen is 5, meaning that its granules are not very specific nor very general, in fact, since α_{max} was not considered in this case, it is unknown if the value of 5 is the exact balance between specificity and numerical evidence of each granule, and considering that the cardinality of each granule affects the non-linear behavior of α_{max}, this value was simply chosen to demonstrate a higher value of specificity as to show how it affects the length of the granule.

Comparing both results for the variable of the petal length on algorithms in contrast to the length obtained after applying the principle of justifiable granularity, we can see that in both cases, the lengths of the sigmas obtained by the algorithms are similar in size, yet the granular gravitational sigmas adapt more to the cardinality of each cluster. The calculated length with a criterion of 5, in both

cases, are similar in size for most clusters, and considering that both algorithms have a similar performance, we can assume that in this case, this similarity in obtained lengths by applying the principle of justifiable granularity is to be expected.

Reducing this discussion to fewer words, both algorithms have a very similar general performance with these datasets. The subtractive algorithm finds sigmas of the same length on the variable level and has mixed results with the specificity of each found cluster, while the granular gravitational algorithm finds sigmas that adapt more to the cardinality of each granule, but since this is done globally on each data point, it also has mixed results in respect to the cardinality of each cluster on the variable level, and its sigmas tend to be less specific that the sigmas found by the subtractive algorithm. And as already stated, both algorithms do not adapt nor consider how specific a granule should be.

5 Conclusion

5.1 Conclusions

In general, this analysis has given some insight into how, and possibly a why, these clustering algorithms obtain such acceptable results. Another finding is that they both obtain sigmas which are normally too generalized in respect to the cardinality of each information granule, resulting in the question of how would the final evaluation of identifications of the datasets would be affected if the lengths of each information granule was reduced according to the principle of justifiable granularity.

Both algorithms have mixed results when it comes to the intrinsic implementation of the principle of justifiable granulation, yet in general, they both find sigmas that are less specific that the most generalized possible length which can be found by applying this principle.

Considering the non-linear behavior of the specificity criterion, and how there is a fuzzy interval where its optimum value is found, an approximate heuristic has been proposed which finds such value.

5.2 Future Work

Regarding the principle of justifiable granularity, a global specificity criterion could make it much easier to implement this theory, since having to choose this value for two lengths, for every granule, and for every variable, is a non-realistic implementation.

Having obtained acceptable results with the application of the principle of justifiable granularity, integrating this into current clustering algorithms would greatly aid in the advancement of the general theory of granular computing, since the granules would now be meaningful and more specific to the needs of the problem to be solved.

More clustering algorithms can be tested in the same way the subtractive and granular gravitational algorithms were, as to measure more algorithms and asses their performance and creation of meaningful information granules.

References

1. Pedrycz, W.: Granular Computing - The Emerging Paradigm. Journal of Uncertain Systems 1, 38–61 (2007)
2. Castillo, O., Melin, P., Pedrycz, W.: Design of interval type-2 fuzzy models through optimal granularity allocation. Applied Soft Computing 11, 5590–5601 (2011)
3. Bargiela, A., Pedrycz, W.: Toward a Theory of Granular Computing for Human-Centered Information Processing. IEEE Transactions on Fuzzy Systems 16, 320–330 (2008)
4. Zadeh, L.A.: Fuzzy Sets. Information and Control 8, 338–353 (1965)
5. Pawlak, Z.: Rough sets. International Journal of Computer & Information Sciences 11, 341–356 (1982)
6. Linda, O., Manic, M.: General Type-2 Fuzzy C-Means Algorithm for Uncertain Fuzzy Clustering. IEEE Transactions on Fuzzy Systems, 1 (2012)
7. Ester, M., Kriegel, H.-P., Sander, J., Xu, X.: A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. Computer, 226–231 (1996)
8. Sibson, R.: SLINK: An optimally efficient algorithm for the single-link cluster method. The Computer Journal 16, 30–34 (1973)
9. Nock, R., Nielsen, F.: On weighting clustering. IEEE Transactions on Pattern Analysis and Machine Intelligence 28, 1223–1235 (2006)
10. Zaït, M., Messatfa, H.: A comparative study of clustering methods. Future Generation Computer Systems 13, 149–159 (1997)
11. Pedrycz, W.: The Principle of Justifiable Granularity and an Optimization of Information Granularity Allocation as Fundamentals of Granular Computing. Journal of Information Processing Systems 7, 397–412 (2011)
12. Chiu, S.L.: Fuzzy model identification based on cluster estimation. Journal of Intelligent and Fuzzy Systems 2, 267–278 (1994)
13. Newton, I.: Philosophiae Naturalis Principia Mathematica (1687)
14. Sanchez, M.A., Castillo, O., Castro, J.R., Rodríguez-Díaz, A.: Fuzzy granular gravitational clustering algorithm. North American Fuzzy Information Processing Society (2012)

Mental Tasks Temporal Classification Using an Architecture Based on ANFIS and Recurrent Neural Networks

Emmanuel Morales-Flores[1], Juan Manuel Ramírez-Cortés[1], Pilar Gómez-Gil[2], and Vicente Alarcón-Aquino[3]

[1] Department of Electronics, Instituto Nacional de Astrofísica, Óptica y Electrónica, Tonantzintla, Puebla, Mexico
[2] Department of Computer Science, Instituto Nacional de Astrofísica, Óptica y Electrónica, Tonantzintla, Puebla, Mexico
[3] Department of Electronics, University of the Americas, Cholula, Puebla, Mexico

Abstract. In this paper, an architecture based on adaptive neuro-fuzzy inference systems (ANFIS) assembled to recurrent neural networks, applied to the problem of mental tasks temporal classification, is proposed. The electroencephalographic signals (EEG) are pre-processed through band-pass filtering in order to separate the set of energy signals in alpha and beta bands. The energy in each band is represented by fuzzy sets obtained through an ANFIS system, and the temporal sequence corresponding to the combination to be detected, associated to the specific mental task, is entered into a recurrent neural networks. This experiment has been carried out in the context of brain-computer-interface (BCI) systems development. Experimentation using EEG signals corresponding to mental tasks exercises, obtained from a database available to the international community for research purposes, is reported. Two recurrent neural networks are used for comparison purposes: Elman network and a fully connected recurrent neural network (FCRNN) trained by RTRL-EKF (real time recurrent learning – extended Kalman filter). A classification rate of 88.12% in average was obtained through the FCRNN during the generalization stage.

1 Introduction

Brain Computer Interfaces (BCIs) are systems aiming to establish communication pathways between the brain and computerized mechanisms in order to perform some control actions. There are three main stages which can be distinguished in a BCI system: detection of the neural signals from the brain, an algorithm for decoding these signals, and a methodology for mapping decoded signals into some predefined activities. In recent years, there has been a growing interest in the research community on signal processing techniques oriented to solve the multiple challenges involved in BCI applications [1-3]. An important motivation to develop BCI systems, among some others, would be to allow an individual with motor disabilities to have control over specialized devices such as computers, speech

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 135–146.
springerlink.com

synthesizers, assistive appliances or neural prostheses. A dramatic relevance arises when thinking about patients with severe motor disabilities such as locked-in syndrome, which can be caused by amyotrophic lateral sclerosis, high-level spinal cord injury or brain stem stroke. BCIs would increase an individual's independence, leading to an improved quality of life and reduced social costs. Among the possible brain monitoring methods for BCI purposes, the EEG constitutes a suitable alternative because of its good time resolution, relative simplicity and noninvasiveness when compared to other methods such as functional magnetic resonance imaging (fMRI), positron emission tomography (PET), magnetoencephalography or electrocorticogram systems.

Current research on BCI systems distinguishes seven main categories according to the neurological mechanisms or processes involved: sensorimotor activity [4,5], P300 [6,7], visual evoked potentials [8,9], slow cortical potentials [10], activity of neural cell and response to mental tasks [11,], as well as multiple neuromechanisms, which use a combination of two or more of the previous (see [2] for a review). Each category constitutes a paradigm which can be used in order to develop a BCI system. Among these, mental task-based BCI systems have captured the attention of the research community, in part due to their independence of additional interfaces such as the screen of alphanumeric characters used in VEP, or the arrows and symbols used in motor imagery experiments, as well as the relative flexibility of the user to carry out some mental tasks at his/her own will. Several feature extraction methods for mental task-based BCI design have been reported, most of them based on parametric, such as autoregressive or adaptive models [12], non-parametric models based on several schemes of spectral analysis such as Wavelet transform or Stockwell transform [13,14], or fuzzy sets [15]. In this sense, it has been shown that information contained in spectral bands α (8-13 Hz), β (14-20 Hz), γ (24-37 Hz), or even in higher frequencies [16], can be used to detect neural activity directly related to specific mental tasks. Time-frequency analysis can be carried out using different approaches such as Wavelet analysis [7,13], filter bank [17], empirical mode decomposition [18] and others. Those approaches reflect only the estimated power across a range of frequencies. In a number of reported works, non-linear classifiers such as neural network and support vector machine algorithms are used [19]. Recently, there has been several studies oriented to capture temporal behavior through predictive schemes and recurrent neural networks with good results, which encourage further research in that direction [20-22]. In this work we present a temporal classification approach on a two-state mental task experiment applying for comparison purposes two recurrent neural networks: Elman and Fully Connected Recurrent Neural Network (FCRNN). The proposed scheme performs the feature extraction based on an Adaptive Neurofuzzy Inference System (ANFIS), previous to the temporal classification stage.

The rest of the paper is organized as follows: Section 2 describes theory associated to recurrent neural networks. Section 3 describes the proposed methodology on temporal classification of the mental task experiment. Section 4 describes and analyzes the obtained results. Section 5 presents some concluding remarks, perspectives, and future direction of this research oriented to the implementation of a BCI system.

2 Neural Network Classifiers

Temporal classification refers to the assignation of a class based on features obtained in different time periods. It is fairly common to use recurrent neural networks for this and other types of temporal information processing [23]. Furthermore, it is important to point out that some recurrent neural architectures are able to model chaos [24] and it has been proved that the dynamics in an EEG is chaotic (for example see [25]). In the results presented here, we used two types of recurrent neural networks to build a temporal classifier: a Simple Recurrent Network (SRN), also known as "Elman network" [26] and a fully connected recurrent neural network (FCRNN) similar to the one described by [27]. SRN was trained using the algorithm "Back Propagation through time" (BPTT) [28] and FCRNN was trained with the algorithm "Real Time Recurrent Learning – Extended Kalman filter" (RTRL-EKF) [29,30]. These architectures and algorithms are briefly described next.

2.1 Simple Recurrent Network

Time can be represented in several ways in recurrent neural networks. In a SRN, time is implicitly represented using a context layer. This model was introduced by Elman [26], which, in spite of being rather simple, is able to memorize previous states of a sequence. SRN architecture has 4 layers: an input layer, a hidden layer, an output layer and a context layer. (see Fig. 1). SRN differs from a feed-forward multi-layer perceptron (MLP) by the fact that SRN can build a representation of past events. This is achieved because nodes in the context layer memorize the outputs of nodes in hidden layer coming from a previous time. This context layer is able to create a map of some temporal properties of the system. SRN is a special case of a state-space model of a recurrent neural network. According to [30] the dynamic of a state-space model can be described by the following equations:

$$\mathbf{x}_{n+1} = \mathbf{a}(\mathbf{x}_n, \mathbf{u}_n), \tag{1}$$

$$\mathbf{y}_n = \mathbf{B}\mathbf{x}_\mathbf{n}, \tag{2}$$

where $\mathbf{y}_n$ represents the output of the system, $\mathbf{u}_n = \{u_n, u_{n-1} \ldots u_{n-q+1}\}$ is a vector of the exogenous inputs in different steps , and $\mathbf{x}_\mathbf{n}$ is the output of a bank of q unit-time delays, q being the number of nodes in the input layer. $\mathbf{a}(\cdot,\cdot)$ is a non-linear function characterizing the hidden layer and $\mathbf{B}$ is the matrix of synaptic weights characterizing the output layer. The hidden layer is non-linear and output layer is linear. For the case of a SRN, the output layer may also be non-linear. SRN may be trained in different ways. We used a gradient descent back propagation algorithm with adaptive learning rate, provided by the neural network toolbox of Matlab V6.0. For the results presented in this paper we use a hyperbolic tangent sigmoid transfer function for the nodes in hidden layer ('tansig' Matlab function) and a logarithmic sigmoid transfer function ('logsig' Matlab function) for the

output layers. Function 'calcgbtt' was used as gradient function, which calculates the bias and weight performance gradients using the back-propagation through time algorithm [28]. For a detailed explanation of BPTT see [30].

2.2 Fully-Connected Recurrent Neural Network

Several of the most useful algorithms for training recurrent neural networks are based on Kalman Filtering (KF) [27]. The Extended Kalman Filtering (EKF) is required in nonlinear systems, as in the case of recurrent neural networks, because a linearization around the current working point requires to be applied before that standard KF is performed. EKF has been applied with different styles to train recurrent neural networks, for example in [31-33]. It also has been combined with other algorithms, for example with "back truncated propagation through time" [32] and with RTRL [29].

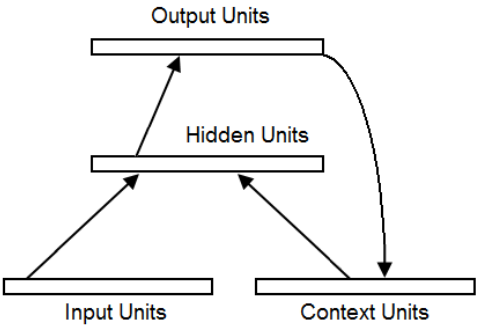


Fig. 1. The Simple Recurrent Network (Elman 1990)

The training algorithm RTRL contains two main steps (see [30,34]): gradient calculation and weights adjustments. RTRL is used to calculate the derivatives of the gradients and EKF is used for modifying the weights. According to [30], the state-space model of the network under training is defined by two models:

1) The system model, described by:

$$\mathbf{w}_{n+1} = \mathbf{w}_n + \omega_n \quad , \tag{3}$$

where $\mathbf{w}_n$ is the weight (state) vector and ω_n is a white Gaussian noise.

2) Measurement model, described by:

$$\mathbf{d}_n = \mathbf{b}(\mathbf{w}_n, \mathbf{v}_n, \mathbf{u}_n) + \mathbf{v}_n \quad , \tag{4}$$

where $\mathbf{d}_n$ is the desirable response o the system, playing the role of the "observable", $\mathbf{v}_n$ represents the recurrent node activities inside the network, $\mathbf{u}_n$ denotes the input signal to the network and $\mathbf{v}_n$ is a vector denoting measurement noise corrupting $\mathbf{d}_n$.

EKF allows the estimation of the value of the correction in the state space model, updating weights as follows:

$$\hat{\mathbf{w}}_n = \hat{\mathbf{w}}_{n-1} + \mathbf{G}_n \boldsymbol{\alpha}_n \tag{5}$$

$$\boldsymbol{\alpha}_n = \mathbf{d}_n - \mathbf{b}(\hat{\mathbf{w}}_{n-1}, \mathbf{v}_n, \mathbf{u}_n), \tag{6}$$

where $\mathbf{G}_n$ is the Kalman gain, calculated using:

$$\mathbf{G}_n = \mathbf{P}_{n-1}\mathbf{B}_n^T[\mathbf{B}_n\mathbf{P}_{n-1}\mathbf{B}_n^T + \mathbf{Q}_{v,n}]^{-1} \tag{7}$$

$$\mathbf{P}_n = \mathbf{P}_{n-1} - \mathbf{G}_n\mathbf{B}_n\mathbf{P}_{n-1} + \mathbf{Q}_{\omega,n} \tag{8}$$

$\mathbf{B}_n$ is the Jacobian of the partial derivatives with respect to the state, that is, the weights, which is calculated using RTRL algorithm. $\mathbf{Q}_{\omega,n}$ is the covariance matrix of the dynamic noise ω_n, $\mathbf{P}_n$ is the prediction error covariance matrix, and $\mathbf{Q}_{v,n}$ is the covariance matrix of the measurement noise $\mathbf{v}_n$. Calculation of partial derivatives $\mathbf{B}_n$ is defined as:

$$\mathbf{B}_n = \begin{bmatrix} \frac{\partial y_1}{\partial w_1} & \frac{\partial y_1}{\partial w_2} & \cdots \frac{\partial y_1}{\partial w_m} \\ \frac{\partial y_2}{\partial w_1} & \frac{\partial y_2}{\partial w_2} & \cdots \frac{\partial y_2}{\partial w_m} \\ \cdot & & \\ \frac{\partial y_p}{\partial w_1} & \frac{\partial y_p}{\partial w_2} & \cdots \frac{\partial y_q}{\partial w_m} \end{bmatrix}, \tag{9}$$

where q is the total number of neurons in the networks and m is the total number of weights. Using RTRL, derivatives in $\mathbf{B}_n$ are calculated as [35]:

$$\frac{\partial y_i(n+1)}{\partial w_{kl}} = \sigma'(x_i(n))\left[\sum_{j=1}^{m} w_{ij}\frac{\partial y_j(n)}{\partial w_{kl}} + \delta_{ik} z_l(n)\right] \tag{10}$$

$\sigma'(\cdot)$ is the derivative of the neuron transfer function $\sigma(\cdot)$; $x_i(n) = \sum_{j=1}^{m} w_{ij} z_j(n)$ is the input to each neuron, δ_{ik} is the Kronocker delta. For further details, see [30, 34, 35].

For the experiments showed here we used an implementation of RTRL-EKF created by [34], which is itself based on the Matlab functions created by [38]. A very good description of the data structures used in such software is given by [27].

3 Proposed Methodology

A block diagram of the proposed scheme is represented in Fig. 2. The algorithm is described as follows: preprocessing of the EEG signals obtained from P4 electrode

includes a blind source separation through Independent Component Analysis (ICA) in order to remove eye blink and other artifacts. The signal is then filtered in order to obtain the alpha and beta bands, and the power signal for each band is computed. The power signal in each band is partitioned into 5 windows with a 50% overlapping as a feature reduction process. The signal is passed through an ANFIS system in order to obtain a representation in fuzzy sets corresponding to the evolution in time of the estimated power across both spectral bands alpha and beta. Temporal sequences corresponding to the combination of energy bands for each mental task are input into a recurrent neural network, which is trained to deliver a classification decision on the corresponding mental task.

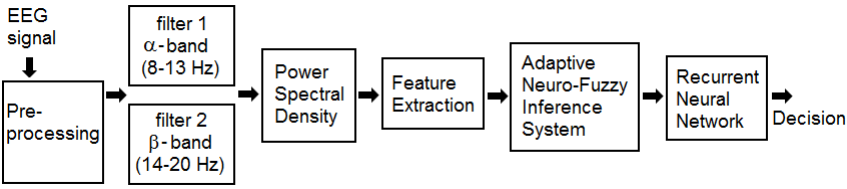


Fig. 2. Block diagram of the proposed architecture for mental tasks classification.

4 Experimental Results

EEG data were obtained previously by Keirn and Aunon [39] and are available on line for research purposes. Ten trials for each mental task resulted in a total of 20 patterns. Details of the procedure followed to detect the signals can be consulted in the cited reference. A brief description is as follows: an Electro-Cap elastic electrode cap was used to record data from positions C3, C4, P3, P4, O1, and O2 defined by the 10-20 system of electrode placement. In the original data set, there were seven subjects performing five different mental tasks and one subject performing two different mental tasks. Signals were recorded for ten seconds during the task at a sampling frequency of 250 Hertz, and each task was repeated five times per session. Subjects attended two sessions recorded on different weeks, resulting in a total of ten trials for each task. The two mental tasks are described as follows.

In the task described as *mental letter composing,* the subjects were instructed to mentally compose a letter to a friend or relative without vocalizing. The second mental task described as *visual counting,* was constructed by asking the subjects to imagine numbers being written sequentially on a blackboard, with the previous number erased before the next number was written. Experiments were executed using MATLAB version 7.6 in a personal computer with a 2.0 GHz AMD Turion processor and 3GB RAM. Figure 3 shows an example of the normalized power signal corresponding to alpha and beta bands for each mental task. According to the proposed procedure previously described, feature extraction is performed on the power signals by a window-averaging with a 50% window overlap. Fig. 4

shows an example of the feature vectors obtained through the described procedure, corresponding to the referred mental tasks. As Fig. 4 illustrates, the power representation of alpha and beta bands presents variations associated to temporal evolution of power bands following each mental task.

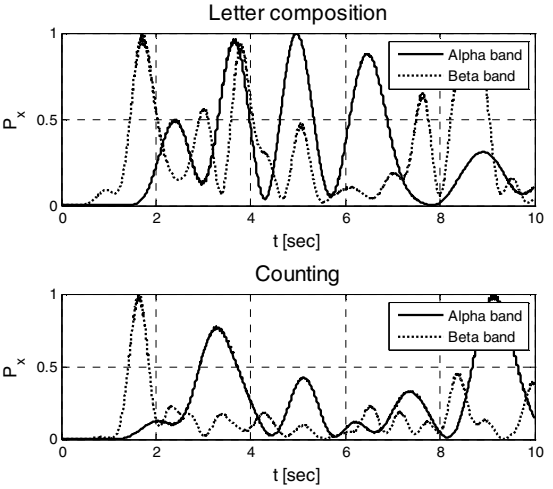


Fig. 3. Alpha and beta band power for letter composition and counting task

Since the power in bands shows variations for each subject and trial, we propose the use of an adaptive system allowing the assignment of membership functions in an automatic way in order to represent the configuration of bands through fuzzy sets, translating each experiment into a simple sequence that preserve the temporal evolution of the performed mental task.

Fig. 5 shows an example of the state assignment corresponding to the case of letter composition task. The feature extraction process is then applied to each trial in the mental tasks database, obtaining some sequences representing the state transitions of power band configurations and corresponding to each mental task. The ANFIS system was trained with the features extracted over all trials, considering an input representation with eight membership functions.

Fig. 6 shows an example of the results obtained from the ANFIS training for the two mental tasks. Temporal classification of the obtained feature vectors representing each mental task was performed using a recurrent neural network. In this paper we compare the performance of two models: a simple recurrent neural network or Elman network and a FCRNN. In both cases, the architecture of the recurrent neural networks was: 1 node in the input layer, 10 nodes in the hidden layer and 1 node in the output layer. The architecture was determined by experimentation, with the best results obtained using the described configuration.

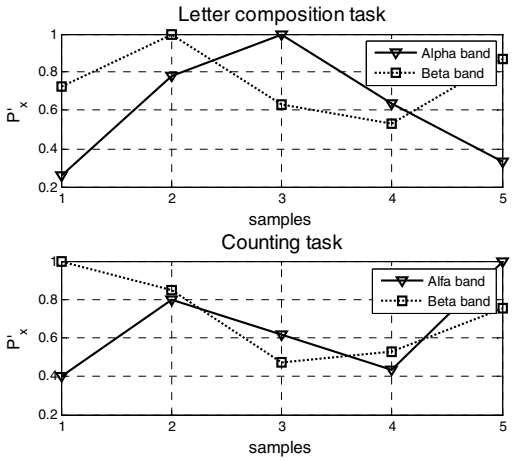


Fig. 4. Result of feature extraction process for two different mental tasks

Temporal classification results are reported based on a leave-one-out (LOO-CV) cross-validation. LOO-CV is typically used in the analysis of small datasets, where the relatively high variance of the estimator is offset by the stability resulting from the greater size of the training partition than is possible using conventional k-fold cross-validation [40]. Ten trials for each mental task result in a total of 20 patterns. The dataset was partitioned in 5 folds with 4 trials each one. LOO-CV was performed using four folds for training and the remaining one for testing. Table 1 summarizes the temporal classification results obtained in average from both, training and testing cases, with the two recurrent neural networks previously described.

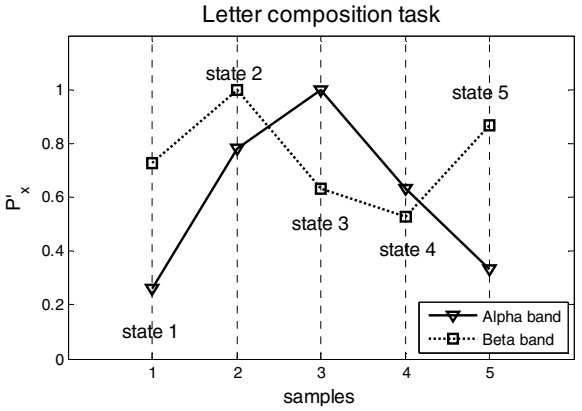


Fig. 5. State assignment for letter composition task

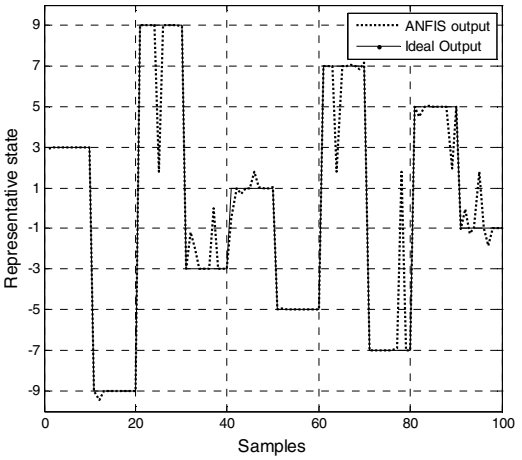


Fig. 6. Result of ANFIS training

Table 1. Results on temporal classification; training and testing

RNN	Training 500 epochs			Testing	
	MSE	Classification	Execution time	MSE	Classification
Elman	0.0328	91.75%	3'49''	0.0401	90.16%
FCRNN	0.0121	94.61%	1' 12''	0.0528	88.12%.

5 Conclusions

In this work, an architecture based on adaptive neuro-fuzzy inference systems (ANFIS) assembled to recurrent neural networks, applied to the problem of mental tasks temporal classification, has been proposed. Information on power signal obtained from Alpha and Beta bands constituted a good descriptor with an adequate separability, providing a good balance between complexity and classification rate. The feature vectors representing each mental task following a fuzzy-set paradigm, provided a good description about the temporal evolution of the power signal. A classification rate in training of 94.61 % in average was obtained through the FCRNN, with an 88.12 % of classification using leave-on-out cross validation in the testing stage. A comparison with the Elman Network indicates a better performance of the FCRNN during the training stage, with a slightly better performance of the Elman network on generalization. In both cases, an architecture of the neural network with 10 nodes in the hidden layer provided the better results. Further experimentation oriented to the construction of a database for BCI applications is currently in progress.

Acknowledgments. The first author acknowledges the financial support from the Mexican National Council for Science and Technology (CONACYT), scholarship No. 224304. This research has been partially supported by CONACYT Grant No. CB-2010-155250.

References

[1] Brunner, P., Bianchi, L., Guger, C., Cincotti, F., Schalk, G.: Current trends in hardware and software for brain–computer interfaces (BCIs). Journal of Neural Engineering 8, 025001 (2011)

[2] Bashashati, M., Fatourechi, R., Ward, K., Birch, G.E.: A survey of signal processing algorithms in brain-computer interfaces based on electrical brain signals. Journal of Neural Engineering 4(2), R32–R57 (2007)

[3] Berger, T.W., Chapin, J.K., Gerhardt, G.A., McFarland, D.J., Principe, J.C., Soussou, W.V., Taylor, D.M., Tresco, P.A.: WTEC Panel Report on International Assessment of Research and Development in Brain-Computer Interfaces, World Technology Evaluation Center, Inc. (2007), http://www.wtec.org/bci/BCI-finalreport-26Aug2008-lowres.pdf

[4] Neuper, C., Scherer, R., Wriessnegger, S., Pfurtscheller, G.: Motor imagery and action observation: Modulation of sensorimotor brain rhythms during mental control of a brain–computer interface. Clinical Neurophysiology 120(2), 239–247 (2009)

[5] Solis-Escalante, T., Muller-Putz, G., Brunner, C., Kaiser, V., Pfurtscheller, G.: Analysis of sensorimotor rhythms for the implementation of a brain switch for healthy subjects. Biomedical Signal Processing and Control 5(1), 15–20 (2010)

[6] McFarland, D.J., Sarnacki, W.A., Townsend, G., Vaughan, T., Wolpaw, J.R.: The P-300-based brain–computer interface (BCI): Effects of stimulus rate. Clinical Neurophysiology 122(4), 731–737 (2011)

[7] Ramirez-Cortes, J.M., Alarcon-Aquino, V., Rosas-Cholula, G., Gomez-Gil, P., Escamilla-Ambrosio, J.: Anfis-Based P300 Rhythm Detection Using Wavelet Feature Extraction on Blind Source Separated EEG Signals. In: Ao, S., Amouzegar, M., Rieger, B.B. (eds.) Intelligent Automation and Systems Engineering, ch. 27. LNEE, vol. 103, pp. 353–365. Springer, New York (2011)

[8] Shyu, K.K., Lee, P.L., Liu, Y.J., Sie, J.J.: Dual-frequency steady-state visual evoked potential for brain computer interface. Neuroscience Letters 483(1), 28–31 (2010)

[9] Horki, P., Solis-Escalante, T., Neuper, C., Muller-Putz, G.R.: Hybrid Motor Imagery and Steady-state Visual Evoked Potential Based BCI for Artificial Arm Control. In: Proceedings of the First Tools for Brain Computer Interaction Workshop, Graz, Austria, p. 46 (2010)

[10] Wang, H., Li, C.S., Li, Y.G.: Brain-computer interface design based on slow cortical potentials using matlab/simulink. In: Proceedings of the International Conference on Mechatronics and Automation, Changchun, China, pp. 1044–1048 (2009)

[11] Khare, V., Santhosh, J., Anand, S., Bhatia, M.: Performance comparison of three artificial neural network methods for classification of electroencephalograph signals of five mental tasks. J. Biomedical Science and Engineering 3, 200–205 (2010)

[12] Wang, J., Xu, G., Wang, L., Zhang, H.: Feature extraction of brain-computer interface based on improved multivariate adaptive autoregressive models. In: Proceedings of the 3rd International Conference on Biomedical Engineering and Informatics (BMEI), Yantai, China, pp. 895–898 (2010)

[13] Kołodziej, M., Majkowski, A., Rak, R.J.: A New Method of EEG Classification for BCI with Feature Extraction Based on Higher Order Statistics of Wavelet Components and Selection with Genetic Algorithms. In: Dobnikar, A., Lotrič, U., Šter, B. (eds.) ICANNGA 2011, Part I. LNCS, vol. 6593, pp. 280–289. Springer, Heidelberg (2011)
[14] Vijean, V., Hariharan, M., Saidatul, A., Yaacob, S.: Mental tasks classifications using S-transform for BCI applications. In: Proceedings of the IEEE Conference on Sustainable Utilization and Development in Engineering and Technology, Semenyih, Malaysia, pp. 69–73 (2011)
[15] Lotte, F.: The use of fuzzy inference systems for classification in EEG-based brain-computer interfaces. In: Proceedings of the 3rd International Brain-Computer Interfaces Workshop and Training Course, Graz, Austria (2006)
[16] Zhang, L., He, W., He, C., Wang, P.: Improving mental task classification by adding high frequency band information. Journal of Medical Systems 34(1), 51–60 (2010)
[17] Palaniappan, R.: Utilizing Gamma band to improve mental task based brain-computer interface design. IEEE Transactions on Neural Systems and Rehabilitation Engineering 14(3), 299–303 (2006)
[18] Park, C., Looney, D., Kidmose, P., Ungstrup, M., Mandic, D.P.: Time-frequency analysis of EEG asymmetry using bivariate Empirical Mode Decomposition. IEEE Transactions on Neural Systems and Rehabilitation Engineering 19(4), 366–373 (2011)
[19] Kousarrizi, M.R.N., Ghanbari, A.A., Teshnehlab, M., Shorehdeli, M.A., Gharaviri, A.: Feature extraction and classification of EEG signals using Wavelet Transform, SVM and artificial neural networks for brain computer interfaces. In: Proceedings of the International Joint Conference on Bioinformatics, Systems Biology and Intelligent Computing, Shanghai, China, pp. 352–355 (2009)
[20] Forney, E.M., Anderson, C.W.: Classification of EEG during imagined mental tasks by forecasting with Elman recurrent neural networks. In: Proceedings of the International Joint Conference on Neural Networks, San Jose, California, USA, pp. 2749–2755 (2011)
[21] Coyle, D., McGinnity, T.M., Prasad, G.: Improving the separability of multiple EEG features for a BCI by neural-time-series-prediction-preprocessing. Biomedical Signal Processing and Control 5(3), 196–204 (2010)
[22] Oliver, G., Gedeon, T.: Brain Computer Interfaces: A Recurrent Neural Network Approach. Australian National University (2010)
[23] Fuchs, E., Gruber, C., Reitmaier, T., Sick, B.: Processing short-term and long-term information with a combination of polynomial approximation techniques and time-delay neural networks. IEEE Transactions on Neural Networks 20(9), 1450–1462 (2009)
[24] Gomez-Gil, P.: Long term prediction, chaos and artificial neural networks. Where is the meeting point? Engineering Letters 15(1), 1–5 (2007)
[25] Skarda, C., Freeman, W.: How brains make chaos in order to make sense of the world. Behavioral and Brain Sciences 10, 161–195 (1987)
[26] Elman, J.: Finding structure in time. Cognitive Science 14, 179–211 (1990)
[27] Čerňanský, M.: Training Recurrent Neural Network Using Multistream Extended Kalman Filter on Multicore Processor and Cuda Enabled Graphic Processor Unit. In: Alippi, C., Polycarpou, M., Panayiotou, C., Ellinas, G. (eds.) ICANN 2009, Part I. LNCS, vol. 5768, pp. 381–390. Springer, Heidelberg (2009)

[28] Werbos, P.: Backpropagation through time: what it does and how to do it. Proceedings IEEE 74(10), 1550–1560 (1990)
[29] Williams, R.: Some observations on the use of the extended Kalman Filter as a recurrent network learning algorithm. Technical Report NU-CCS-92-1. Northeastern University, Boston, MA (1992)
[30] Haykin, S.: Neural Networks, 2nd edn. Prentice Hall, Upper Saddle River (1999)
[31] Alanis, A., Sanchez, E., Loukianov, A.: Discrete-time adaptive backstepping nonlinear control via high-order neural networks. IEEE Transactions on Neural Networks 18(4), 1185–1195 (2007)
[32] Prokhorov, D.: Toyota prius hev neurocontrol and diagnostics. Neural Networks 21, 458–465 (2008)
[33] Ralaivola, L., d'Alché-Buc, F.: Nonlinear Time Series Filtering, Smoothing and Learning using the Kernel Kalman Filter. Technical Report. Universite Pierre et Marie Curie, Paris France (2004)
[34] García-Pedrero, A.: Arquitectura neuronal apoyada en señales reconstruidas con wavelets para predicción de series de tiempo caóticas, M. Sc. Thesis (in spanish), INAOE, Tonantzintla, Puebla (2009)
[35] Doka, K.: Handbook of brain theory and neural networks, 2nd edn. MIT Press, Cambridge (2002)
[36] Cernansky, M.: Matlab functions for training recurrent neural networks RTRL-EKF (2008), http://www2.fiit.stuba.sk/~cernans/main/download.html (accessed January 2009)
[37] Gomez-Gil, P., Garcia-Pedrero, A., Ramirez-Cortes, J.M.: Composite Recurrent Neural Networks for Long-Term Prediction of Highly-Dynamic Time Series Supported by Wavelet Decomposition. In: Castillo, O., Kacprzyk, J., Pedrycz, W. (eds.) Soft Computing for Intelligent Control and Mobile Robotics. SCI, vol. 318, pp. 253–268. Springer, Heidelberg (2010)
[38] Cernansky, M.: Matlab functions for training recurrent neural networks RTRL-EKF (2009), http://www2.fiit.stuba.sk/~cernans/main/download.html
[39] Keirn, Z.A., Aunon, J.I.: A new mode of communication between man and his surroundings. IEEE Transactions in Biomedical Engineering 37(12), 1209–1214 (1990)
[40] Cawley, G.C.: Leave-One-Out Cross-Validation Based Model Selection Criteria for Weighted LS-SVMs. In: Proceedings of the International Joint Conference on Neural Networks, Vancouver, Canada, pp. 1661–1668 (2006)

Interval Type-2 Fuzzy System for Image Edge Detection Quality Evaluation Applied to Synthetic and Real Images

Felicitas Perez-Ornelas[1], Olivia Mendoza[1], Patricia Melin[2], and Juan R. Castro[1]

[1] Baja California University, Tijuana, Mexico
{fperez,omendoza,jrcastro}@uabc.edu.mx
[2] Tijuana Institute of Technology, Tijuana, México
pmelin@tectijuana.mx

Abstract. In this paper a new method to calculate a quality index for edge detection of an image is proposed. The new method can be applied to synthetic or real images, and consists on an interval type-2 fuzzy system (IT2FS). The inputs for the IT2FS correspond to a combination of parameters representing the most influential characteristics of an edge image according on literature and our previous experience. This new index can be calculated for any edge detected image, including the traditional and fuzzy methods.

1 Introduction

The edge detection is very important in the pre-processing of an image [9][6], already have direct impact on the final result of the processing of the same; these methods are used in the areas of analysis and applications in the real world, as well as in the investigation of the vision of computer [9].

At the present, exist different metric to evaluate the edges of the image, among these are the figure of Merit (FOM) de Pratt [1], index of similarity Jaccard also called the Coefficient of Tanimoto and the coefficient of Dice, the last two are seeking the similarity between two image [22].

This paper show the design and experiment of a method that use interval type 2 fuzzy logic [6][7][1], for evaluate the edge of the image [8]. The edges detector to be used Gradient Morphological (GM), Sobel, Prewitts, Roberts, as well as the fuzzy versions. Have as input variables the most influential parameters of the edges of an image [6], according to literature review. These parameters are used by other metrics so separated, and the use in this method together [7].

2 Measure Index

The indices used to evaluate the edges detected an image are figure of Merit (FOM) Pratt [1], Jaccard index and coefficient of Dice, which are explained below. The first index is FOM, this is one of the most frequently used, calculate

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 147–157.
springerlink.com

distance perform between the two images [1], the edge of detected (Id) with the edge of reference (or) or ground truth, comparing [6] [8] the position pixel by pixel searching for the similarities of the gap and after the sum found the same. As shown in the equation (1).

$$FOM = \frac{1}{\max(N_i, N_d)} \sum_{i=1}^{N_d} \frac{1}{1+\propto d_i^2} \tag{1}$$

The second and the third index are the Jaccard and the coefficient of Dice, these are the figure of sets, represented for two circles, as shown in Fig. 1, that show the intersection of two sets, the set of results, that main, the edge detected and the set true which represent the reference edge or ground truth. The Jacard index is also called Tanimoto coefficient which expresse the division index of true positives (TP) by the sum of false positives (FP), true positives (TP) and false negatives (FN), this shown in equation 2 [22].

$$mj = \frac{TP}{FP + TP + FN} \tag{2}$$

$$md = 2\left(\frac{TP}{(FP + TP) + (TP + FN)}\right) \tag{3}$$

The last index is the coefficient of Dice, like the above is obtained based on the two sets shown in Fig. 1, using the following calculation: the division of true positives (TP) by the sum of false positives (FP) plus the true positive (TP) and true positives (TP) plus false negatives (FN), all multiplied by two, this is shown in equation 3. It is noteworthy that if the result is 0 there is no overlap, i.e. the images have no similarity [14], and if the calculated value is 1 indicating that they have a perfect overlap [22].

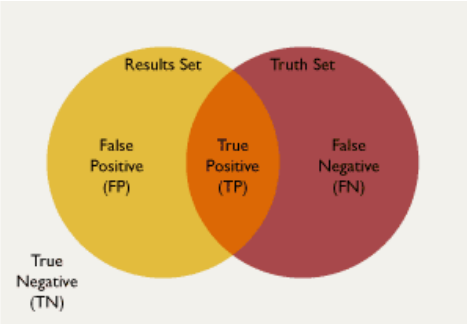


Fig. 1. Measuring Set

Most influential parameters, according to the revised literature, are shown in Table 1, these are the input variables to the fuzzy system [7][1], which will serve to calculate the index to evaluate the detection of edges in an image. These variables represent the sum used by the Pratt index, which uses the Euclidean distance to make a comparison [5][8] between the reference image and the image of detected edges, the other parameter is the Jaccard and Dice PCD this variable represents the pixels of correctly detected edges and the last parameter is PFA which means the pixels false alarms, it means that the pixels that are marked as edges and are not.

The methodology followed to perform the index calculation is described below, is shown in Fig. 2, first compute the edges of the original image, after the two images are taken, the edge detected and the reference image (ground truth) [4] and calculated their respective thresholds, these values are appropriate to the size of the images. The next step is to convert to binary images so that we can make the counts and to determine where the edges, and later extract the characteristics thereof. These are placed in the input variables to the fuzzy system for the output variable, i.e. the proposed index. Once the index value is calculated, rates of Pratt, Jaccard and Dice for comparisons of the calculated values.

Table 1. Parameters of the indixes

	Parameters		
Indexes	Summation (Sumdist)	PCD	PFA
FOM	x		
Jaccard		X	x
Dice		X	x
Index proposed	x	X	x

Described below are the characteristics of intervals type 2 fuzzy system to evaluate the detection of edges in an image [14][8]. To construct it, use a graphical tool specially made to work with these systems [2]. Fig. 3 shows the input variables, as already described above, these are Sumdist, PCD and PFA, each of the variables contains 4 membership functions which are of Gaussian type, these linguistic values are as LOW, MEDIUM , HIGH and VERYHIGH.

Fig. 4 show the output variable, the proposed index [5], which has 4 membership functions with the same linguistic values mentioned above. These variables are matched to the size of the image that is being evaluated and are normalized between 0 and 1.

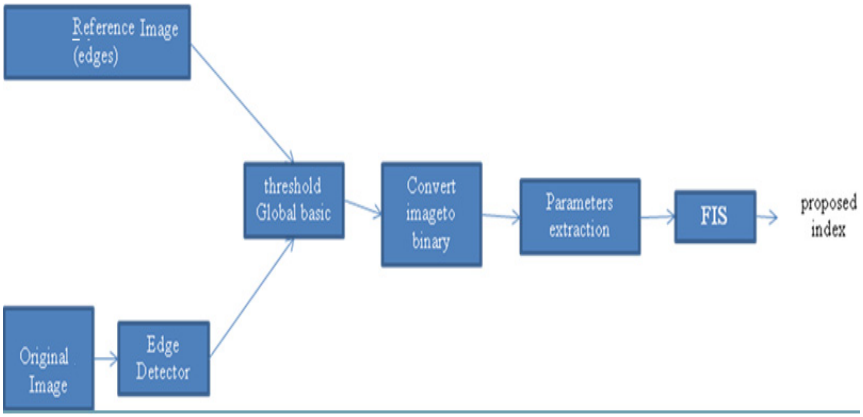


Fig. 2. Model for evaluating the edge detection

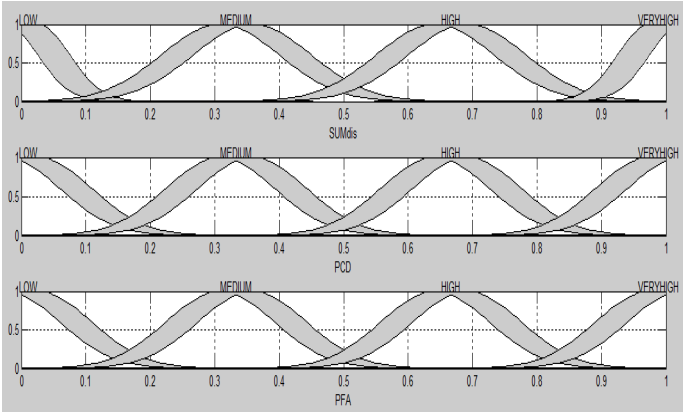


Fig. 3. Input variables and their membership functions

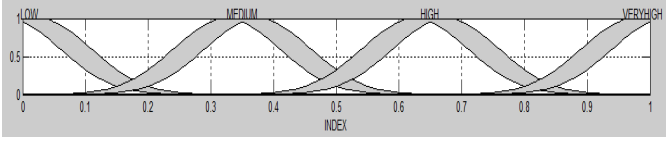


Fig. 4. Output variable and their membership function

The next step is to find the fuzzy rules to be used for the fuzzy system, which is shown in Fig. 5, which calculated the proposed index. Was performed combinatorial each of the variables and their respective linguistic values, and 64 were obtained as a result of these rules is show in Fig. 6, then with the aid of arrangements, which are displayed images to fig. 7, values of 0 and 255,

respectively edge and no edge, specially made to refine the rules and using the interpretability, we reduce them to 34. In Fig. 7 contained in a) show only edges in the middle of the image, in b) represent the edges in the top half and so on in other images. Performing a combination of these images was able to obtain the fuzzy rules.

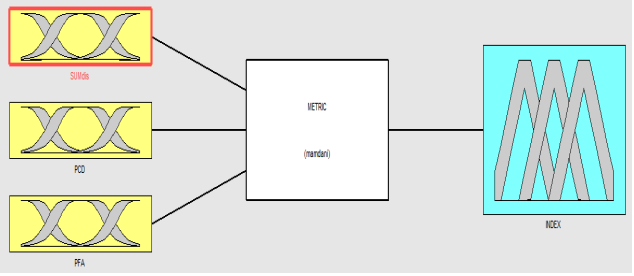


Fig. 5. Fussy System for evaluate edge detection

1. If (SUMdis is LOW) and (PCD is MEDIUM) and (PFA is HIGH) then (INDEX is LOW) (1)
2. If (SUMdis is MEDIUM) and (PCD is MEDIUM) and (PFA is HIGH) then (INDEX is MEDIUM) (1)
3. If (SUMdis is HIGH) and (PCD is MEDIUM) and (PFA is HIGH) then (INDEX is MEDIUM) (1)
4. If (SUMdis is VERYHIGH) and (PCD is MEDIUM) and (PFA is HIGH) then (INDEX is LOW) (1)
5. If (SUMdis is LOW) and (PCD is HIGH) and (PFA is HIGH) then (INDEX is LOW) (1)
6. If (SUMdis is MEDIUM) and (PCD is HIGH) and (PFA is HIGH) then (INDEX is MEDIUM) (1)
7. If (SUMdis is HIGH) and (PCD is HIGH) and (PFA is HIGH) then (INDEX is MEDIUM) (1)
8. If (SUMdis is VERYHIGH) and (PCD is HIGH) and (PFA is HIGH) then (INDEX is LOW) (1)
9. If (SUMdis is LOW) and (PCD is VERYHIGH) and (PFA is HIGH) then (INDEX is MEDIUM) (1)
10. If (SUMdis is MEDIUM) and (PCD is VERYHIGH) and (PFA is HIGH) then (INDEX is MEDIUM) (1)
11. If (SUMdis is HIGH) and (PCD is VERYHIGH) and (PFA is HIGH) then (INDEX is MEDIUM) (1)
12. If (SUMdis is VERYHIGH) and (PCD is VERYHIGH) and (PFA is HIGH) then (INDEX is MEDIUM) (1)
13. If (SUMdis is LOW) and (PCD is MEDIUM) and (PFA is MEDIUM) then (INDEX is LOW) (1)
14. If (SUMdis is MEDIUM) and (PCD is MEDIUM) and (PFA is MEDIUM) then (INDEX is MEDIUM) (1)
15. If (SUMdis is HIGH) and (PCD is MEDIUM) and (PFA is MEDIUM) then (INDEX is HIGH) (1)
16. If (SUMdis is VERYHIGH) and (PCD is MEDIUM) and (PFA is MEDIUM) then (INDEX is MEDIUM) (1)
17. If (SUMdis is LOW) and (PCD is HIGH) and (PFA is MEDIUM) then (INDEX is MEDIUM) (1)
18. If (SUMdis is MEDIUM) and (PCD is HIGH) and (PFA is MEDIUM) then (INDEX is MEDIUM) (1)
19. If (SUMdis is HIGH) and (PCD is HIGH) and (PFA is MEDIUM) then (INDEX is HIGH) (1)
20. If (SUMdis is VERYHIGH) and (PCD is HIGH) and (PFA is MEDIUM) then (INDEX is HIGH) (1)
21. If (SUMdis is LOW) and (PCD is MEDIUM) and (PFA is LOW) then (INDEX is LOW) (1)
22. If (SUMdis is MEDIUM) and (PCD is MEDIUM) and (PFA is LOW) then (INDEX is LOW) (1)
23. If (SUMdis is HIGH) and (PCD is MEDIUM) and (PFA is LOW) then (INDEX is MEDIUM) (1)
24. If (SUMdis is VERYHIGH) and (PCD is MEDIUM) and (PFA is LOW) then (INDEX is MEDIUM) (1)
25. If (SUMdis is LOW) and (PCD is HIGH) and (PFA is LOW) then (INDEX is MEDIUM) (1)
26. If (SUMdis is LOW) and (PCD is HIGH) and (PFA is LOW) then (INDEX is MEDIUM) (1)
27. If (SUMdis is MEDIUM) and (PCD is HIGH) and (PFA is LOW) then (INDEX is HIGH) (1)
28. If (SUMdis is HIGH) and (PCD is HIGH) and (PFA is LOW) then (INDEX is HIGH) (1)
29. If (SUMdis is VERYHIGH) and (PCD is HIGH) and (PFA is LOW) then (INDEX is HIGH) (1)
30. If (PFA is VERYHIGH) then (INDEX is LOW) (1)
31. If (PCD is LOW) then (INDEX is LOW) (1)
32. If (PCD is VERYHIGH) and (PFA is LOW) then (INDEX is VERYHIGH) (1)
33. If (PCD is VERYHIGH) and (PFA is MEDIUM) then (INDEX is VERYHIGH) (1)
34. If (PFA is VERYHIGH) then (INDEX is LOW) (1)

Fig. 6. Fuzzy Rules

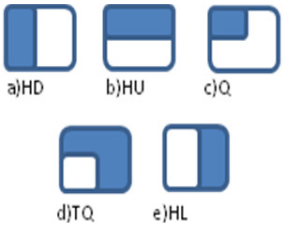


Fig. 7. Images of the arrays

3 Reference Image (Ground Truth)

There are different methods to determine the reference image or ground truth [4], i.e. the edges of an image, one of them is called Automatic generation of ground truth consensus. This methodology uses a comparison [6][8] of edge detection techniques, using reference images obtained in various ways, the first is the artificial ground truth, which easily can be obtained synthetically [7], the second is to obtain the ground truth manually, this is tedious and heavy [4]. The latest

Fig. 8. Images for first experiment of Automatic generation of consensus ground truth

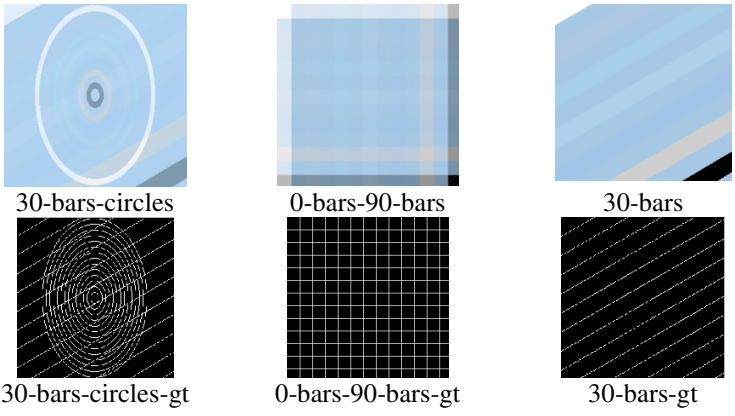


Fig. 9. Images for second experiment of Automatic generation of consensus ground truth

methodology is to get the ground truth through a comparison of 10 edge detection technique and the reference image is obtained by the methods of Minimean and Minimax [4]. To obtain a comparison of different sources for obtaining ground truth, we will conduct the following experiments, the first being a comparison between real images and ground truth consensus, Fig. 8, the second will be artificial image with ground truth artificial (synthetic) Fig. 9 and the third and last will be with real images and manual ground truth fig. 10.

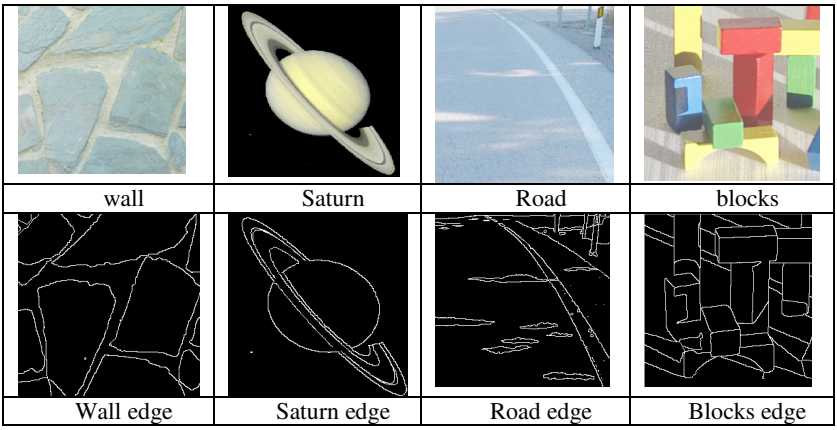


Fig. 10. Images for third experiment of Automatic generation of consensus ground truth

4 Experiments

The first experiment that was working with the arrangements show in Fig. 7, these were used to determine the fuzzy rules, which are show in Fig. 6, each of the images have 100 rows by 100 columns and the elements that comprise it are 0 and 255 means no edge means to ensure the number of edges that had the image for example we have HR in item a) of Fig. 7 which contains half of its elements 0 and the other half 255, i.e., Q in b) we have elements with value 0 only a quarter of all elements and so on, with other images.

The following experiments were performed from the methodology proposed by Automatic generation of consensus ground truth [4], the first was that of the images shown in Fig. 8, and these are real images and ground truth consensus. What was done was to use the reference image obtained the consensus and measures the results with the index proposed by this paper comparing [6][8] rates Jaccar, Dice and Pratt. To make comparisons of results obtained with different edges detector such as Sobel, Prewitts, Roberts, Morphological Gradient as well as fuzzy variants. In the second experiment were used images that shown in Fig. 9, these are synthetic images [7] as well as their edges. In the third experiment were used real images, shown in Fig. 10 with ground truth images manually made.

5 Results

The result of the arrangements show in Table 2, which were obtained from the images shown in Fig 7, we remember that these arrangements were made to help define the fuzzy rules to be used in intervals type 2 fuzzy system. The first column show II, this refer to the actual image, and the ID come to mean the reference image, in some cases the results are very different, for example in the comparison of LH and RH, the index shown in the third column with a FOU = 0 is 0.03 while the rate of FOM in the last column is 0.5435, meaning that it evaluates FOM almost 50% more than metFis, as this index is the false alarms parameter and FOM take not in their calculations. So we could say that this new index evaluated in a more objective image. In Table 3, show the results obtained with the first ground truth experiment consensus [4], which have real image shown in Fig. 8, and their respective ground truth consensus.

Table 2. Result of measurements obtained with arrays

		FOU 0	FOU 0.2	FOU 0.4	FOU 0.6			
II	ID	metFis	metFis	metFis	metFis	mj	mc	FOM
HL	Q	0.4262	0.4143	0.4087	0.4092	0.5000	0.6667	0.5000
Q	HL	0.3702	0.3751	0.4035	0.4463	0.5000	0.6667	0.7717
HL	HU	0.3702	0.3751	0.4035	0.4463	0.3333	0.5000	0.7717
HU	HL	0.3702	0.3751	0.4035	0.4463	0.3333	0.5000	0.7717
HR	TQ	0.3506	0.3506	0.3495	0.3487	0.6667	0.8000	0.8478
TQ	HR	0.6544	0.6532	0.6500	0.6373	0.6667	0.8000	0.6667
HL	HL	0.9630	0.9615	0.9564	0.9461	1.0000	1.0000	1.0000
HL	HR	0.0366	0.0377	0.0418	0.0495	0.0000	0.0000	0.5434
HR	HL	0.0366	0.0377	0.0418	0.0495	0.0000	0.0000	0.5434

Table 3. Results of the measurements made in lenna

	FOU 0	FOU 0.2	FOU 0.4	FOU 0.6			
ED	metFis	metFis	metFis	metFis	mj	mc	FOM
sobel	0.3183	0.3196	0.3225	0.3501	0.1429	0.2501	0.3839
prewitt	0.3304	0.332	0.3407	0.3687	0.1463	0.2553	0.3962
gm	0.4619	0.4642	0.4706	0.4779	0.185	0.3122	0.4889
roberts	0.346	0.3476	0.3621	0.3812	0.1568	0.271	0.4035
fuzzysobel	0.386	0.3912	0.4071	0.4252	0.1597	0.2755	0.4541
fuzzyprewitt	0.395	0.4017	0.4174	0.4322	0.1614	0.2779	0.4612
fuzzygm	0.4334	0.4425	0.4591	0.4823	0.2038	0.3386	0.5829
fuzzyroberts	0.3756	0.3794	0.3959	0.4203	0.1671	0.2864	0.4391

Table 4. Results of the measurements made in the 0-bars and 90-bars image

	FOU 0	FOU 0.2	FOU 0.4	FOU 0.6			
ED	metFis	metFis	metFis	metFis	mj	mc	FOM
sobel	0.3093	0.3139	0.3322	0.3712	0.1446	0.2526	0.4059
prewitt	0.3112	0.3166	0.3353	0.3743	0.1446	0.2526	0.4083
gm	0.3881	0.3923	0.4012	0.4110	0.1383	0.2430	0.4717
roberts	0.3836	0.3881	0.3981	0.4091	0.1385	0.2432	0.4693
fuzzysobel	0.3884	0.3925	0.4014	0.4112	0.1383	0.243	0.4719
fuzzyprewitt	0.3884	0.3925	0.4014	0.4112	0.1383	0.243	0.4719
fuzzygm	0.3884	0.3925	0.4014	0.4112	0.1383	0.243	0.4719
fuzzyroberts	0.3893	0.3934	0.4021	0.4116	0.1386	0.2434	0.4722

Table 5. Results of the measurements made in the 30-bars-ellipses image

	FOU 0	FOU 0.2	FOU 0.4	FOU 0.6			
ED	metFis	metFis	metFis	metFis	mj	mc	FOM
sobel	0.6171	0.6087	0.577	0.5236	0.1657	0.2842	0.5826
prewitt	0.6151	0.6064	0.5755	0.5241	0.1669	0.286	0.5766
gm	0.6152	0.6065	0.5757	0.5244	0.167	0.2862	0.5766
roberts	0.6191	0.6109	0.5788	0.5235	0.1631	0.2805	0.5963
fuzzysobel	0.6191	0.6110	0.5798	0.5265	0.1621	0.2790	0.6241
fuzzyprewitt	0.6127	0.6009	0.5645	0.5135	0.1608	0.2771	0.638
fuzzygm	0.3537	0.3543	0.3515	0.3679	0.1473	0.2567	0.7693
fuzzyroberts	0.6193	0.6113	0.4126	0.5241	0.1633	0.2808	0.5969

Table 6. Results of the measurements made in the 30-bars image

	FOU 0	FOU 0.2	FOU 0.4	FOU 0.6			
ED	metFis	metFis	metFis	metFis	mj	mc	FOM
sobel	0.0498	0.054	0.0669	0.1069	0.0598	0.1103	0.1549
prewitt	0.0498	0.054	0.0669	0.1069	0.0584	0.1103	0.1549
gm	0.0498	0.054	0.0669	0.1069	0.0598	0.1112	0.1463
roberts	0.0498	0.054	0.0669	0.1069	0.0571	0.108	0.1745
fuzzysobel	0.0498	0.0540	0.067	0.107	0.0557	0.1056	0.1987
fuzzyprewitt	0.0499	0.054	0.0671	0.1073	0.0541	0.1027	0.2272
fuzzygm	0.5336	0.5235	0.4984	0.4697	0.1498	0.2606	0.5218
fuzzyroberts	0.0498	0.0571	0.0571	0.1069	0.0571	0.108	0.1745

The second experiment work with images created synthetically and their respective reference image is also obtained ground truth synthetic [7], shows in fig. 9. From these results we can see that very similar and we can still say that the index proposed evaluation makes a more objective to the image, this sow in table 4, 5, and 6.

Table 7. Results of the measurements made in image Blocks, Road, Saturn and Wall.

				SOBEL EDGE				
		FOU 0	FOU 0.2	FOU 0.4	FOU 0.6			
I	II			metFis		mj	mc	FOM
blocks.png	blocks-manual-	0.445	0.4455	0.4444	0.4883	0.3176	0.4821	0.9193
road.png	road-manual-gt.png	0.380	0.459	0.4974	0.4999	0.2915	0.4514	0.8205
saturn.png	saturn-manual-	0.177	0.1773	0.2164	0.2221	0.3832	0.5541	0.9290
wall.png	wall-manual-gt.png	0.064	0.0652	0.073	0.0947	0.1296	0.2294	0.4985
				SOBEL EDGE - FUZZY - FOU = 0				
blocks.png	blocks-manual-	0.167	0.1677	0.1678	0.1678	0.3189	0.4835	0.8513
road.png	road-manual-gt.png	0.129	0.1287			0.2564	0.4081	0.7408
saturn.png	saturn-manual-	0.167				0.3533	0.5220	0.9077
wall.png	wall-manual-gt.png	0.062				0.1171	0.2096	0.4664
				PREWITT EDGE				
blocks.png	blocks-manual-	0.445	0.426	0.4382	0.4643	0.3139	0.4778	0.920
road.png	road-manual-gt.png	0.500	0.491	0.4991	0.4999	0.2973	0.4583	0.836
saturn.png	saturn-manual-	0.170	0.170	0.203	0.2212	0.3654	0.5352	0.925
wall.png	wall-manual-gt.png	0.065	0.065	0.0737	0.0953	0.1308	0.2313	0.505
				PREWITT EDGE - FUZZY - FOU = 0				
blocks.png	blocks-manual-	0.167	0.167	0.1678	0.1678	0.3189	0.4835	0.851
road.png	road-manual-gt.png	0.128	0.127	0.1573	0.166	0.2564	0.4058	0.734
saturn.png	saturn-manual-	0.168	0.168	0.1814	0.2194	0.3572	0.5263	0.913
wall.png	wall-manual-gt.png	0.062	0.062	0.0649	0.0861	0.1142	0.2049	0.462
				GM EDGE				
blocks.png	blocks-manual-	0.449	0.450	0.4578	0.4984	0.2906	0.4504	0.896
road.png	road-manual-gt.png	0.113	0.113	0.1139	0.148	0.2193	0.3597	0.653
saturn.png	saturn-manual-	0.203	0.210	0.2219	0.3692	0.4424	0.5940	0.928
wall.png	wall-manual-gt.png	0.063	0.063	0.0673	0.089	0.1217	0.2171	0.469
				ROBERTS EDGE				
blocks.png	blocks-manual-	0.262	0.326	0.4056	0.4	0.2885	0.4478	0.898
road.png	road-manual-gt.png	0.112	0.113	0.1134	0.1	0.2084	0.3450	0.639
saturn.png	saturn-manual-	0.226	0.2303	0.4032	0.4	0.4080	0.5795	0.933
wall.png	wall-manual-gt.png	0.061	0.0621	0.064	0.0	0.1067	0.1929	0.448

The latest results are show in Table 7, here we compare [6][8] the values of real images with ground truth were obtained manually. The edge detectors [8] we use are Sobel, Prewitt, GM and Roberts, like fuzzy variants [14]. The result of processing of these images was more significant because here reflect the most relevant results, and make evaluation more objective, i.e. all image in the values were lower than the giver by FOM, where it is clear that the parameters are used to evaluate metFis most influential in the evaluation of the edges.

6 Conclusions

We can evaluate edge detection with a fuzzy method [5][7][17][8]. This is the first time that a fuzzy method [9][14] for evaluating edge detection in an image has been proposed. We can include parameters that other metrics use separately. This index is more objective to evaluate image edge detection [5].

References

1. Abdou, E., Pratt, W.K.: Quantitative design and evaluation of enhancement/ thresholding edge detectors. Proceedings of the IEEE 67(5), 753–763 (1979)
2. Castro, J.R., Castillo, O., Melin, P.: An Interval Type-2 Fuzzy Logic Tool-box for Control Applications, pp. 1–6 (July 2007)

3. Castro, J.R., Castillo, O., Melin, P., Rodríguez-Díaz, A.: A hybrid learning algorithm for a class of interval type-2 fuzzy neural networks. Information Sciences 179(13), 2175–2193 (2009)
4. Fernandez-Garcia, N.L., Carmona-Poyato, A., Medina-Carnicer, R., Madrid-Cuevas, F.J.: Automatic generation of concensus ground truth for the comparison of edge detection techniques. Image and Vision Computing 26, 496–511 (2008)
5. Kan, S.H.: Metrics and Models in Software Quality Engineering, 2nd edn. Addison-Wesley (2003)
6. Melin, P., Felix, C., Castillo, O.: Face recognition using modular neural networks and the fuzzy Sugeno integral for response integration. Int. J. Intell. Syst (IJIS) 20(2), 275–29 (2005)
7. Melin, P., Mendoza, O., Castillo, O.: Face Recognition With an Improved Interval Type-2 Fuzzy Logic Sugeno Integral and Modular Neural Networks. IEEE Transactions on Systems, Man, and Cybernetics, Part A (TSMC) 41(5), 1001–1012 (2011)
8. Melin, P.: Image Processing and Pattern Recognition with Mamdani Interval Type-2 Fuzzy Inference Systems. Combining Experimentation and Theory, 179–190 (2012)
9. Melin, P.: Modular Neural Networks and Type-2 Fuzzy Systems for Pattern Recognition. Springer (2012)
10. Mendel, J.M.: A quantitative comparison of interval type-2 and type-1 fuzzy logic systems: first results, pp. 1–8 (July 2010)
11. Mendel, J.M.: On the Continuity of Type-1 and Interval Type-2 Fuzzy Logic Systems. IEEE Transactions on Fuzzy Systems 19(1), 179–192 (2011)
12. Mendoza, O., Melin, P., Licea, G.: A New Method for Edge Detection in Image Processing Using Interval Type-2 Fuzzy Logic. In: IEEE International Conference on Granular Computing, GRC 2007, p. 151 (2007)
13. Mendoza, O., Licea, G., Melin, P.: Modular Neural Networks and Type-2 Fuzzy Logic for Face Recognition. In: Fuzzy Information Processing Society, NAFIPS 2007. Annual Meeting of the North American, pp. 622–627 (2007)
14. Mendoza, O., Melin, P., Licea, G.: Type-2 Fuzzy Systems for Improving Training Data and Decision Making in Modular Neural Networks for Image Recognition. In: International Joint Conference on Neural Networks, IJCNN 2007, pp. 2931–2935 (2007)
15. Mendoza, O., Melín, P., Castillo, O.: Interval type-2 fuzzy logic and modular neural networks for face recognition applications. Applied Soft Computing 9(4), 1377–1387 (2009)
16. Mendoza, O., Melin, P., Licea, G.: Interval type-2 fuzzy logic for edges detection in digital images. Int. J. Intell. Syst (IJIS) 24(11), 1115–1133 (2009)
17. Mendoza, O., Melin, P., Licea, G.: A hybrid approach for image recognition combining type-2 fuzzy logic, modular neural networks and the Sugeno integral. Information Sciences 179(13), 2078–2101 (2009)
18. Mendoza, O., Melin, P., Castillo, O.: Neural networks recognition rate as index to compare the performance of fuzzy edge detectors. In: IJCNN 2010, pp. 1–6 (2010)
19. Mendoza, O., Melin, P., Castillo, O., Castro, J.R.: Comparison of Fuzzy Edge Detectors Based on the Image Recognition Rate as Performance Index Calculated with Neural Networks. Soft Computing for Recognition Based on Biometrics 2010, 389–399 (2010)
20. Pratt, W.K.: Digital Image Processing, 3rd edn. John Wiley & Sons, Inc. (2001)
21. Prieto, M.S., Allen, A.R.: A similarity metric for edge images. IEEE Transactions on Pattern Analysis and Machine Intelligence 25(10), 1265–1273 (2003)
22. Shattuck, D.W., Prasad, G., Mirza, M., Narr, K.L.: Online resource for validation of brain segmentation methods. In: Laboratory of Neuro Imaging, David Geffen School of Medicine, University of California, Los Angeles, 635 Charles Young Drive South, NRB1, Suite 225, Los Angeles, California 90095, USA

Genetic Optimization of Type-2 Fuzzy Weight Adjustment for Backpropagation in Ensemble Neural Network

Fernando Gaxiola, Patricia Melin, and Fevrier Valdez

Tijuana Institute of Technology, Tijuana México
fergaor_29@hotmail.com, pmelin@tectijuana.mx,
fevrier@tectijuana.edu.mx

Abstract. In this paper a genetic algorithm is used to optimize the three neural networks forming an ensemble. Genetic algorithms are also used to optimize the two type-2 fuzzy systems that work in the backpropagation learning method with type-2 fuzzy weight adjustment. The mathematical analysis of the proposed learning method architecture and the adaptation of type-2 fuzzy weights are presented. The proposed method is based on recent methods that handle weight adaptation and especially fuzzy weights. In this work an ensemble neural network of three neural networks and average integration to obtain the final result is presented. The proposed approach is applied to a case of time series prediction.

1 Introduction

This paper is focused on the optimization of a neural network ensemble with type-2 fuzzy weights. The optimization is performed in the number of neurons in the hidden layer and in the type-2 fuzzy inference systems used in the hidden and output layer to obtain the type-2 fuzzy weights of each neural network.

The proposed approach is applied to time series prediction for the Mackey-Glass series. The objective is obtaining the minimum prediction error for the data of the time series.

A supervised neural network was used, because this type of network is the most commonly used in the area of time series prediction.

This neural network is based on supervised learning, where the network operates by having both the correct input and output, and the network adjusts its weights to try in minimize the error of the calculated output.

The research is based on working with the weights of a neural network in a different way to the traditional approach, which is important because this affect the performance of the learning process of the neural network.

This conclusion is based on the practice of neural networks of this type, where some research works have shown that the training of neural networks for the same problem initialized with different weights or its adjustment in a different way, but at the end is possible to reach a similar result.

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 159–171.
springerlink.com

The next section presents a background about modifications of the backpropagation algorithm and different management strategies of weights in neural networks, and basic concepts of neural networks. Section 3 explains the proposed method and the problem description. Section 4 describes the optimization of the ensemble neural network with type-2 fuzzy weights proposed in this paper. Section 5 presents the simulation results for the proposed method. Finally, in section 6, some conclusions are presented.

2 Background and Basic Concepts

In this section a brief review of basic concepts is presented.

2.1 Neural Network

An artificial neural network (ANN) is a distributed computing scheme based on the structure of the nervous system of humans. The architecture of a neural network is formed by connecting multiple elementary processors, this being an adaptive system that has an algorithm to adjust their weights (free parameters) to achieve the performance requirements of the problem based on representative samples [8][22].

The most important property of artificial neural networks is their ability to learn from a training set of patterns, i.e. they are able to find a model that fits the data [31][9].

The artificial neuron consists of several parts (see Figure. 1). On one side are the inputs, weights, the summation, and finally the transfer function. The input values are multiplied by the weights and added: $\sum x_i w_{ij}$. This function is completed with the addition of a threshold amount i. This threshold has the same effect as an input with value -1. It serves so that the sum can be shifted left or right of the origin. After addition, the function f is applied to the sum, resulting on the final value of the output, also called y_i [28], obtaining the following equation:

$$y_i = \left(\sum_{i=1}^{n} x_i w_{ij} \right) \tag{1}$$

Where f may be a nonlinear function with binary output + -1, a linear function f (z) = z, or as sigmoidal logistic function:

$$f(z) = \frac{1}{1+e^{-z}}. \tag{2}$$

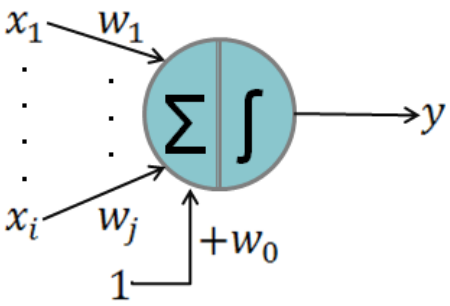


Fig. 1. Schematics of an artificial neuron

2.2 Overview of Related Works

The backpropagation algorithm and its variations are the most useful basic training methods in the area of neural networks. However, these algorithms are usually too slow for practical applications.

When applying the basic backpropagation algorithm to practical problems, the training time can be very high. In the literature several methods have been proposed to accelerate the convergence of the algorithm [2] [13] [24] [36].

There exists many works about adjustment or managing of weights but only the most important and relevant for this research will be considered here [4] [10] [27] [35]:

Momentum method.- Rumelhart, Hinton and Williams suggested adding in the increased weights expression a momentum term β, to filter the oscillations that can be formed a higher learning rate that lead to great change in the weights [28] [14].
Adaptive learning rate.- focuses on improving the performance of the algorithm by allowing the learning rate changes during the training process (increase or decrease) [14].

Conjugate Gradient algorithm.- A search of weight adjustment along conjugate directions. Versions of conjugate gradient algorithm differ in the way in which a constant βk is calculated. Some examples of this case are

- Fletcher-Reeves update [12].
- Polak-Ribiere updated [12].
- Powell-Beale Restart [3] [29].
- Scaled Conjugate Gradient [25].

Kamarthi and Pittner [20], focused in obtaining a weight prediction of the network at a future epoch using extrapolation.

Ishibuchi et al. [17], proposed a fuzzy network where the weights are given as trapezoidal fuzzy numbers, denoted as four trapezoidal fuzzy numbers for the four parameters of trapezoidal membership functions.

Ishibuchi et al. [18], proposed a fuzzy neural network architecture with symmetrical fuzzy triangular numbers for the fuzzy weights and biases, denoted by the lower, middle and upper limit of the fuzzy triangular numbers.

Feuring [11], based on the work by Ishibuchi, where triangular fuzzy weights are used, developed a learning algorithm in which the backpropagation algorithm is used to compute the new lower and upper limits of weights. The modal value of the new fuzzy weight is calculated as the average of the new computed limits.

Castro et al. [6], use interval type-2 fuzzy neurons for the antecedents and interval of type-1 fuzzy neurons for the consequents of the rules. This approach handles the weights as numerical values to determine the inputs of the fuzzy neurons, as the scalar product of the weights for the input vector.

Recent works on type-2 fuzzy logic have been developed in time series prediction, like that of Castro et al. [7], and other researchers [21][1].

Recent research on genetic algorithm optimization have been developed in neural networks and fuzzy logic, like that of Sanchez et al. [32], and other researchers [30][34].

3 Proposed Method and Problem Description

The focus of this work is to generalize the backpropagation algorithm using type-2 fuzzy sets to allow the neural network to handle data with uncertainty. At the same time, it will be necessary to optimize type-2 fuzzy sets for the corresponding applications and this will require a method to automatically vary the footprint of uncertainty (FOU) of the membership functions.

The initial weight selection will be done differently to the traditional random initialization of weights performed with the backpropagation algorithm (Fig. 2); the proposed method will work with type-2 fuzzy weights, taking into account the possible change in the way the neuron works internally and the adaptation of the weights given in this way (Fig. 3) [26].

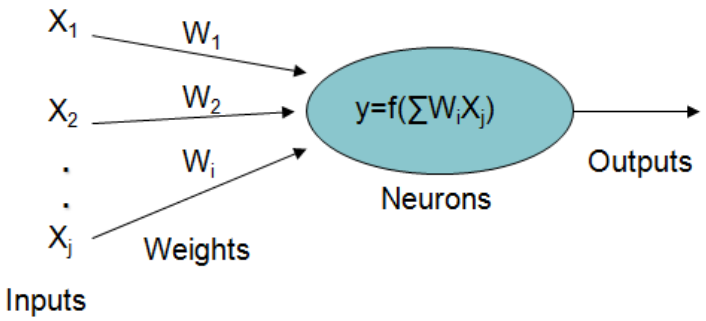


Fig. 2. Scheme of current management of numerical weights (type-0) for the inputs of each neuron

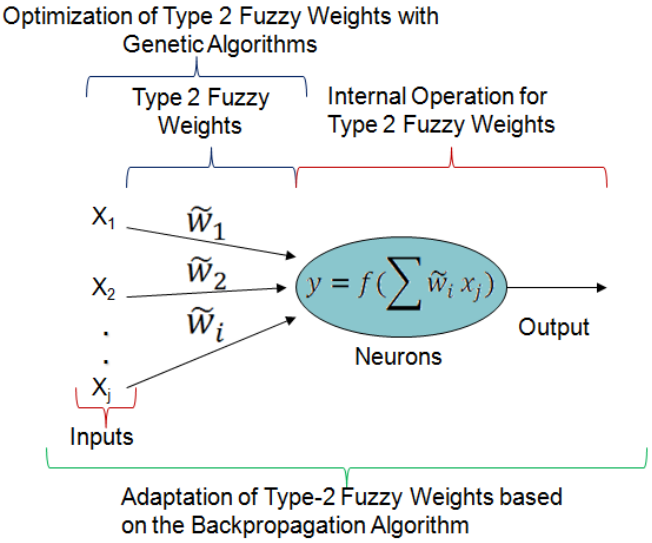


Fig. 3. Schematic of the proposed management of type-2 fuzzy weights for the inputs of each neuron

A modification of the current methods of adjusting weights that allow convergence to the correct weights for the problem was considered. A method for adjusting weights to achieve the desired result, searching for the optimal way to work with type-2 fuzzy weights was proposed [19].

A genetic algorithm for obtaining the optimal type-2 fuzzy weights of the neural network is used; because in the literature it can be found that it has been very difficult and exhaustive to manually find optimal values for a problem [16].

To define the activation function f (-) to use, the linear and sigmoidal functions were tested, because these functions have been used in similar approaches.

4 Optimization of Ensemble Neural Network Architecture with Type-2 Fuzzy Weights

The proposed ensemble neural network architecture with type-2 fuzzy weights (see Fig. 4) is described as follows:

Layer 0: Inputs.

$$x = [x_1, x_2, \cdots, x_n] \tag{2}.$$

Layer 1: Interval type-2 fuzzy weights for the hidden layer of each neural network.

$$\widetilde{w} = \left[\underline{w}, \overline{w}\right] \tag{3}.$$

Where $[\underline{w}, \overline{w}]$ are the weights of the consequents of each rule of the type-2 fuzzy system with inputs (current fuzzy weight, change of weight) and output (new fuzzy weight).

Layer 2: Hidden neuron with interval type-2 fuzzy weights.

$$Net = \sum_{i=1}^{n} x_i \widetilde{w_i} \tag{4}$$

Layer 3: Output neuron with interval type-2 fuzzy weights.

$$Out = \sum_{i=1}^{n} y_i \widetilde{w_i} \tag{5}$$

Layer 4: Obtain a single output of each one of the three neural networks.

Layer 5: Obtain a final output with the average integration.

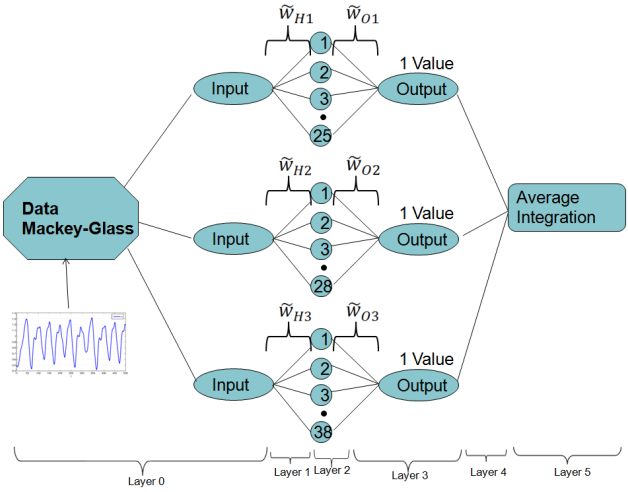


Fig. 4. Ensemble neural network architecture with type-2 fuzzy weights

The optimization was performed for the numbers of neurons in the hidden layer of each neural network, and for the weights of the hidden layer and output layer, in Figure 5 is described:

Experiments in time-series prediction were performed, specifically for the Mackey-Glass time series (τ=17).

Three neural networks were considered in the ensemble: the first network with 25 neurons in the hidden layer and 1 neuron in the output layer; the second network with 28 neurons in the hidden layer and 1 neuron in the output layer; and the third network with 38 neurons in the hidden layer and 1 neuron in the output layer. This ensemble neural network handles type-2 fuzzy weights in each one of its hidden layers and output layer. In each hidden layer and output of each network a type-2 fuzzy inference system is used to obtain new weights in each epoch of the network [5][23][15][33].

The combination of responses of the ensemble neural network is performed by average integration.

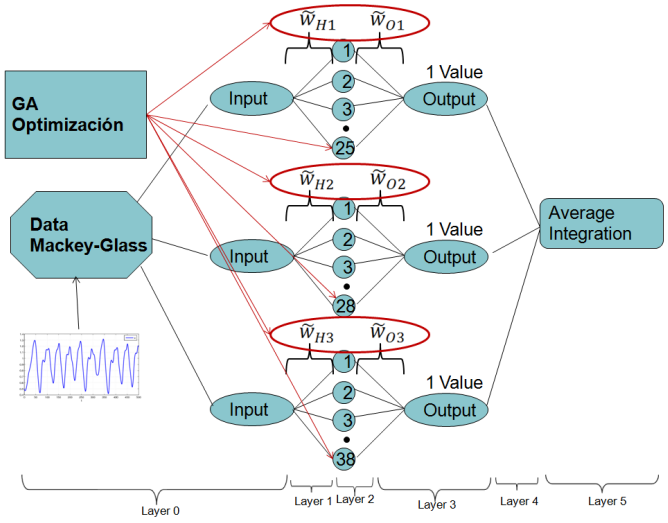


Fig. 5. Proposed optimization of the ensemble neural network architecture with type-2 fuzzy weights

5 Simulation Results

The obtained results without optimizing the neural network and type-2 fuzzy systems are shown on Table 1 and Fig. 6, which means that all parameters of the neural network and type-2 fuzzy systems are established empirically. The best prediction error is of 0.0788.

Table 1. Results for the ensemble neural network for series Mackey-Glass

No.	Epoch	Network error	Time	Prediction error
E1	100	0.000000001	00:01:09	0.0788
E2	100	0.000000001	00:02:11	0.0905
E3	100	0.000000001	00:02:12	0.0879
E4	100	0.000000001	00:01:14	0.0822
E5	100	0.000000001	00:01:13	0.0924
E6	100	0.000000001	00:02:13	0.0925
E7	100	0.000000001	00:01:08	0.0822
E8	100	0.000000001	00:01:09	0.0924
E9	100	0.000000001	00:01:07	0.0826
E10	100	0.000000001	00:01:07	0.0879

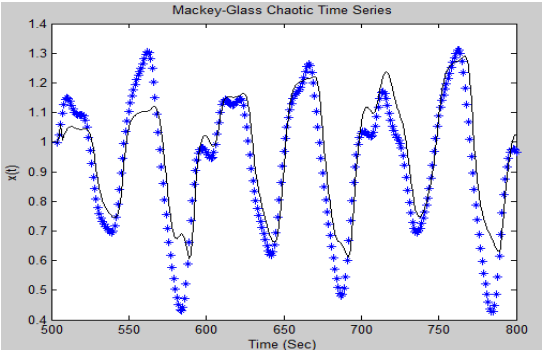


Fig. 6. Plot of real data against prediction data of the Mackey-Glass time series for the ensemble neural network with type-2 fuzzy weights

The parameters for the GA used to optimize the ensemble neural network are described in Table 2:

Table 2. Parameters of the genetic algorithm used for optimization the ensemble neural network

Individuals	40
Gens	81 (binary)
Generations	20
Assign Fitness	Ranking
Selection	Stochastic Universal Sampling
Crossover	Single-Point
Mutation	0.0086
Individuals	40

The obtained results of the GA optimizing the ensemble neural network are shown on Table 3 and Fig. 7. The best error is of 0.0518 optimizing the numbers of neurons and type-2 fuzzy systems.

Table 3. Results for the ensemble neural network for series Mackey-Glass optimized

No.	Prediction error
E1	0.0518
E2	0.0611
E3	0.0787
E4	0.0715
E5	0.0655
E6	0.0614
E7	0.0724
E8	0.0712
E9	0.0724

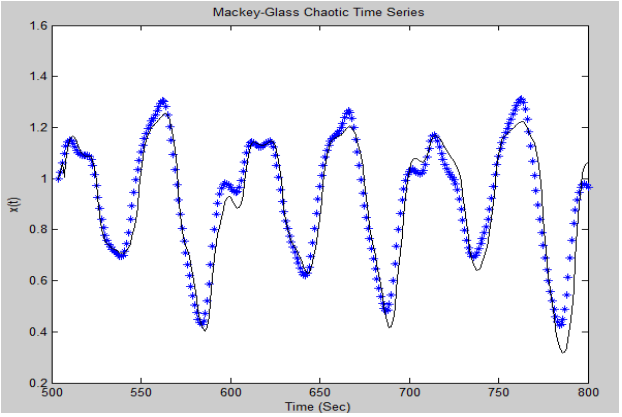


Fig. 7. Plot of real data against prediction data of the Mackey-Glass time series for the ensemble neural network with optimized type-2 fuzzy weights

We obtained 2 similar type-2 fuzzy systems in each neural network with 2 inputs and 1 output for the hidden layer (See Figs. 8(a) and 9) and 6 rules (see Fig. 11(a)); and output layer (See Figs. 8(b) and 10) and 6 rules (see Fig. 11(b)):

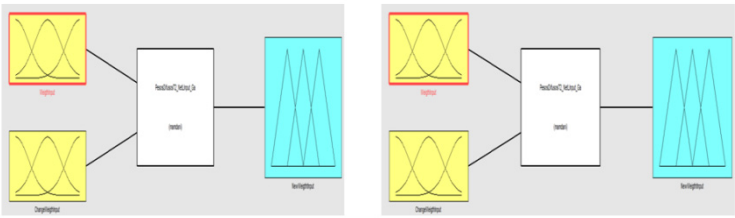


Fig. 8. Structure of the used type-2 fuzzy inference system in the hidden and output layer

Fig. 9. Inputs and outputs of the type-2 fuzzy inference system for the hidden layer

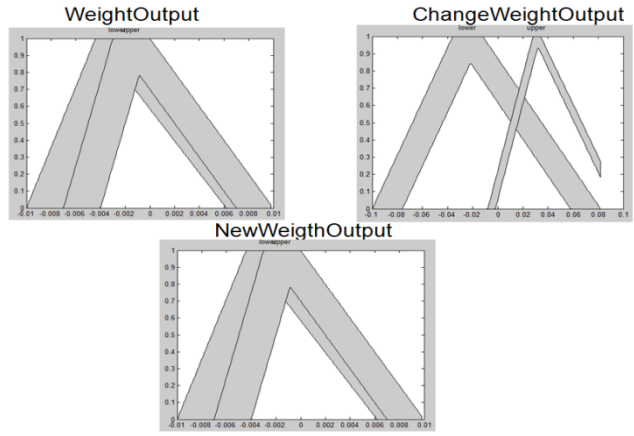


Fig. 10. Inputs and outputs of the type-2 fuzzy inference system for the output layer

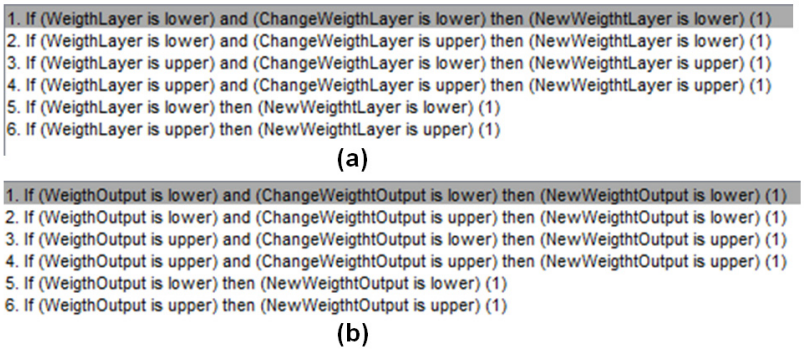


Fig. 11. Structure of the used type-2 fuzzy inference system in the hidden and output layer

6 Conclusions

An ensemble neural network learning method with type-2 fuzzy weights was optimized with a genetic algorithm. The result with the ensemble neural network with type-2 fuzzy weights optimized for the Mackey-Glass time series is a prediction error of 0.0518. The architecture for the optimized ensemble neural network: the first network with 30 neurons in the hidden layer and 1 neuron in the output layer; the second network with 29 neurons in the hidden layer and 1 neuron in the output layer; and the third network with 26 neurons in the hidden layer and 1 neuron in the output layer.

This result is good considering that the number of GA generations was relatively small.

References

1. Abiyev, R.H.: A Type-2 Fuzzy Wavelet Neural Network for Time Series Prediction. In: García-Pedrajas, N., Herrera, F., Fyfe, C., Benítez, J.M., Ali, M. (eds.) IEA/AIE 2010, Part III. LNCS, vol. 6098, pp. 518–527. Springer, Heidelberg (2010)
2. Barbounis, T.G., Theocharis, J.B.: Locally Recurrent Neural Networks for Wind Speed Prediction using Spatial Correlation. Information Sciences 177(24), 5775–5797 (2007)
3. Beale, E.M.L.: A Derivation of Conjugate Gradients. In: Lootsma, F.A. (ed.) Numerical Methods for Nonlinear Optimization, pp. 39–43. Academic Press, London (1972)
4. Casasent, D., Natarajan, S.: A Classifier Neural Net with Complex-Valued Weights and Square-Law Nonlinearities. Neural Networks 8(6), 989–998 (1995)
5. Castillo, O., Melin, P.: A review on the design and optimization of interval type-2 fuzzy controllers. Applied Soft Computing 12(4), 1267–1278 (2012)
6. Castro, J., Castillo, O., Melin, P., Rodríguez-Díaz, A.: A Hybrid Learning Algorithm for a Class of Interval Type-2 Fuzzy Neural Networks. Information Sciences 179(13), 2175–2193 (2009)

7. Castro, J., Castillo, O., Melin, P., Mendoza, O., Rodríguez-Díaz, A.: An Interval Type-2 Fuzzy Neural Network for Chaotic Time Series Prediction with Cross-Validation and Akaike Test. Soft Computing for Intelligent Control and Mobile Robotics, 269–285 (2011)
8. Cazorla, M., Escolano, F.: Two Bayesian Methods for Junction Detection. IEEE Transaction on Image Processing 12(3), 317–327 (2003)
9. De Wilde, O.: The Magnitude of the Diagonal Elements in Neural Networks. Neural Networks 10(3), 499–504 (1997)
10. Draghici, S.: On the Capabilities of Neural Networks using Limited Precision Weights. Neural Networks 15(3), 395–414 (2002)
11. Feuring, T.: Learning in Fuzzy Neural Networks. In: IEEE International Conference on Neural Networks, vol. 2, pp. 1061–1066 (1996)
12. Fletcher, R., Reeves, C.M.: Function Minimization by Conjugate Gradients. Computer Journal 7, 149–154 (1964)
13. Gedeon, T.: Additive Neural Networks and Periodic Patterns. Neural Networks 12(4-5), 617–626 (1999)
14. Hagan, M.T., Demuth, H.B., Beale, M.H.: Neural Network Design, p. 736. PWS Publishing, Boston (1996)
15. Hagras, H.: Type-2 Fuzzy Logic Controllers: A Way Forward for Fuzzy Systems in Real World Environments. In: IEEE World Congress on Computational Intelligence, pp. 181–200 (2008)
16. Haupt, R., Haupt, S.: Practical Genetic Algorithms, p. 272. John Wiley and Sons, Inc., Hoboken (2004)
17. Ishibuchi, H., Morioka, K., Tanaka, H.: A Fuzzy Neural Network with Trapezoid Fuzzy Weights, Fuzzy Systems. In: IEEE World Congress on Computational Intelligence, vol. 1, pp. 228–233 (1994)
18. Ishibuchi, H., Tanaka, H., Okada, H.: Fuzzy Neural Networks with Fuzzy Weights and Fuzzy Biases. In: IEEE International Conference on Neural Networks, vol. 3, pp. 1650–1655 (1993)
19. Jang, J.S.R., Sun, C.T., Mizutani, E.: Neuro-Fuzzy and Soft Computing: a Computational Approach to Learning and Machine Intelligence, p. 614. Prentice Hall (1997)
20. Kamarthi, S., Pittner, S.: Accelerating Neural Network Training using Weight Extrapolations. Neural Networks 12(9), 1285–1299 (1999)
21. Karnik, N., Mendel, J.: Applications of Type-2 Fuzzy Logic Systems to Forecasting of Time-Series. Information Sciences 120(1-4), 89–111 (1999)
22. Martinez, G., Melin, P., Bravo, D., Gonzalez, F., Gonzalez, M.: Modular Neural Networks and Fuzzy Sugeno Integral for Face and Fingerprint Recognition. Advances in Soft Computing 34, 603–618 (2006)
23. Melin, P.: Modular Neural Networks and Type-2 Fuzzy Systems for Pattern Recognition, pp. 1–204. Springer (2012)
24. Meltser, M., Shoham, M., Manevitz, L.: Approximating Functions by Neural Networks: A Constructive Solution in the Uniform Norm. Neural Networks 9(6), 965–978 (1996)
25. Moller, M.F.: A Scaled Conjugate Gradient Algorithm for Fast Supervised Learning. Neural Networks 6, 525–533 (1993)
26. Monirul Islam, M.D., Murase, K.: A New Algorithm to Design Compact Two-Hidden-Layer Artificial Neural Networks. Neural Networks 14(9), 1265–1278 (2001)
27. Neville, R.S., Eldridge, S.: Transformations of Sigma–Pi Nets: Obtaining Reflected Functions by Reflecting Weight Matrices. Neural Networks 15(3), 375–393 (2002)

28. Phansalkar, V.V., Sastry, P.S.: Analysis of the Back-Propagation Algorithm with Momentum. IEEE Transactions on Neural Networks 5(3), 505–506 (1994)
29. Powell, M.J.D.: Restart Procedures for the Conjugate Gradient Method. Mathematical Programming 12, 241–254 (1977)
30. Pulido, M., Melin, P., Castillo, O.: Genetic Optimization of Ensemble Neural Networks for Complex Time Series Predicition. In: IJCNN, pp. 202–206 (2011)
31. Salazar, P.A., Melin, P., Castillo, O.: A New Biometric Recognition Technique Based on Hand Geometry and Voice Using Neural Networks and Fuzzy Logic. Soft Computing for Hybrid Intelligent Systems, 171–186 (2008)
32. Sanchez, D., Melin, P.: Modular Neural Network with Fuzzy Integration and its Optimization Using Genetic Algorithms for Human Recognition based on Iris, Ear and Voice Biometrics. Soft Computing for Recognition Based on Biometrics, 85–102 (2010)
33. Sepúlveda, R., Castillo, O., Melin, P., Montiel, O.: An Efficient Computational Method to Implement Type-2 Fuzzy Logic in Control Applications. Analysis and Design of Intelligent Systems using Soft Computing Techniques, 45–52 (2007)
34. Valdez, F., Melin, P., Parra, H.: Parallel Genetic Algorithms for Optimization of Modular Neural Networks in Pattern Recognition. In: IJCNN, pp. 314–319 (2011)
35. Yam, J., Chow, T.: A Weight Initialization Method for Improving Training Speed in Feedforward Neural Network. Neurocomputing 30(1-4), 219–232 (2000)
36. Yeung, D., Chan, P., Ng, W.: Radial Basis Function Network Learning using Localized Generalization Error Bound. Information Sciences 179(19), 3199–3217 (2009)

A New Method for Type-2 Fuzzy Integration in Ensemble Neural Networks Based on Genetic Algorithms

Martha Pulido and Patricia Melin

Tijuana Institute of Technology, Tijuana México
pmelin@tectijuana.mx

Abstract. This paper describes a proposed method for type-2 fuzzy integration that can be used in the fusion of responses for an ensemble neural network. We consider the case of the design of a type-2 fuzzy integrator for fusion of a neural network ensemble. The network structure of the ensemble may have a maximum of 5 modules. This integrator consists of 32 fuzzy rules, with 5 inputs depending on the number of modules of the neural network ensemble and one output. Each input and output linguistic variable of the fuzzy system uses Gaussian membership functions. The performance of type-2 fuzzy integrators is analyzed under different levels of uncertainty to find out the best design of the membership functions. In this case the proposed method is applied to time series prediction.

Keywords: Ensemble Neural Networks, Genetic Algorithms, Optimization, Time Series Prediction.

1 Introduction

In the literature there are methods for building intelligent systems using type-2 fuzzy logic and soft computing techniques. The soft computing paradigm includes techniques such as: neural networks, type-1 fuzzy systems and genetic algorithms, which can be used to create hybrid intelligent systems. So we can use type-2 fuzzy logic combined with these conventional soft computing techniques for the creation of hybrid intelligent systems and the in this way combining advantages that each technique offers us, as this allows us to solve problems of pattern recognition, time series and control.

Neural networks (NNs) are composed of many elements (Artificial Neurons), grouped into layers that are highly interconnected (with the synapses), this structure has several inputs and outputs, which are trained to react (or give values) in a way you want to input stimuli. These systems emulate in some way, the human brain. Neural networks are required to learn to behave (Learning) and someone should be

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 173–182.
springerlink.com

responsible for the teaching or training, based on prior knowledge of the environment problem [8,13,14].

Neural network ensemble is a learning paradigm where many neural networks are jointly used to solve a problem [1]. Neural network ensemble is a learning paradigm where a collection of a finite number of neural networks is trained for the same task [27]. It originates from Hansen and Salamon's work [10], which shows that the generalization ability of a neural network system can be significantly improved through ensembling a number of neural networks, i.e. training many neural networks and then combining their predictions.

The basics of fuzzy logic do not change from type-1 to type-2 fuzzy sets, and in general, will not change for any type-n (Karnik & Mendel 1998). A higher-type number just indicates a higher "degree of fuzziness". Since a higher type changes the nature of the membership functions, the operations that depend on the membership functions change; however, the basic principles of fuzzy logic are independent of the nature of membership functions and hence, do not change. Rules of inference like Generalized Modus Ponens or Generalized Modus Tollens continue to apply.

The structure of the type-2 fuzzy rules is the same as for the type-1 case because the distinction between type-2 and type-1 is associated with the nature of the membership functions. Hence, the only difference is that now some or all the sets involved in the rules are of type-2 [2].

Genetic Algorithms (GA) as tool for a search and optimizing methodology has now reached a mature stage. The term *genetic algorithm,* almost universally abbreviated nowadays to GA, was first used by Holland [11, whose book *Adaptation in Natural and Artificial Systems* of 1975 was instrumental in creating what is now a flourishing field of research and application that goes much wider than the original GA. The subject now includes evolution strategies (ES), evolutionary programming (EP), artificial life (AL), classifier systems (CS), genetic programming (GP), and most recently the concept of evolvable hardware.[7, 15].

In this paper the contribution is the proposed method for creating a type-2 fuzzy systems for integration and optimization for to integrate responses ensemble neural network that can be applied to different cases, as can be data, images, is presented. In this case we show the application to time series prediction. The time series prediction are very important because we can analyze past events to know the possible behavior of futures events and thus we can take preventive or corrective decisions to help avoid unwanted circumstances.[16]. In the literature there have been recent work of time series [2,3, 4, 9, 10, 11, 12,17, 24, 25, 26, 27, 28].

2 Problem Statement and Proposed Method

The objective of this work is to develop a model that is based on integrating the responses of an ensemble neural network using ype-2 fuzzy systems and optimization.

Figure 1 represents the general arquitecture of the proposed method, where historical data, analyzing data, create the ensemble neural network and integrate responses of the ensemble neural network with type-2 fuzzy system integration and we obtaining the ouput are shown. The information can be historical data, these can be images, time series, etc., in this case we show the application to time series prediction of the Dow Jones where we obtain good results with this series.

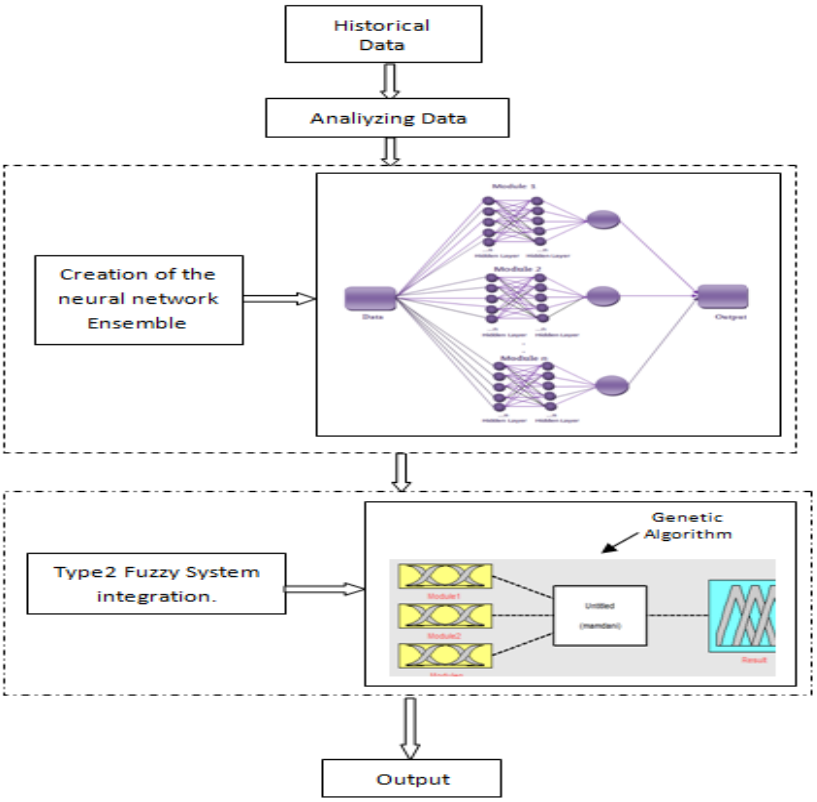


Fig. 1. General Architecture of the proposed method

In this Figure 2 shows a type-2 fuzzy system consisting of with 5 inputs depending on the number of modules of the neural network ensemble and one output. Each input and output linguistic variable of the fuzzy system uses 2 Gaussian membership functions. The performance of the type-2 fuzzy integrators is analyzed under different levels of uncertainty to find out the best design of the membership functions and consist of 32 rules.

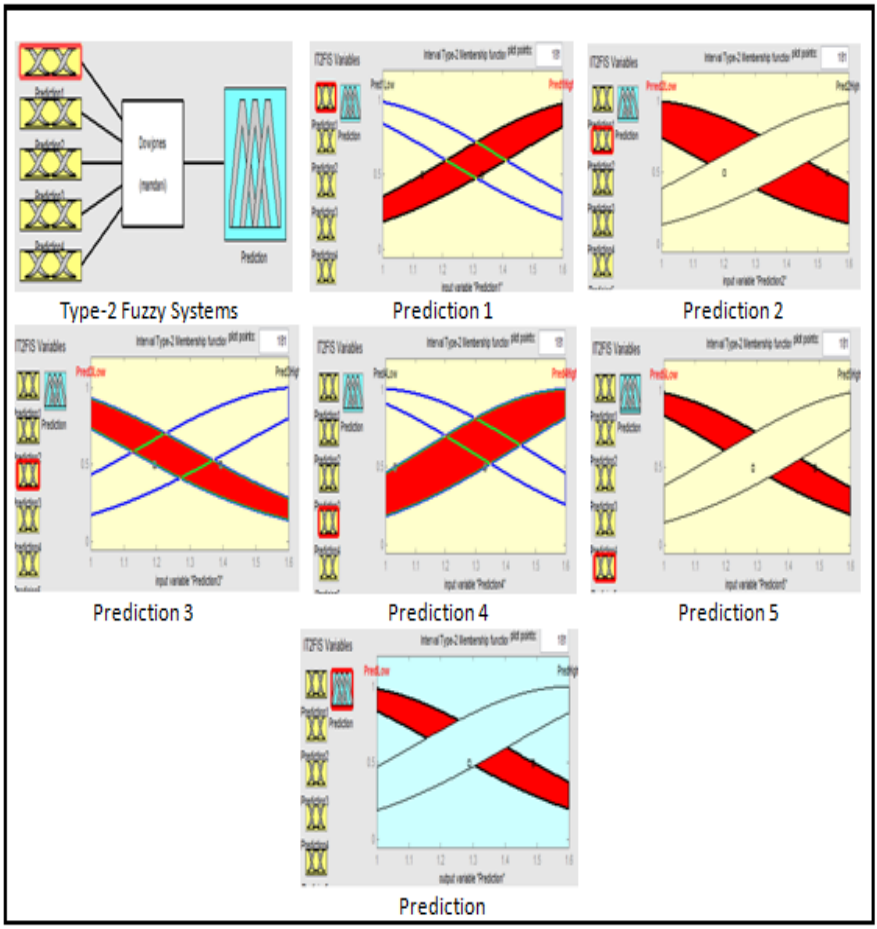


Fig. 2. Type-2 Fuzzy System for Dow Jones time series

In this Figure 3 shows the possible rules of a type-2 fuzzy system.

Chromosome shown in Figure 4 is composed of 326 genes which allow us to optimize the structure of the fuzzy integrator. Genes 1-4 (binary) are activating genes that define the structure of the fuzzy integrator such as the number of entries in the ensemble neural network, system type (Mamdani or Sugeno) and type of membership functions (Gaussian, bell generalized triangular and trapezoidal). Genes 5-292 (real numbers) allow us to manage the parameters of membership functions for input and output (depending on the type of logic, system type and function type use only part of the chromosome). 293-324 genes (binary) allow us to reduce the number of fuzzy rules, activating or deactivating them.

Fig. 3. Rules of Type-2Fuzzy Inference System of the Dow Jones Time Series

Historical data of the Dow Jones time series was used for the ensemble neural network trainings, where each module was fed with the same information, unlike the modular networks, where each module is fed with different data, which leads to architectures that are not uniform.

The Dow Jones (DJ) is a U.S. company that publishes financial information. Founded in 1882 by three reporters: Charles Henry Dow, Edward David Jones and Charles Milford Bergstresser.

In the same year it began publishing a financial newsletter called "The Customer` s Afternoon Letter which "would be the precursor of the famous financial newspaper The Wall Street Journal first published on July 8, 1889.

The newsletter showed publicly share prices and the financial accounts of companies, information that until then had only the people close to the companies.

To better represent the movements of the stock market at the time, the Dow Jones designed a barometer of economic activity meter with twelve companies creating the Dow Jones stock index.

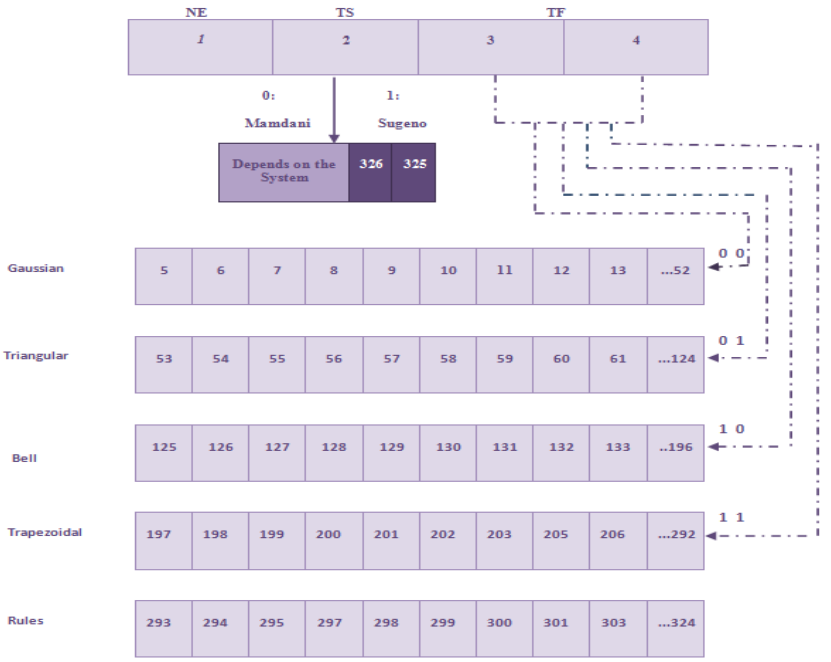


Fig. 4. Chromosome to optimize the structure of the fuzzy system to integrate ensemble neural network

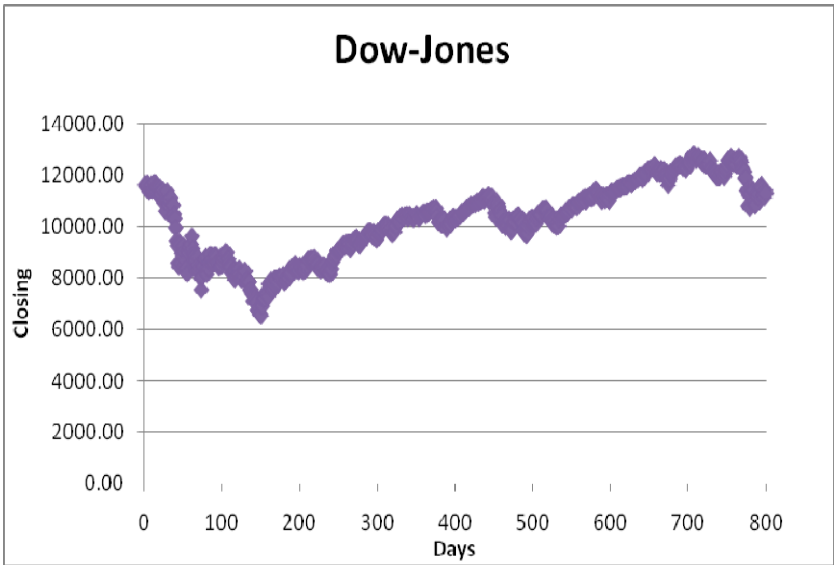


Fig. 5. Dow Jones Time Serie

Like the New York Times and Washington Post newspapers, the company is open to the market but is controlled the by the private sector. So far, the company is controlled by the Bancroft family, which controls 64% of the shares entitled to vote [6].

Data of the Dow Jones time series: We are using 800 points that correspond to a period from 08/12/2008 to 09/09/2011 (as shown in Fig. 5). We used 70% of the data for the ensemble neural network trainings and 30% to test the network [7].

3 Simulation Results

In this section we present the simulation results obtained with the integration of ensemble neural network with type-2 fuzzy integration and optimization with the genetic algorithm for the Dow Jones time series.

Table 1 shows the genetic algorithm results (as shown in Figure 5) where the prediction error is of 0.0022606.

Table 1.

No.	Gen.	Ind.	GGP	Selectión	Mutatión	Pm	Crossover	Pc	Núm. Modules	Núm. Layers	Núm. Neurons	Duratión	Predicción Error
1	100	100	0.85	rws	mutbga	0.02	xovdprs	0.5	4	2	15 21 9 10 16 18 15 26	11:14:55	0.0022606
2	100	100	0.85	rws	mutbga	0.09	xovdprs	0.5	5	2	18 9 15 24 8 11 22 15 1 5	06:23:23	0.0035056
3	100	100	0.85	rws	mutbga	0.06	xovdprs	1	4	2	2 22 9 19 19 4 8 18	04:28:10	0.0039193
4	100	100	0.85	rws	mutbga	0.08	xovdprs	0.5	5	1	2 23 23 6 2	08:43:09	0.0036413
5	100	20	0.85	rws	mutbga	0.07	xovdprs	0.5	2	2	3 2 12 19	00:45:02	0.0042879
6	100	30	0.85	rws	mutbga	0.05	xovdprs	0.7	4	3	19 25 7 18 28 26 8 24 15 1 7	02:36:09	0.0038975
7	100	30	0.85	rws	mutbga	0.06	xovdprs	0.9	5	3	26 3 18 16 9 8 12 18 20 19 25 1 2 27 20	13:23:21	0.0040413
8	100	30	0.85	rws	mutbga	0.06	xovdprs	0.9	5	1	2 14 2 11 1	03:45:19	0.0038831
9	100	25	0.85	rws	mutbga	0.09	xovdprs	0.3	5	2	13 22 8 1 16 18 16 25 1 10	02:11:25	0.0036539
10	100	20	0.85	rws	mutbga	0.03	xovdprs	0.6	5	3	12 8 25 17 29 19 12 10 26 9 3 9 1 3 6	04:36:10	0.0040967

Fuzzy integration is also performed by implementing a type-2 fuzzy system in which the results were as follows: for the best evolution with a degree of uncertainty of 0.3 a forecast error of 0.0121 was obtained, and with a degree of uncertainty of 0.4 the error: 0.0127 and with a degree uncertainty 0.5 the error of 0.0134, as shown in Table 2.

Table 2. Results of Type-2 Fuzzy Integration of DJ

Evolution	0.3 Uncertainty	0.4 Uncertainty	0.5 Uncertainty
Evolution 1	0.0179	0.0179	0.0165
Evolution2	0.0121	0.0127	0.0134
Evolution3	0.0167	0.0167	0.0162
Evolution4	0.0158	0.0178	0.0156
Evolution5	0.0237	0.0224	0.0231
Evolution6	0.0172	0.0172	0.0164
Evolution 7	0.0162	0.0162	0.0164
Evolution8	0.0168	0.0169	0.0141
Evolution 9	0.0202	0.0202	0.0212
Evolution 10	0.0154	0.0152	0.0149

The type-2 fuzzy system than obtained by applying the genetic algorithm was of Mamdani type, the membership functions of triangular type and consists of 32 rules, as shown in Table 3. Where now the prediction error is of 0.0021271.

Table 3. Results of Type-2 Fuzzy Integration of DJ with a GA

No.	Ind.	Gen.	GGP	Selection	Mutation	Pm	Crossover	Pc	Pm Rules	TS	TF	Num Rules	Duration	GA Error	Prediction Error
1	200	200	0.85	rws	Mutbga	0.7	Xovdprs	0.5	0.002	M	Tri	32	04:25:45	0.0031271	0.0021271
2	100	100	0.85	rws	mutbga	0.05	Xovdprs	0.75	0.002	M	Gbell	28	03:54:13	0.0045036	0.0035036
3	200	200	0.85	rws	mutbga	0.07	Xovdprs	0.9	0.002	M	Tri	32	04:16:06	0.0051251	0.0041251
4	100	100	0.85	rws	mutbga	0.06	Xovdprs	0.8	0.002	M	Tri	32	05:51:55	0.0047015	0.0037015
5	200	200	0.85	rws	mutbga	0.07	Xovdprs	0.6	0.002	M	Tri	32	04:03:14	0.0038993	0.0028993
21	200	300	0.85	rws	mutbga	0.07	Xovdprs	0.5	0.002	M	Tri	32	20:21:20	0.0038587	0.0028587
22	300	300	0.85	rws	mutbga	0.08	Xovdprs	0.5	0.002	M	Tri	32	22:03:52	0.0038926	0.0028926
23	300	300	0.85	rws	mutbga	0.03	Xovdprs	0.8	0.002	M	Tri	32	21:26:08	0.0040999	0.0030999
24	100	100	0.85	rws	mutbga	0.02	Xovdprs	0.8	0.002	M	Gbell	32	03:52:16	0.005615	0.004615
25	200	200	0.85	rws	mutbga	0.04	Xovdprs	0.6	0.002	M	Tri	32	18:24:03	0.0039764	0.0029764
26	100	100	0.85	rws	mutbga	0.0001	Xovdprs	0.6	0.002	M	Gbell	28	04:55:54	0.0049827	0.0039827
27	100	100	0.85	rws	mutbga	0.01	Xovdprs	0.6	0.002	M	Gbell	32	03:58:50	0.0058492	0.0048492
28	150	100	0.85	rws	mutbga	0.0001	Xovdprs	0.9	0.002	M	Gbell	26	05:27:33	0.0058857	0.0048857
29	300	300	0.85	rws	mutbga	0.01	Xovdprs	0.6	0.002	M	Tri	32	23:23:45	0.0050869	0.0040869
30	200	200	0.85	rws	mutbga	0.05	Xovdprs	0.7	0.002	M	Tra	32	18:45:25	0.0045608	0.003608
31	100	100	0.85	rws	mutbga	0.04	Xovdprs	0.8	0.002	M	Tra	32	05:51:42	0.0055117	0.0045117
32	100	100	0.85	rws	mutbga	0.03	Xovdprs	1	0.002	M	Gau	32	05:50:45	0.0044821	0.0034821
33	100	100	0.85	rws	mutbga	0.03	Xovdprs	0.5	0.002	M	Tri	32	05:052:48	0.0050814	0.0040814
34	200	200	0.85	rws	mutbga	0.007	Xovdprs	0.09	0.002	M	Gbell	28	08:32:14	0.005907	0.004907
35	300	300	0.85	rws	mutbga	0.07	Xovdprs	0.5	0.002	M	Tri	32	08:06:26	0.0048829	0.0038829

4 Conclusions

Type-2 fuzzy systems handle uncertainty in the membership functions and this helps us obtain good results in the integration of this time series as the data used are complex. After achieving these results, we have verified the efficiency of the algorithms applied to optimize the type-2 fuzzy System architecture. In this case, the method was efficient but it also has certain disadvantages, sometimes the results are not good but genetic algorithms can be considered as good techniques for solving search and optimization problems.

Acknowledgment. We would like to express our gratitude to the CONACYT, Tijuana Institute of Technology for the facilities and resources granted for the development of this research.

References

[1] Brockwell, P.D., Richard, A.D.: Introduction to Time Series and Forecasting, pp. 1–219. Springer, New York (2002)
[2] Castillo, O., Melin, P.: Type-2 Fuzzy Logic: Theory and Applications. Neural Networks, 30–43 (2008)
[3] Castillo, O., Melin, P.: Hybrid intelligent systems for time series prediction using neural networks, fuzzy logic, and fractal theory. IEEE Transactions on Neural Networks 13(6), 1395–1408 (2002)
[4] Castillo, O., Melin, P.: Simulation and Forecasting Complex Economic Time Series Using Neural Networks and Fuzzy Logic. In: Proceedings of the International Neural Networks Conference, vol. 3, pp. 1805–1810 (2001)
[5] Castillo, O., Melin, P.: Simulation and Forecasting Complex Financial Time Series Using Neural Networks and Fuzzy Logic. In: Proceedings the IEEE the International Conference on Systems, Man and Cybernetics, vol. 4, pp. 2664–2669 (2001)
[6] Dow Jones Company (10 de Enero de, 2010), http://www.dowjones.com
[7] Dow Jones Indexes (September 5, 2010), http://www.djindexes.com
[8] Golberg, D.: Genetic Algorithms in search, optimization and machine learning. Addison Wesley (1989)
[9] Jang, J.S.R., Sun, C.T., Mizutani, E.: Neuro-Fuzzy and Sof Computing. Prentice Hall (1996)
[10] Karnik, N., Mendel, M.: Applications of type-2 fuzzy logic systems to forecasting of time-series. Information Sciences 120(1-4), 89–111 (1999)
[11] Krogh, A., Vedelsby, J.: Neural network ensembles, cross validation, and active learning. In: Tesauro, G., Touretzky, D., Leen, T. (eds.) Advances in Neural Information Processing Systems 7, Denver, CO, pp. 231–238. MIT Press, Cambridge (1995) 1001
[12] Holland, J.: Adaptation in natural and artificial systems. University of Michigan Press (1975)
[13] Maguire, L.P., Roche, B., McGinnity, T.M., McDaid, L.J.: Predicting a chaotic time series using a fuzzy neural network 12(1-4), 125–136 (1998)

[14] Melin, P., Castillo, O., Gonzalez, S., Cota, J., Trujillo, W., Osuna, P.: Design of Modular Neural Networks with Fuzzy Integration Applied to Time Series Prediction, vol. 41, pp. 265–273. Springer, Heidelberg (2007)
[15] Multaba, I.M., Hussain, M.A.: Application of Neural Networks and Other Learning. Technologies in Process Engineering. Imperial College Press (2001)
[16] Plummer, E.A.: Time series forecasting with feed-forward neural Networks: uidelines and limitations. University of Wyoming (July 2000)
[17] Pulido, M., Mancilla, A., Melin, P.: An Ensemble Neural Network Architecture with Fuzzy Response Integration for Complex Time Series Prediction. In: Castillo, O., Pedrycz, W., Kacprzyk, J. (eds.) Evolutionary Design of Intelligent Systems in Modeling, Simulation and Control. SCI, vol. 257, pp. 85–110. Springer, Heidelberg (2009)
[18] Sharkey, A.: One combining Artificial of Neural Nets. Department of Computer Science University of Sheffield, U.K (1996)
[19] Sharkey, A.: A Combining artificial neural nets: ensemble and modular multinet systems. Springer, London (1999)
[20] Shimshoni, Y.N.: Intrator Classification of seismic signal by integrating ensemble of neural networks. IEEE Transactions Signal Processing 461(5), 1194–1201 (1998)
[21] Sollich, P., Krogh, A.: Learning with ensembles: how over-fitting can be useful. In: Touretzky, D.S., Mozer, M.C., Hasselmo, M.E. (eds.) Advances in Neural Information Processing Systems 8, Denver, CO, pp. 190–196. MIT Press, Cambridge (1996)
[22] Yadav, R.N., Kalra, P.K., John, J.: Time series prediction with single multiplicative neuron model. Soft Computing for Time Series Prediction, Applied Soft Computing 7(4), 1157–1163 (2007)
[23] Yao, X., Liu, Y.: Making use of population information in evolutionary artificial neural networks. IEEE Transactions on Systems, Man and Cybernetics - Part B: Cybernetics 28(3), 417–425 (1998)
[24] Zhao, L., Yang, Y.: PSO-based single multiplicative neuron model for time series prediction. Expert Systems with Applications 36(2), Part 2, 2805–2812 (March 2009)
[25] Zhou, Z.-H., Jiang, Y., Yang, Y.-B., Chen, S.-F.: Lung cancer cell identification based on artificial neural network ensembles. Artificial Intelligence in Medicine 24(1), 25–36 (2002)
[26] Reeves, R.C., Row, E.J.: Genetic Algorithms: Principles and Perspectives, A Guide to GA Theory, pp. 2–17. Kluwer Academic Publishers, New York (2003)
[27] Yadav, R.N., Kalra, P.K., John, J.: Time series prediction with single multiplicative neuron model. Soft Computing for Time Series Prediction, Applied Soft Computing 7(4), 1157–1163 (2007)
[28] Zhao, L., Yang, Y.: PSO-based single multiplicative neuron model for time series prediction. Expert Systems with Applications 36(2), Part 2, 2805–2812 (2009)

A Hand Geometry Biometric Identification System Utilizing Modular Neural Networks with Fuzzy Integration

José Luis Sánchez and Patricia Melin

Tijuana Institute of Technology, Tijuana, México
pmelin@tectijuana.mx

Abstract. The present work deals with the problem of identifying individuals from a database, and in so doing utilizing measurements taken from handpalm images. The techniques utilized for performing identifications are mainly those of artificial neural networks, which work upon the data through the use of two modular neural networks, one which is concerned solely with the handpalm image, another with the measurements taken thereof. Outputs from these two networks are integrated through a fuzzy inference system. Subsequent work will comprise improvement of the obtained results.

Keywords: Biometry, Fuzzy Logic, Modular Neural Networks.

1 Introduction

In recent decades an ever increasing number of automatic recognition methods have been developed, all of which are intended to help manage and control access to many different goods, places and services. Computational Intelligence paradigms such as Artificial neural networks (ANN) and Fuzzy systems (based on fuzzy logic) have proven invaluably useful when applied to pattern recognition problems, as well as being able to perform at very good levels of performance when dealing with such problems, these being the reason they are used in this work.

Features to be found in a human's hands are attractive as a means to build upon for the construction of methods for recognition; not the least of qualities associated with them are their permanence and uniqueness.

Among these methods only two will be mentioned: palmprint recognition and hand geometry recognition [3].

Palmprint recognition is an amplification of fingerprint recognition methods, and may or may not build upon the presence of three principal lines on most everyone's hands.

This kind of methods tend to be very precise, yet spend sizable amounts of computing power.

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 183–199.
springerlink.com

Hand geometry methods, in their original implementation, look to make several measurements from images of the outline of the hand (these images are taken while the hand is on a level surface and placed between unmovable pegs, to improve measurements).

The measurements commonly include: length of fingers, palm width, knuckle size.

Commonly, a very reduced amount of memory is needed for a single individual, but identification tasks aren't as accurately resolved through such methods, more so if the database is really large; for verification purposes, performance can be good [3].

The rest of the paper is organized as follows: Section 2 describes the tools used for this work and what they are about, section 3 describes the methodology applied for the resolution of the stated problem, section 4 describes the results obtained so far, section 5 gives the conclusions drawn so far, along with ideas for further work.

2 Computational Intelligence Tools

There is an enormous amount of Computational Intelligence tools that are useful for pattern recognition purposes. Mentioned below are those that are important to this work.

2.1 Fuzzy Logic

Fuzzy logic follows a path unlike that of traditional logic, allowing not only two possible states for a situation, but an infinite gamut between those two traditional states, which in a sense approaches more closely the way in which humans use language and what has traditionally been considered a byproduct of language: reasoning.

From this particular standpoint, fuzzy logic builds until it reshapes every tool of logic, including deductive inference [2].

2.2 Fuzzy Systems

A fuzzy inference system (FIS) is a system that takes advantage of fuzzy logic to create and control models of complex systems and situations without making use of ground-up specifications, which are many times unavailable and relying instead on descriptions more or less at a human language level.

A much used tool for working with fuzzy systems is that included within the Matlab programming language [2].

2.3 Artificial Neural Networks

Artificial neural networks are much simplified models of generic nervous systems, studied with the idea of applying them for the same tasks organisms excel at, including pattern recognition.

Nowadays there are many kinds of ANNs, but all build upon using lots of simple elements (neurons) capable of being interconnected and of maintaining weighted responses to "stimuli" between those connections; the similarities end there for many types of ANNs.

The afore mentioned networks are tipically divided into layers, and for those nets needing training to get the desired response to a given input, such training is accomplished through a training algorithm, like the much used backpropagation algorithm.

There would be a last stumbling stone to consider when in need to have a certain output derive from a given input: not all network architectures are equally successful at attaining this goal [5].

2.4 Modular Neural Networks (MNN)

It is possible to divide a problem so that several networks (modules or experts) work separately to solve it. Much needed improvements can be obtained with these MNN, such as a lower training time or an efficiency boost.

The final step in getting an answer to a given problem with an MNN involves the integration of individual results, to mention just a few (not all applicable to every MNN): gating networks, voting, weighted average, winner-takes-all.

Winner takes all is possibly the simplest method and works by considering the highest ranking results in each module and comparing all such results, allowing only the very highest ranking to remain and be part of the final result [8].

It is also possible to have a fuzzy system integrate responses to several modules, according to the system's rules and the compared modules' results.

3 Methodology

The following items were necessary aspects for this work's problem solving methodology:

1. Having a database.
2. Having a general architecture for the system.
3. Having an architecture for each of the used modular neural networks.
4. Having an architecture for each of the modules within a modular neural network.
5. Having preprocessing modules for each of the MNNs.
6. Having an architecture for each of the used fuzzy integrators.

7. Having fuzzy rules for each of the fuzzy integrators.
8. Train all modules of every modular neural network and integrate results.

3.1 Utilized Database

The database used with this work is the Polytechnic University of Hong Kong's multispectral palmprint database (MS PolyU database) [7].

The images contained in the database are very nearly fingerless, and the middle and ring fingers are opened wide due to the presence of a central metallic stud; each image has a size of 352 by 288 pixels.

The database has four directories with a total of 6000 images each, taken from 250 volunteers.

The database gets its name from the volunteers being taken an image of every hand four times, under red, green, blue or infrared illumination.

The only directory used for this work is the "Red" one.

3.2 The General System's Architecture

The system comprises two modular neural networks, one dealing with the images and the other with the geometric measurements, with each of them having a different preprocessing for the whole dataset. At the end, the results of both MNNs are integrated by a FIS, as shown in Fig. 1:

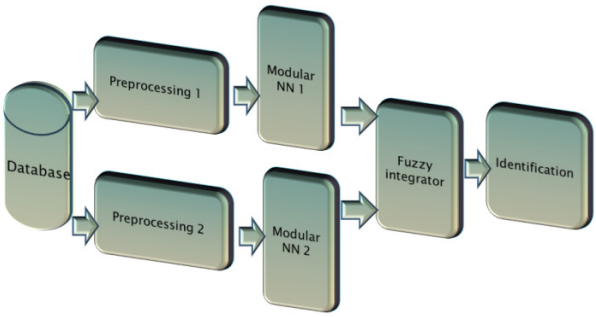


Fig. 1. Complete system's achitecture

3.3 Modular Neural Network 1's Architecture

This MNN is comprised of ten modules, which equally divide their dataset, along with an integrator (winner takes all).

The layout for this MNN is shown in Fig.2:

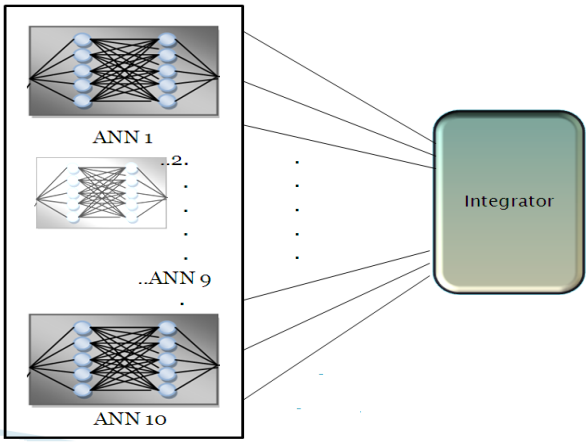


Fig. 2. MNN 1's architecture

3.3.1 MNN 1's Parameters by Module

The parameters for this MNN are as follows (all further networks share these parameters, except the number of neurons):

1. Number of layers: 3
2. Training function: scaled conjugate gradient (though in the first series of trainings for this MNN, it was Fletcher's conjugate gradient).
3. Transference function, layer 1: Tangent sigmoid
4. Transference function, layer 2: Logarithmic sigmoid
5. Transference function, layer 3: Linear
6. Number of neurons, layer 1: 350
7. Number of neurons, layer 2: 175
8. Number of neurons, layer 3: 50

As with all modules, the training set is 58% of the whole dataset.

3.4 Preprocessing for MNN 1

The preprocessing consists (as shown in Fig. 3) of:

1. Equalization of histogram.
2. Edge detection.
3. Reduction of images.
4. Binarizing and standardization of images.

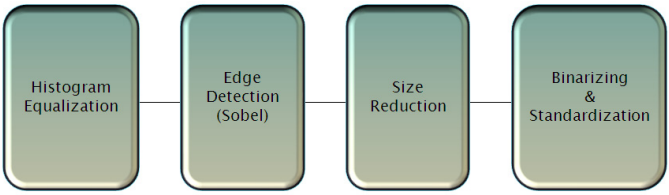


Fig. 3. Block diagram for MNN 1's preprocessing

3.4 Modular Neural Network 2's Architecture

This MNN is comprised of four modules, which equally divide their dataset, along with an integrator (winner takes all).

The layout for this MNN is shown in Fig. 4.

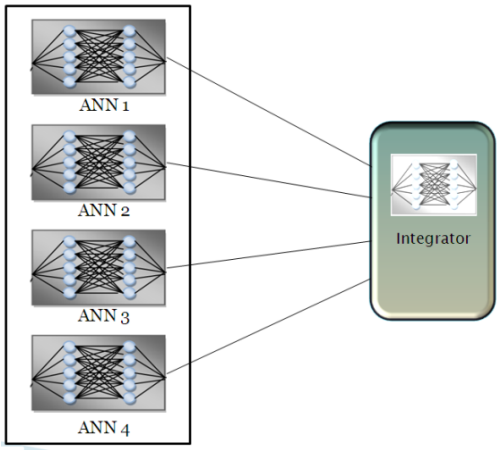


Fig. 4. MNN 2's architecture

3.4.1 MNN 2's Parameters by Module

1. Number of neurons, layer 1: 875
2. Number of neurons, layer 2: for succesive series, 60, 654, 1200
3. Number of neurons, layer 3: 125

(At the beginning, the input to these modules were formatted to Gray code, but it proved unfruitful).

3.4.2 MNN 2's Integrator's Network

When it was evident that results of modules in this MNN were ill-scaled for integration, a NN was developed that could tell which module should be carrying an

individual's identity, by first placing each one in the appropriate fourth of the dataset, and therefore the right module.

This network has the following parameters:

1. Number of neurons, layer 1: 3500
2. Number of neurons, layer 2: 34
3. Number of neurons, layer 3: 4

Success rate for such a net is around 95% and it took some 20 hours of training in a 2.7 MHz dual core machine with 2 Gb of RAM.

3.4.3 Preprocessing for MNN 2

Preprocessing for this modular neural network consists of:

1. Custom filtering image to enhance principal lines.
2. Getting edges of above mentioned image.
3. Getting outline of the whole palm.
4. Finding valleys near index and small fingers, with a process shown in Fig. 5:

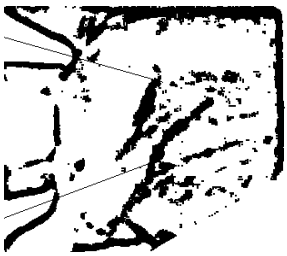


Fig. 5. Finding outer valleys

5. Getting the finger baseline.
6. Getting finger width aided by prior steps plus location of central reference (stud).
7. Starting from fixed points unique to all images, three for each principal line, the nearest points to them, presumably belonging to the principal lines are found and three splines are traced through them. This is necessary because edge detection leaves segments too fragmented.
8. Center of each principal line is located with aid from its spline.
9. Total length of each spline is found.
10. Distance from each "center" to central reference is found.
11. Wrist width is found.
12. Distances from each splines extremes to central reference is found.

Figs. 6 and 7 show an image prior to spline tracing and other, with its spline overlayed (notice the central reference):

Fig. 6. Image of origin

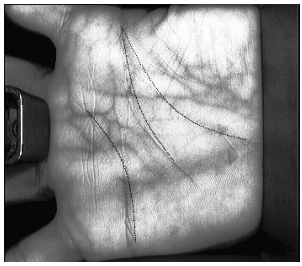


Fig. 7. Overlayed image

Hands that were left were reflected to be right. Preprocessing of the whole dataset took 1.91 hours.

3.5 Fuzzy Integrator

The output from both MNNs is used as input to a fuzzy system to give the final system's response. The integrator is of Mamdani type (Fig. 8):

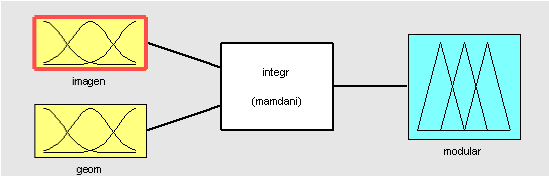


Fig. 8. Fuzzy integrator's architecture

Input variables are "imagen" and "geom", with three membership functions (MF's) each, as shown plotted in Figs. 9 and 10:

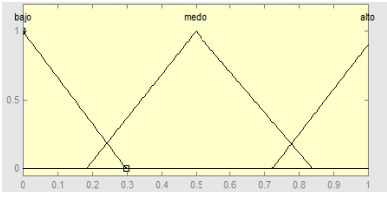


Fig. 9. "imagen" variable

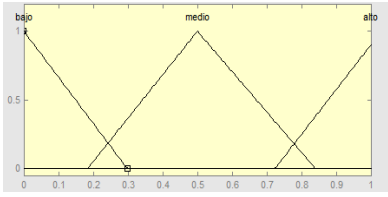


Fig. 10. "geom" variable

Where the MF's "bajo", "medio", "alto" mean "low", "medium", "high". The output variable is "modular" (Fig 11):

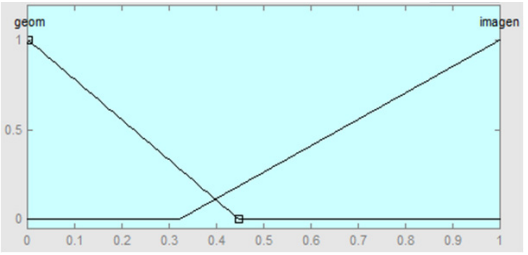


Fig. 11. "modular" variable

3.5.1 Fuzzy Rules

There's a total of nine fuzzy rules:

1. If (imagen is bajo) and (geom is bajo) then (modular is imagen)
2. If (imagen is bajo) and (geom is medio) then (modular is imagen)
3. If (imagen is bajo) and (geom is alto) then (modular is geom)
4. If (imagen is medio) and (geom is bajo) then (modular is imagen)
5. If (imagen is medio) and (geom is medio) then (modular is imagen)
6. If (imagen is medio) and (geom is alto) then (modular is geom)
7. If (imagen is alto) and (geom is bajo) then (modular is imagen)
8. If (imagen is alto) and (geom is medio) then (modular is imagen)
9. If (imagen is alto) and (geom is alto) then (modular is imagen)

3.5.2 Fuzzy System # 2

Another fuzzy system was created starting from the first one, with the same rules, variables and membership functions, the difference being that instead of triangular membership functions, generalized bells were used.

Figs. 12 and 13 graphically show the input variables:

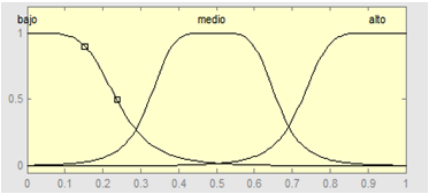


Fig. 12. "imagen" variable

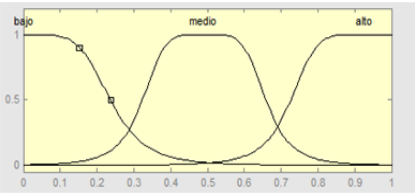


Fig 13. "geom" variable

Fig. 14 shows a plotting of the output variable:

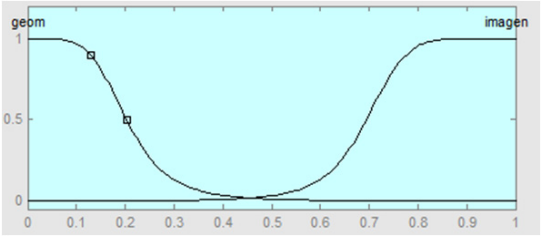


Fig. 14. "modular" variable

4 Results

The following are several tables for series of ten trainings each with the best identification performer marked in blue.

4.1 Results for the MNN 1

Results for the first series are shown in Table 1; the best success rate (SR) was 93.33%.

Table 1. Results for MNN 1, series 1

Training	Σ Epochs	training time	ident. time	identified	success rate
1	4316	41.649 min	18.2500 s	5526	92.10
2	4124	40.190 min	13.8187 s	5527	92.12
3	3887	37.867 min	14.0418 s	5552	92.53
4	3711	35.797 min	14.9982 s	5600	93.33
5	4198	40.662 min	15.1318 s	5527	92.12
6	4087	39.447 min	17.9906 s	5526	92.10
7	4254	41.458 min	14.5147 s	5503	91.71
8	4205	40.583 min	14.6658 s	5517	91.95
9	4076	39.707 min	14.0422 s	5526	92.10
10	4069	39.273 min	14.1105 s	5587	93.11

Results for the second series are shown in Table 2; the best SR was 93.74%.

Table 2. Results for MNN 1, series 2

Training	Σ Epochs	training time	ident. time	identified	success rate
1	4220	40.919 min	20.465 s	5558	92.63
2	4005	39.028 min	13.930 s	5528	92.13
3	3798	36.832 min	14.520 s	5556	92.60
4	3954	38.229 min	15.065 s	5589	93.15
5	4143	40.055 min	16.562 s	5624	93.74
6	4170	38.452 min	19.728 s	5511	91.85
7	4229	41.021 min	14.591 s	5545	92.42
8	4143	40.145 min	14.978 s	5524	92.07
9	4072	39.490 min	14.076 s	5526	92.09
10	4193	40.461 min	15.180 s	5562	92.70

Results for the third series are shown in Table 3; the best SR was 97.73%.

Table 3. Results for MNN 1, series 3

Training	Epochs	training time	ident. time	identified	success rate
1	2151	29.1508 min	4.5072 s	5860	97.67
2	2177	29.1985 min	4.3679 s	5864	97.73
3	2169	28.8461 min	4.6182 s	5861	97.68
4	2181	28.8524 min	4.8882 s	5861	97.68
5	2151	28.5450 min	5.1730 s	5864	97.73
6	2178	28.8729 min	4.3149 s	5856	97.60
7	2180	28.9235 min	4.4670 s	5858	97.63
8	2185	28.9521 min	4.3540 s	5860	97.67
9	2174	28.8521 min	4.3386 s	5860	97.67
10	2113	28.0448 min	4.4143 s	5851	97.52

4.2 Results for the MNN 2

Results for the first series are shown in Table 4; the best SR was 71.74%.

Table 4. Results for MNN 2, series 1

Training	Σ Epochs	training time	ident. time	identified	success rate
1	3422	74.684 min	31.571 s	4280	71.33
2	7897	173.03 min	31.555 s	4057	67.62
3	4079	88.96 min	31.861 s	4304	71.74
4	4955	108.97 min	32.033 s	4071	67.85
5	6017	131.71 min	31.83 s	4087	68.12
6	5762	126.30 min	32.2514 s	4091	68.18
7	4051	88.757 min	32.1057 s	4053	67.55
8	3386	74.188 min	31.729 s	4063	67.72
9	3407	74.687 min	31.4619 s	4078	67.97
10	3846	83.492 min	31.903 s	4029	67.15

Results for the second series are shown in Table 5; the best SR was 77.95%.

Table 5. Results for MNN 2, series 2

Training	Σ Epochs	training time	ident. time	identified	success rate
1	5874	86.820 min	31.779 s	4653	77.550
2	6289	87.635 min	31.785 s	4670	77.833
3	4811	85.611 min	31.924 s	4655	77.583
4	5487	90.340 min	31.932 s	4668	77.800
5	5879	89.005 min	32.041 s	4661	77.683
6	4906	102.528 min	32.178 s	4677	77.950
7	3818	71.473 min	31.917 s	4662	77.699
8	3494	57.437 min	31.595 s	4673	77.883
9	3627	79.091 min	31.682 s	4669	77.817
10	3851	92.051 min	31.974 s	4675	77.917

Results for the third series are shown in Table 6; the best SR was 80.45%.

Table 6. Results for MNN 2, series 3

Training	Σ Epochs	training time	ident. time	identified	success rate
1	4990	250.0487 min	14.6676 s	4813	80.22
2	4857	232.5927 min	14.5411 s	4812	80.20
3	4978	236.7514 min	14.5536 s	4817	80.28
4	4926	240.5805 min	14.6846 s	4809	80.15
5	4970	238.6703 min	14.8786 s	4807	80.12
6	5049	218.5406 min	14.6734 s	4813	80.22
7	4992	222.8399 min	14.6320 s	4810	80.17
8	4979	227.9744 min	14.6869 s	4816	80.27
9	5010	223.3908 min	14.6582 s	4827	80.45
10	4991	224.7490 min	14.9844 s	4814	80.23

4.3 Complete System Results

Results for the first series are shown in Table 7; the best SR was 95.25%.

Table 7. Complete system results, series 1

Training	Σ Epochs	training time	ident. time	identified	success rate
1	7738	116.33 min	45.324 s	5712	95.20
2	12021	213.22 min	44.519 s	5690	94.83
3	7966	126.80 min	45.693 s	5715	95.25
4	8665	144.77 min	45.227 s	5617	93.62
5	10215	172.37 min	47.631 s	5671	94.51
6	9849	165.75 min	45.086 s	5652	94.20
7	8305	130.22 min	44.479 s	5608	93.47
8	7591	114.77 min	47.208 s	5514	91.90

Table 7. (*continued*)

9	7483	114.39 min	46.881 s	5661	94.35
10	7915	122.76 min	44.671 s	5579	92.98

Results for the second series are shown in Table 8; the best SR was 93.949%.

Table 8. Complete system results, series 2

Training	Σ Epochs	training time	ident. time	identified	success rate
1	10094	127.74 min	45.726 s	5561	92.683
2	10294	126.67 min	45.108 s	5528	92.133
3	8609	122.44 min	45.926 s	5553	92.550
4	9441	128.57 min	46.429 s	5588	93.133
5	10022	129.06 min	44.904 s	5637	93.949
6	9076	140.98 min	45.389 s	5516	91.933
7	8047	112.49 min	45.843 s	5543	92.383
8	7637	97.59 min	47.371 s	5539	92.317
9	7699	118.58 min	47.986 s	5527	92.117
10	8044	132.52 min	44.997 s	5562	92.699

Results for the third series are shown in Table 9; the best SR was 97.75%.

Table 9. Complete system results, series 3

Training	Σ Epochs	training time	ident. time	identified	success rate
1	7141	279.1995 min	23.4003 s	5859	97.65
2	7034	261.7912 min	23.1624 s	5865	97.75
3	7147	265.5975 min	23.2537 s	5860	97.67
4	7107	269.4329 min	23.2042 s	5861	97.68
5	7121	267.2153 min	23.1404 s	5864	97.73

Table 9. (*continued*)

6	7227	247.4135 min	22.9807 s	5856	97.60
7	7172	251.7634 min	22.9578 s	5858	97.63
8	7164	256.9265 min	23.9409 s	5860	97.67
9	7184	252.2429 min	23.1345 s	5860	97.67
10	7104	252.7938 min	23.1556 s	5851	97.52

4.4 Complete System Results (2)

Results for the third series are shown in Table 10; the best SR was 97.77%.

Table 10. Complete system(2) results, series 3

Training	**Σ Epochs**	**training time**	**ident. time**	**identified**	**success rate**
1	7141	279.1995 min	23.5901 s	5860	97.67
2	7034	261.7912 min	23.3187 s	5866	97.77
3	7147	265.5975 min	23.5999 s	5861	97.68
4	7107	269.4329 min	24.0719 s	5860	97.67
5	7121	267.2153 min	24.4985 s	5864	97.73
6	7227	247.4135 min	23.3928 s	5858	97.63
7	7172	251.7634 min	23.5359 s	5858	97.63
8	7164	256.9265 min	23.4708 s	5860	97.67
9	7184	252.2429 min	23.3946 s	5860	97.67
10	7104	252.7938 min	23.9661 s	5851	97.52

In all previously shown tables, identification time leaves out preprocessing time.

The tables in section 4 show that the greatest boost in performance when integrating both of the MNNs occurred in series 1, with an advantage of some 2.7 % over MNN 1, which, as readily seen always carries most of the weight in finding the correct identifications, and it happens to be the non-geometric MNN.

What happens in later series is easy to explain, even though both MNNs get to perform better: since MNN 2 doesn't improve as much as MNN 1, there is close to no net gain in using MNN 2, and in fact, some times there is a loss; this is only marginally better with FIS #2, which gave the best overall performance, at 97.77%.

A few comparisons with prior works in the same (or close) area of research can be very illustrative:

Zhenhua Guo uses the same database we do, but using all spectra at the same time; he uses a fusion algorithm based on Haar wavelets and PCA, and on one of many configurations tried, obtained 97.877 as a success rate [1].

Kumar and Zhang use entropy based discretization for a database smaller than ours and use Support vector machines and neural networks as classifiers for a respective accuracy of 95 and 94% identifying, for a hand geometry method [4].

Cenker Öden's group claims that their method, using implicit polynomials, is capable of achieving 95% accuracy in identification tasks [6].

It is noteworthy that, when dealing with hand geometry systems in verification mode, Ingersoll Rand's ID3D system is reported by Sandia Labs as performing as low as 0.2% of total error, this as far back as 1991 [9].

5 Conclusions and Future Work

As seen in the last section, the best overall performance so far obtained with our method is 97.77%.

Performance, compared to prior works does not seem too bad, but as a whole, comparisons show that there is still much to be improved and open a few lines of future work.

First, trying to elevate performance of the net within MNN 2's integrator; that would gain no more than a few tens of well identified individuals over MNN 1 alone.

Second, the problem might reside in the arrangement of modules (their number) in MNN 2.

Third, a true verification comparison should take into account that MNN 1's modules on their own have a success rate at least two percentage points higher than MNN 1 itself.

Fourth, comparison with other systems, including verification, would be aided by performing complete statistical evaluations of the system.

References

[1] Guo, Z.: Online Multispectral Palmprint Recognition. Thesis, Hong Kong Polytechnic University (2009)

[2] Jang, R., Tsai-Sun, C., Mizutani, E.: Neuro-Fuzzy and Soft Computing, 1st edn. Prentice Hall, Upper Saddle River (1996)

[3] Kong, A., Zhang, D., Kamel, M.: A survey of palmprint recognition. In: Pattern Recognition, vol. 42, pp. 1408–1418. Elsevier (2009)

[4] Kumar, A., Zhang, D.: Biometric recognition using entropy-based discretization. In: Proceedings of 2007 IEEE International Conference on Acoustics Speech and Signal Processing, pp. 125–128 (2007)
[5] Luger, G., Stubblefield, W.: Artificial Intelligence, 3rd edn. Structures and strategies for complex problem solving. Addison Wesley, Reading (1998)
[6] Öden, C., Erçil, A., Yldz, V.T., Krmzta, H., Büke, B.: Hand Recognition Using Implicit Polynomials and Geometric Features. In: Bigun, J., Smeraldi, F. (eds.) AVBPA 2001. LNCS, vol. 2091, pp. 336–341. Springer, Heidelberg (2001)
[7] PolyU Multispectral Palmprint Database, `http://www4.comp.polyu.edu.hk/Ebiometrics/MultispectralPalmprint/MSP_files` (accessed October 2011)
[8] Rojas, R.: Neural Networks - A Systematic Introduction. Springer, New York (1996)
[9] Zunkel, R.: Hand geometry based authentication. In: Jain, A.K., Bolle, R., Pankanti, S. (eds.) Biometrics: Personal Identification in Networked Society. Kluwer Academic Publishers, Boston (1998)

Development of an Automatic Method for Classification of Signatures in a Recognition System Based on Modular Neural Networks

Verónica Carrera, Patricia Melin, and Diana Bravo

Tijuana Institute of Technology
pmelin@tectijuana.mx

Abstract. This paper presents the development of an automatic method for classification of signatures according to a geometric shape recognition system based on a modular neural network (MNN). To carry out the method of automatic classification, we apply pre-processing to our database which consists of 30 individuals, first the image goes to a high pass filter, which allows the passage of signals depending on their frequency and finally we apply the Fourier transform, which is essentially a wave phenomenon which serves to measure the distribution of amplitudes of the frequency of our image (signatures), and presents to certain extent as they are, rise time, peak parking. Thus the signatures are automatically sorted to the module that corresponds in the Modular Neural Network, which contains three separate modules, each one uses different feature extraction methods: edge extraction, wavelet transform and Hough transform, where this results in the identification or recognition of signatures.

1 Introduction

Humans have an ability to recognize patterns that we use constantly without realizing it. The skill with which the human mind operates to distinguish the characteristics of a pattern led to the need to understand how it works and try to imitate.

Artificial neural networks, which are mathematical models inspired by biology, are able to extract the relevant features (sometimes hidden) of a set of data, to learning of their properties and then apply them to recognize a new element, have been proposed in different works [6].

There has been as always a constant search for new ways to improve their living conditions. These efforts have served to reduce the work in the operations in which force plays a major role. The progress achieved has allowed directing these efforts to other fields, such as the construction of calculating machines to help resolve automatically and quickly certain operations that are tedious when done by hand.

Neural networks are nothing more than an artificial and simplified model of the human brain, which is the most perfect example we have for a system that is

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 201–210.
springerlink.com

capable of acquiring knowledge through experience. A neural network is "a new system for the treatment of information processing, whose basic unit is inspired by the fundamental cell of the human nervous system: the neuron" [6].

In this paper, we present an automatic method for classification of signatures according to a geometric shape recognition system based on a modular neural network (MNN). We decided for an MNN, as they have proven to be a powerful, robust and flexible, useful in many pattern recognition problems [1]. The proposed automatic classification method is based on applying a pre-processing step and application of the Fourier transform to analyze their frequency and amplitude of the signatures in order to determine which of the three modules belong, each one of which receives a specific characteristic of the signatures image as its input, which include edges of the image, the wavelet transform and the Hough transform.

The neural network will result in a very precise discrimination of the input data used in the experimental evidence and we have confirmed that this system can solve a difficult biometric problem with a MNN with a simple set of image features. This paper is organized as follows.

In section 2, a brief introduction of modular neural networks is presented, in Section 3, we explain the extraction of features such as: the detector edges, Wavelet Transform and the Hough transform used in the recognition system and Section 4 shows the proposed method, which consists in pre-processing and application of the Fourier transform for the automatic classification of signatures. The experimental results are presented in Section 5 and finally conclusions in Section 6.

2 Modular Neural Networks

Modular neural networks have a hierarchical organization, to cover several artificial neural networks; its architecture basically consists of two main components: local experts and an integration unit. To simplify this, we can call Modular Neural Network a collection of artificial neural networks and the variations that exist between them. The central idea is the decomposition of tasks; by using a modular type network, a given task is divided among various local experts neural networks, the average load in each neural network module is reduced compared with the use of a network simple neural learning to be completely original task, and so the combined model tends to exceed the limitations of a simple neural network [5].

Recent advances in neurobiological sciences have strengthened the belief of the existence of modularity at different levels of functional and structural brain. The concept of modularity is an extension of the principle "Divide and Conquer", this principle has no formal definition, is an intuitive way by which a complex task, in this case the computation can be divided into simpler subtasks. The solution to the overall task is achieved by combining the individual results of local computer systems [1].

3 Feature Extractors

There are three main reasons why we use the feature extractor; the first is that it reduces the computational complexity of classification algorithms to work on a lower dimensional space. The second reason, statistical estimation methods become more reliable in a reduced dimensional space. The third and final reason, the possibility that the space dimension of the features does not exceed three, to allow a graphical display of the classes involved [1].

3.1 Edge Detector

The edges generally limit the objects and can be useful in the segmentation and identification of objects in scenes [8]. Edge detection is a binarization, where a binary image is an image where each pixel can have one of only two possible values 0 or 1, as might be expected, in an image of these conditions is much easier to find and/or distinguish structural features. For purposes of achieving a certain standardization regarding the spatial parameters of the two-dimensional images and their content, the technique known as geometric center is used.

3.2 Wavelet Transform

It is a transformation of the image and is divided in two types of smaller images, the trend and fluctuations. The trend is to be a copy of the original image at lower resolution and fluctuations stored information related to local changes in the original image. The trend and most significant fluctuations allow a compression of the image to discard irrelevant information exchange and removal of noise. Studying the trend and fluctuations allows, among other things, the comparison with patterns to detect forms in an image [7].

3.3 Hough Transform

This is a technique to discover ways in an image. It's based on trans-forming image points in a parameter space. The idea is to find curves parameterized as lines, circles and polynomials. Generally edge detection is performed to the image, and then this transform is applied. Thus fewer points to go and therefore the faster the algorithm [4].

4 Proposed Method

The problem we address in this paper is concerned with the automatic classification of the signature according to its geometric shape and the automatic recognition of it. It is performed using the database acquired by students and faculty of the Graduate Program in Computer Science Technological Institute of Tijuana, Mexico. Where 12 samples were taken of the signature of 30 different people,

giving a total of 360 images. The system is trained with 7 samples of the signatures of each person and then test with the other. This means that the total number of 210 images is used for training and the number of images for testing are 150.

This paper proposes a method of automatic classification according to the shape of the signature, in a recognition system based on a modular neural network, which consists of 3 modules. The system consists of three stages, which are, pre-processing the images, automatic sorting of signatures and feature extraction (Fig. 1).

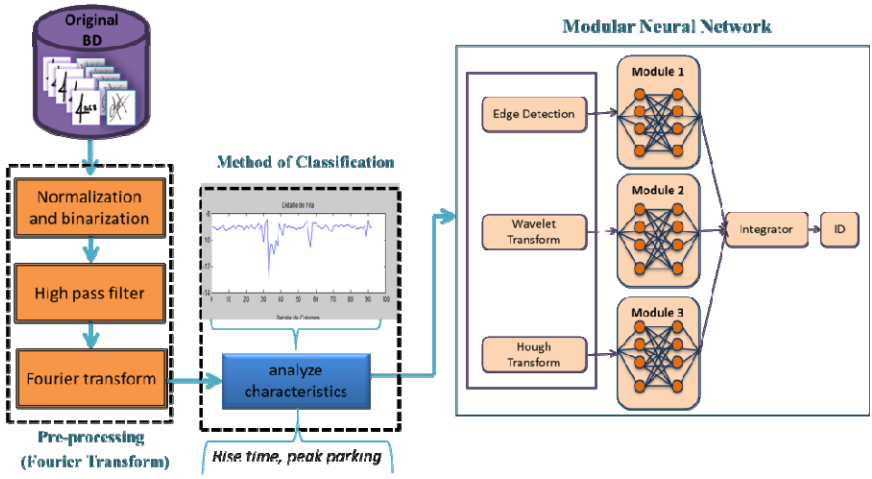


Fig. 1. Proposed system architecture for automatic classification and recognition of the signature

4.1 Pre-processing

For pre-processing is basically a normalization, which is performed to make standard size dimensions of an image, the image without causing significant distortion. It binarizes images for only two colors: black and white or binary numbers, 0 and 1 (Fig. 2).

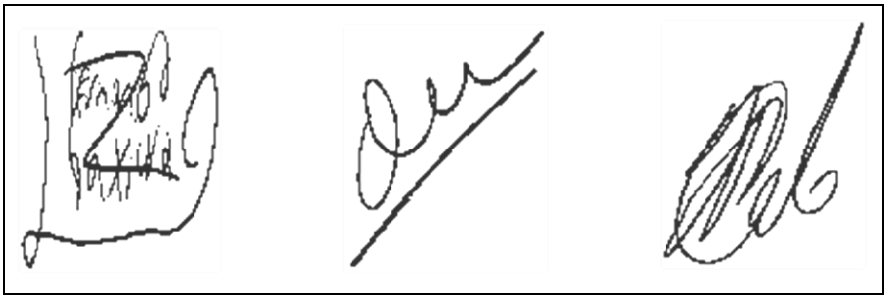


Fig. 2. Image binarization

We applied a high pass filter to the signatures with a cutoff frequency of 130 and show the sample of the filter transfer function, as shown in Fig. 3. This filter is used to allow the passage of signals depending on their frequency, and it can be considered as circuit which attenuates all signals whose frequency is below a specified cutoff frequency and allows the passage of the frequencies above the cutoff frequency.

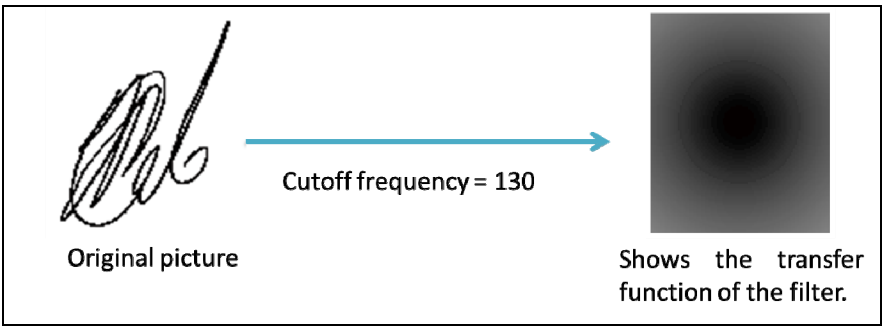


Fig. 3. Applying high-pass filter with cutoff frequency of 130

4.2 Method of Classification

After we did perform the pre-processing the images, then a Fourier transform was applied to obtain the frequency and amplitude of the signatures as shown in Fig. 4., in order to analyze their characteristics and automatically sort them according to their geometric shape.

The Fourier Transform is basically the frequency spectrum of a function. A good example is what the human ear and auditory receiving a wave and transforms it into decomposition into different frequencies (which is what finally be heard).

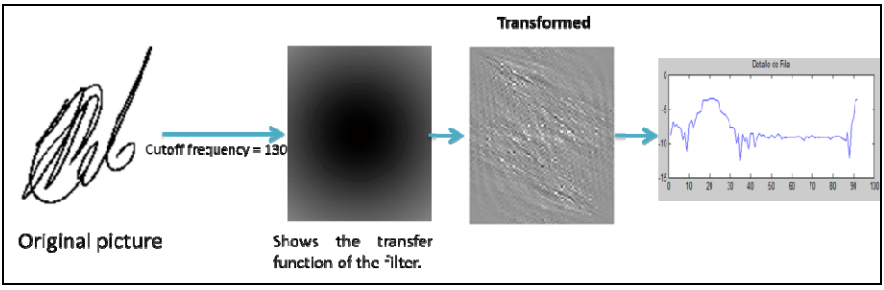


Fig. 4. Applying Fourier transform to the signature

By analyzing the frequency and amplitude of the signatures we chose to include conditions, which help us to classify signatures automatically in three different types, and then are entered into the modular neural net-work with three modules, which each has a feature extractor, Edge Detector, Hough Transform and Wavelet

Transform, which will serve for the recognition of the signature. For the automatic classification method we did as follows; values were added and averaged per column, storing the result in a vector. We tried to make the minimum rate is zero and the maximum estimate 108 using this so that the values obtained are positive and there are no problems when adding up the values. We check what the maximum and minimum frequencies were and store the values and locations in a Variable called max and min. After measuring the highest amplitude and lowest frequency, we were being able to obtain the current and previous position of these amplitudes.

The following conditions were used to classify signatures automatically. For signatures oval or circular lz1 we declare the variable which represents our first list of signatures that meet these conditions;

if frec > 39 and frec < 64 or frec > 41.1 and frec < 44.5

if amp > 3.9 and amp <4.35

For some signatures, we declare the inclined variable LZ2 which repre-sents our second list of signatures that meet these conditions;

Frec < 40 or frec > 69

Amp > 2.9 and amp > 9.17

For handwritten signatures LZ3 declare the variable is represented as follows;

LZ3 = 30 - (lz1 + LZ2)

The classification method gives results in 11 signatures for the first group, 6 signatures for the second group and 13 for the third (Fig. 5).

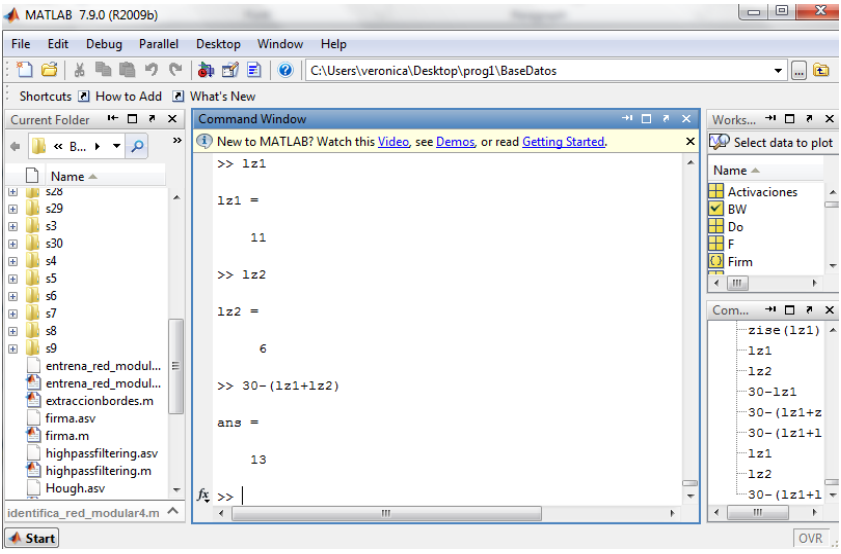


Fig. 5. Number of signatures classified in each group

4.3 Feature Extractor

In this paper, we have three modules, and each has different characteristics of images taken from the original image of the signatures of a person. Each of these feature extraction methods is described briefly below.

4.3.1 Edge Detector

For images of handwritten signatures, the edges can capture much of its structure, because people tend to write with a single color on a white background. Therefore, we have chosen to apply the Canny edge detector to each image to enter the corresponding module as in Fig. 6. , where it can be noted the edge detection process applied to a particular signatures database.

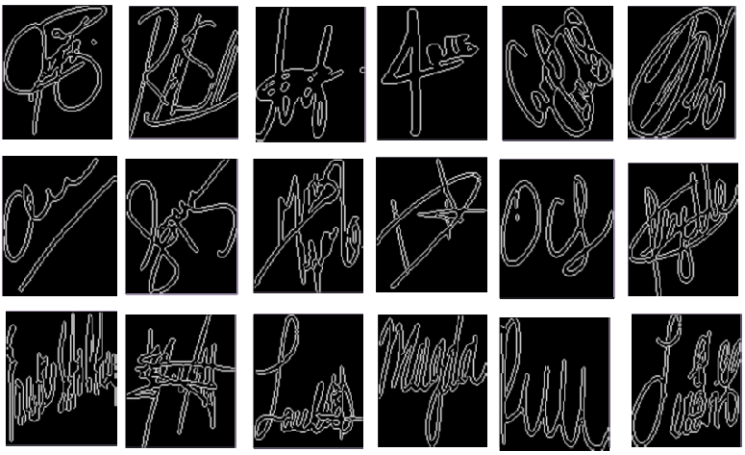


Fig. 6. Signatures with the Edge Detector

4.3.2 Wavelet Transform

The wavelets are a powerful tool to address fundamental problems in image processing. These include reduced noise, compression (of vital importance in the transmission of large amounts of data as in storage) or the detection of certain objects in certain types of images. Some of the main problems affecting the processing of digital images are the compression of data for subsequent storage or transmission, noise removal, contrast enhancement and texture analysis [2]. We apply the wavelet transform to some signatures as shown in Fig. 7.

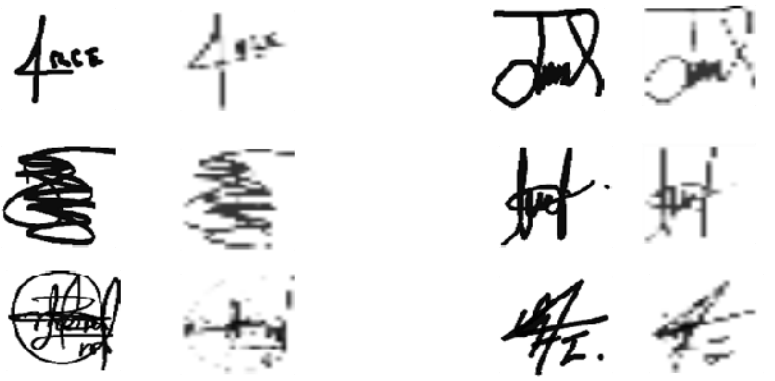


Fig. 7. Using Wavelet Transform in signatures

4.3.2 Hough Transform

This transformation can extract line segments of the image. After obtaining a transformation matrix of the original images, this process depends on an important parameter "threshold", which provides the best value possible after conducting some experiments. Demonstration the use of the Hough transform in the image (Fig. 8.).

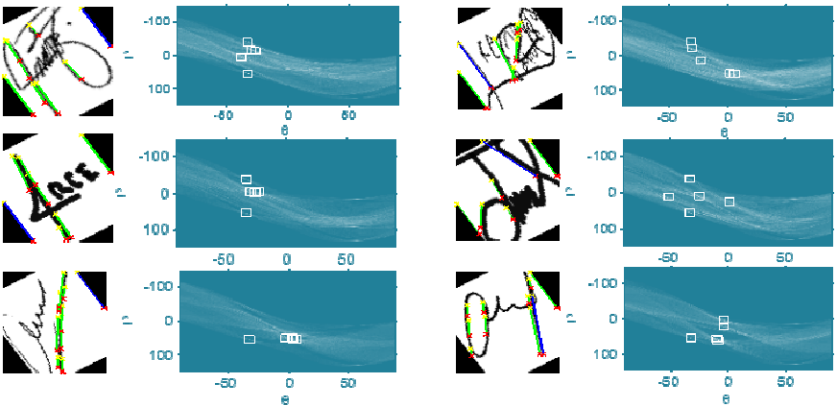


Fig. 8. Implementation of the Hough transform in signatures

5 Experimental Results

In the following we show some results and evidence obtained during the development and testing of this work, for the automatic classification of signatures in a

recognition system based on the neural network. Tests were performed using modular neural network, to see how the network behaved and what was the percentage of recognition, that without feature extraction methods. Ten experiments with the same parameters were performed for Module1.

Table 1 Shows the parameters used in 10 experiments that were done with our Modular Neural Network for the recognition of the signatures.

Table 1. Parameters to be trained with the Network

	Parameters					
experiment	show	goal	lr	epochs	time	type
1	60	0.001	0.161	100	00:00:52	traingdx
2	60	0.001	0.161	100	00:00:53	traingdx
3	60	0.001	0.161	100	00:00:53	traingdx
4	60	0.001	0.161	100	00:00:52	traingdx
5	60	0.001	0.161	100	00:00:55	traingdx
6	60	0.001	0.161	100	00:00:50	traingdx
7	60	0.001	0.161	100	00:00:50	traingdx
8	60	0.001	0.161	100	00:00:52	traingdx
9	60	0.001	0.161	100	00:00:51	traingdx
10	60	0.001	0.161	100	00:00:51	traingdx

Table 2 shows the results of our Modular Neural Network for the recognition of signatures, using a database consisting of 30 persons with 12 images each. Five images for recognition, leading to better results in the experiment number 6 with a rate of 100%.

Table 2. Results to test the Network

experiment	identified	error	percentage	time
1	139	11	92.666667	00:02:01
2	134	16	89.333333	00:02:02
3	131	19	87.333333	00:02:20
4	145	5	96.666667	00:02:03
5	133	17	88.666667	00:02:08
6	150	0	100	00:02:03
7	144	6	96	00:01:56
8	104	46	69.333333	00:02:04
9	134	16	89.333333	00:02:04
10	131	19	87.333333	00:02:25

6 Conclusions

In this paper a method of automatic classification of signatures was proposed, which can integrate the modular neural network, this modular neural network consists of three modules; the first module uses a edges detector preprocessing, the second one the wavelet transform and the third one the Hough transform. In developing the method of classification, we could really tell that it classifies signatures according to their shape, in this particular case study gives a ranking for the first module 11 signatures for the second 6, and for the third module 13. At this stage more tests are underway, in the classification method with modular neural network, to obtain a better result in the recognition of the signature.

References

[1] Beltrán, M., Melin, P., Trujillo, L.: Modular Neural Networks with Fuzzy Response Integration for Signature Recognition. Bio-inspired Hybrid Intelligent Systems for Image Analysis and Pattern Recognition, 81–91 (2009)

[2] Dong, W., Jun, T., Raymond, O., Sidney, B.C.: A new class of biorthogonal wavelet systems for image transforms coding. IEEE Transactions on Image Processing 7(7), 1000–1013 (1998)

[3] Gorunescu, F., Gorunescu, M., Saftoiu, A., Vilmann, P., Belciug, S.: Competitive/collaborative neural computing system for medical diagnosis in pancreatic cancer detection. Expert Systems 28(1), 33–48 (2011)

[4] Melin, P., Castillo, O., Gonzalez, S., Cota, J., Trujillo, W.L., Osuna, P.: Design of Modular Neural Networks with Fuzzy Integration Applied to Time Series Prediction. In: Analysis and Design of Intelligent Systems using Soft Computing Techniques, pp. 265–273 (2007)

[5] Palmer, P., Montaño, A.: What are neural networks? Applications made in the field of addictions, pp. 243–255. Department of Psychology, University of the Balearic Islands (2005)

[6] Queipo, N.V., Zerpa, L.E., Goicochea, J.V., Verde, A.J., Pintos, S.A., Zambrano, A.: A model for the integrated optimization of oil production systems. Eng. Comput. (Lond.) 19(2-3), 130–141 (2003)

[7] Ruz, G.A., Estévez, P.A., Ramírez, P.A.: Automated visual inspection system for wood defect classification using computational intelligence techniques. Int. J. Systems Science 40(2), 163–172 (2009)

[8] Salvador, E., Luca, F.G.: Interactive Reception Desk with Face Recognition-Based Access Control. In: ICDSC 2007, pp. 140–146 (2007)

Architecture of Modular Neural Network in Pattern Recognition

Manuel Leobardo Zavala-Arriaza, Fevrier Valdez, and Patricia Melin

Tijuana Institute of Technology, Tijuana, Mexico
fevrier@tectijuana.mx

Abstract. This paper introduces on architecture of a modular neural network (MNN) for pattern recognition, more recently, the addition of modular neural network techniques theory have been receiving significant attention. The design of a recognition system requires careful. The paper also aims to use the architecture of this Modular Neural Network for pattern recognition in order to optimize the architecture, and used an integrator that will get a good percentage of image identification and in the shortest time possible.

Keywords: Image face recognition, Modular Neural Network.

1 Introduction

Pattern recognition is the study of how machines can observe the environment, learn to distinguish patterns of interest from their background, and make sound and reasonable decisions about the categories of the patterns. The best pattern recognizers in most instances are humans, yet we do not understand how humans recognize patterns. This is hopeful news to proponents of artificial intelligence, since computer can surely be taught to recognize patterns [1]. In machine learning, pattern recognition is the assignment of a label to a given input value. An example of pattern recognition is classification, which attempts to assign each input value to one of a given set of classes (for example, determine whether a given email is "spam" or "non-spam"). However, pattern recognition is a more general problem that encompasses other types of output as well.

2 Pattern Recognition

The recognition, description, classification, and grouping of patterns are important problems in a variety of engineering and scientific disciplines such as biology, psychology, medicine, computer vision, artificial intelligence, and remote sensing [2]. A pattern could be a fingerprint image [3], a handwritten cursive word [4], a human face [5], ear [6], eyes [7], hands [8] and character recognition [9]. Within medical science, pattern recognition is the basis for computer-aided diagnosis

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 211–219.
springerlink.com

(CAD) systems. CAD describes a procedure that supports the doctor's interpretations and findings. Pattern recognition is generally categorized according to the type of learning procedure used to generate the output value. Supervised learning assumes that a set of training data (the training set) has been provided, consisting of a set of instances that have been properly labeled by hand with the correct output.

3 Artificial Neural Networks

An Artificial Neural Network (ANN), usually called neural network (NN), is a mathematical model or computational model that is inspired by the structure and/or functional aspects of biological neural networks. A neural network consists of an interconnected group of artificial neurons, and it processes information using a connectionist approach to computation. In most cases an ANN is an adaptive system that changes its structure based on external or internal information that flows through the network during the learning phase.

Once we are modeling an artificial functional model from the biological neuron, we must take into account three basic components. First all, the synapses of the biological neuron are modeled as weights. Let's remember that the synapse of the biological neuron is the one which interconnects the neural network and gives the strength of the connection. For an artificial neuron, the weight is a number, and represents the synapse. A negative weight reflects an inhibitory connection, while positive values designate excitatory connections. The following components of the model represent the actual activity of the neuron cell. All inputs are added altogether and modified by the weights. This activity is referred as a linear combination. Finally, an activation function controls the amplitude of the output. [10]This process is described in the Fig.1.

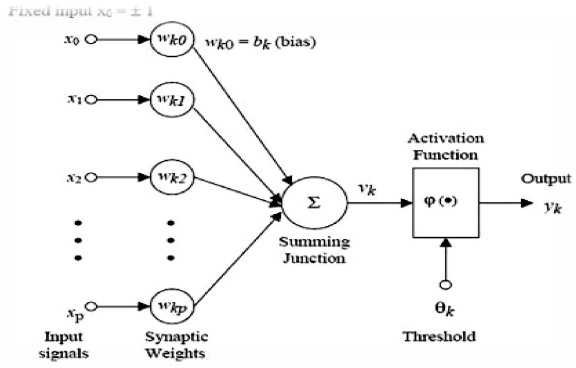


Fig. 1. Artificial neural networks how to use

From this model the interval activity of the neuron can be defined as:

$$V_k = \sum_{j=1}^{p} W_{kj} X_j \quad (5)$$

The output of the neuron, yk, would therefore be the outcome of some activation function on the value of vk.

4 Methodologies

The methodologies used in this work are:

1. Obtain a database.
2. Determining division of the database in terms of individuals per module the modular neural network.
3. Find methods and/or pre-processing techniques for application to image database and get a better identification.
4. Programming in Matlab the modular neural network architectures consist of three modules.
5. Find a modular integration method that provides the desired results.
6. Optimize the architecture of the MNN using the ACO algorithm.
7. Compare results obtained with previous approaches.
8. Document the investigation.

4.1 Problem Description

We use a database for images provide by Dr. Libor Spacek, Department of Computer Science, and University of Essex. Wivenhoe Park. Description of the Collection of Facial Images [8]:

- Total number of individuals: 395
- Number of images per individual: 20
- Total number of images: 7900
- Gender: contains images of male and female subjects
- Race: contains images of people of various racial origins
- Age Range: the images are mainly of first year undergraduate students, so the majority of indivuals are between 18-20 years.
- Glasses: Yes
- Beards: Yes
- Image format: 24bit colour JPEG
- Camera used: S-VHS camcorder
- Lighting: artificial and fluorescent overhead

4.1.1 Different Facial Movements

- Example of a persons with twenty images with different facial movements, which are trained with the modular neural network, which is shown in Fig 2.

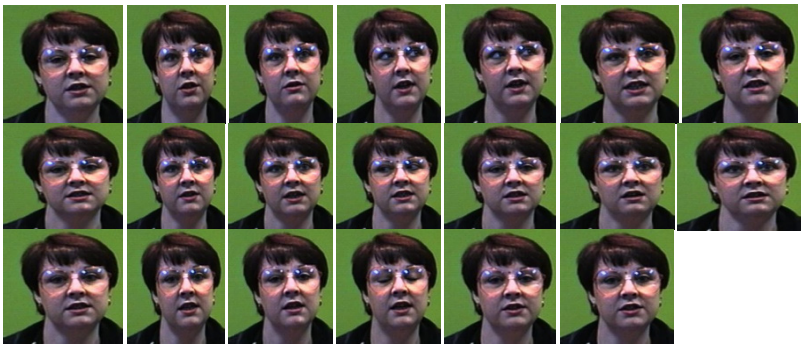


Fig. 2. One person to different facial movements

4.2 Modular Neural Network (MNN)

We work a face database. There are 20 images per person, a total of 395 people, and a total of 7900 images. The dimension of images is the 180 x 200. In this work is applied to recognition at persons using modular neural network architecture. We use in three modules, the first module using 132 person (1848 images for training – 792 images for test), second module 132 person (1848 images for training – 792 images for test), third 131 person (1834 images for training – 786 images for test), with first hidden layer 292, second hidden layer 292 and layer output 132, being as in Fig 3.

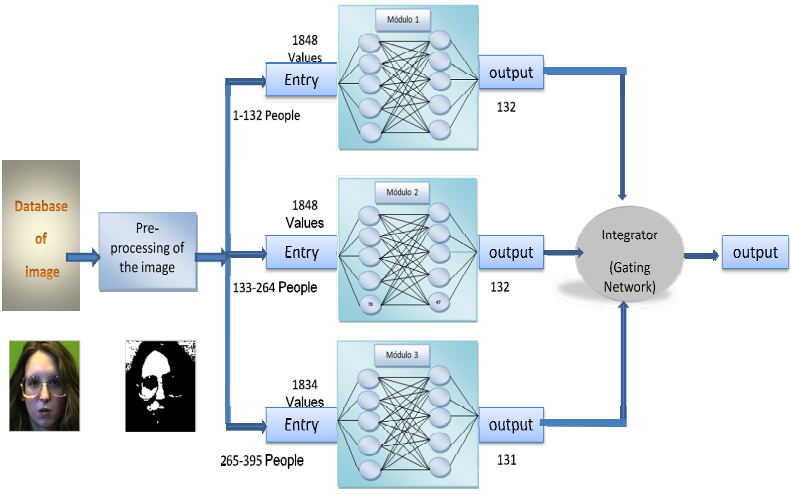


Fig. 3. Modular neural network architecture

4.3 Experimental Result of the MNN

The experimental results were obtained with the determined MNN architecture determined, and these experiments with three types of learning algorithms: gradient descent momentum and an adaptive learning rate, gradient descent with adaptive learning rate, scaled conjugate gradient method. The following results were obtained for each the three modules in terms of percentage of identification.

4.2.1 Module1

In this module the best result was in run number seven with a percentage of identification 94.570707 in a time of 0:36:25 second using the gradient descent momentum and an adaptive learning rate algorithm, Show in Table 1.

Table 1. Experimental result module1

#	goal	seasons	1 layer	2 layer	3 layer	time	Total identified	% identified
1	0.000001	375	290	290	132	0:35:19	740/792	93.434343
2	0.000001	376	291	291	132	0:37:09	737/792	93.055556
3	0.000001	377	292	292	132	0:35:36	744/792	93.939394
4	0.000001	378	293	293	132	0:34:12	747/792	94.218182
5	0.000001	379	294	294	132	0:48:32	739/792	93.308081
6	0.000001	380	295	295	132	0:34:15	738/792	93.181818
7	**0.000001**	**381**	**296**	**296**	**132**	**0:36:25**	**749/792**	**94.570707**
8	0.000001	382	297	297	132	0:37:28	718/792	90.656566
9	0.000001	383	298	298	132	0:36:22	728/792	91.919192
10	0.000001	384	299	299	132	0:34:44	726/792	91.666667
11	0.000001	385	300	300	132	0:36:50	730/792	92.171717
12	0.000001	386	301	301	132	0:37:14	749/792	94..570707
13	0.000001	387	302	302	132	0:41:12	744/792	93.939394
14	0.000001	388	303	303	132	0:39:42	735/792	92.803030
15	0.000001	389	304	304	132	0:38:31	730/792	92.171717

4.2.2 Module2

In this module the best result was in run number seven with a percentage of identification 99.242424 in a time of 1:26:18 second using the scaled conjugate gradient method algorithms, Show in Table 2:

Table 2. Experimental result module2

#	goal	seasons	1 layer	2 layer	3 layer	time	Total identified	% identified
1	0.000001	375	290	290	132	1:19:07	702/792	88.636364
2	0.000001	400	290	290	132	1:19:32	768/792	96.969697
3	0.000001	425	290	290	132	1:25:10	696/792	87.878788
4	0.000001	450	290	290	132	1:28:18	786/792	99.242424
5	0.000001	475	290	290	132	1:38:04	726/792	91.666667
6	0.000001	500	290	290	132	1:36:17	774/792	97.727273
7	**0.000001**	**525**	**290**	**290**	**132**	**1:26:18**	**786/792**	**99.242424**
8	0.000001	550	290	290	132	2:35:11	744/792	93.939394
9	0.000001	575	290	290	132	2:02:14	744/792	97.727273
10	0.000001	600	290	290	132	1:53:03	744/792	93.939394
11	0.000001	625	290	290	132	1:43:15	786/792	99.242424
12	0.000001	650	290	290	132	3:20:07	780/792	98.484848
13	0.000001	675	290	290	132	2:02:49	780/792	98.484848
14	0.000001	700	290	290	132	2:06:26	744/792	93.939394
15	0.000001	750	290	290	132	2:21:20	780/792	98.484848

4.2.3 Module3

In this module the best result was in run number five with a percentage of identification 100 in a time of 00:30:52 second using the gradient descent with adaptive learning rate algorithms, show in Table 3:

Table 3. Experimental result module 3

#	goal	seasons	1 layer	2 layer	3 layer s	time	Total identified	% identified
1	0.000001	375	290	290	131	0:39:28	786/786	100
2	0.000001	400	291	291	131	0:38:55	786/786	100
3	0.000001	425	292	292	131	0:38:49	786/786	100
4	0.000001	450	293	293	131	0:43:50	774/786	98.473282
5	**0.000001**	**475**	**294**	**294**	**131**	**0:30:52**	**786/786**	**100**
6	0.000001	500	295	295	131	0:49:52	786/786	100
7	0.000001	525	296	296	131	0:41:43	786/786	100
8	0.000001	550	297	297	131	0:39:16	786/786	100
9	0.000001	575	298	298	131	0:45:13	774/786	98.473282
10	0.000001	600	299	299	131	0:44:17	778/786	98.473282

Table 3. (*continued*)

11	0.000001	625	300	300	131	0:34:24	778/786	100
12	0.000001	650	301	301	131	0:47:11	778/786	100
13	0.000001	675	302	302	131	0:44:13	778/786	100
14	0.000001	700	303	303	131	0:42:08	778/786	100
15	0.000001	725	304	304	131	0:54:50	778/786	100

4.2.4 Integrator

Taking the results of each module now the integration of all modules is performed, taking into account that in these three modules the learning algorithm of gradient descent with adaptive learning result the most efficient of all. The integration we use is Gating Network for our work [9] with identification of 97.93771% (2321/2370).

5 Comparison with Previous Work

In Table 4, we can observe the behavior of experiments with the Essex database in the different research paper. Also a comparison was made with the method that was developed in this research work. In this research is being compared with our modular neural network with another investigation reference is shown in the table 4, in position number 15 the propose method with MNN is shown and the percentage identification is **97.93 %**.

Table 4. Experimental use the database Essex in different work.

Position	method	Database	Individuals/image	To train	To identify	identification
1	NMF [11]	Essex (Face-94)	50/10	150 (30%)	350 (70%)	93.38 %
2	LNMF [11]	Essex (Face-94)	50/10	150 (30%)	350 (70%)	94.95 %
3	SFNMF [11]	Essex (Face-94)	50/10	150 (30%)	350 (70%)	97.58 %
4	RNM P[]	Essex (Face-94)	50/10	150 (30%)	350 (70%)	97.14 %
5	SVM [12]	Essex (Grimace)	18/20	216 (66%)	144 (34%)	98.13 %
6	RNM P[]	Essex (Grimace)	18/20	216 (66%)	144 (34%)	95.83 %
7	Eigen face [13]	Essex (Grimace)	18/20	144 (34%)	216 (66%)	69.40 %
8	W + PCA [13]	Essex (Grimace)	18/20	144 (34%)	216 (66%)	98.50 %

Table 4. (*continued*)

9	CB PCA [13]	Essex (Grimace)	18/20	144 (34%)	216 (66%)	100 %
10	RNM P[]	Essex (Grimace)	18/20	144 (34%)	216 (66%)	95.13 %
11	RNM []	Essex	195/20	2730 (70%)	1170 (30%)	91.58 %
12	RNM P[]	Essex	195/20	2730 (70%)	1170 (30%)	91.90 %
13	RNM[]	Essex (Complete)	389/20	4668 (60%)	3112 (40%)	85.49 %
14	RNM P[]	Essex (Complete)	389/20	4668 (60%)	3112 (40%)	84.67 %
15	**MNN**	**Essex (Face-94)**	**395/20**	**6320 (70%)**	**2370 (30 %)**	**97.93 %**

In the Table above we can see the results generated by the developed method compared to other studies that were performed with the data bade Essex. In the first 3 rows on the Table shows the methods: NMF (Non-Negative Matrix Factorization), LNMF (Local NMF) and SFNMF (Spatially Confined). Subsequently there is a variant of the method developed which is RNM P (Modular Neural network Parallel). And the record is show to method SVM (Support Vector Machine) and RNM P where you can see a slight improvement of the method SVM. The following methods in the table are: Eigen face, W (Wavelet) + PCA (Principal Component Analysis) and CB PCA (Curve let Based PCA).

6 Conclusions

In this paper we work with the modular neural network for pattern recognition using the Essex data base, which is working with 395 people with 20 images of people with a total of 7900 images, working with three training modules and different algorithms training. Unlike mentioned in the Table IV at position 14 they working with database of Essex with 389 people, but we work with the database of 395 people, and the position number 9 on total identification was 100%, but with few images and our research work which resulted a total of 97.93% but with a higher number of images.

References

[1] Hopfield, J.J.: Artificial neural network. IEEE Circuits and Devices Magazine (1988)
[2] Jain, A.K., Mao, J., Mohiuddin, K.M.: Artificial Neural Networks: A Tutorial. Computer, 31–44 (March 1996)
[3] Nagaty, K.A.: Fingerprints classification using artificial neural networks: a combined structural and statistical approach. Neural Networks 14(9), 1293–1305 (2001)

[4] Blumenstein, M., Verma, B.: A Neural Based Segmentation and Recognition Technique for Handwritten Words. School of Information Technology. Griffith University, Gold Coast Campus, Qld 9726, Australia
[5] Ahmad Fadzil, M.H.: Huma face recognition using neural networks. In: IEEE International Conference, November 13-16 (1994)
[6] El-Bakry, H.M., Mastorakis, N.: Ear Recognition by using Neural Networks. In: Faculty of Computer Science & Information Systems. Mansoura University, EGYPT
[7] Patricia, M.: Reconocimiento de personas basado en medida biométrica de iris humano usando redes neuronales modulares. Komputer Sapiens (Julio 2010)
[8] Oniga, S.: Hand Posture Recognition System Using Artificial Neural Networks Implementede in FPGA, May 9-13 (2007)
[9] Ni, D.X.: Application of Neural Network to Character Recognition. Seidenberg School of CSIS. Pace University, White Plains (2007)
[10] Mani, N., Srinivasan, B.: Application of Artificial Neural Network Model for Optical Character Recognition. In: IEEE International Conference, October 12-15 (1997)
[11] Neo, H., Teoh, B., Ngo, C.L.: A Novel Spatially Confined Non-Negative Matrix Factorization for Face Recognition. In: MVA 2005 IAPR Conference on Machine Vision Applications, Tsukuba Science City, Japan (May 2005)
[12] Tanaya, M., Angshul, M., Jonathan, W.: Face Recognition by Curvelet Based Feature Extraction. Department of Electrical and Computer Engineering, University of Windsor, Canada, Springer, Heidelberg (2007)
[13] Tanaya, M., Angshul, M., Jonathan, W.: Face Recognition by using Curvelet Base PCA. Department of Electrical and Computer Engineering, University of Windsor, Canada. IEEE (2008)
[14] Fevrier, V., Patricia, M., Herman, P.: Parallel genetic algorithms for optimization of Modular Neural Networks in pattern recognition. In: IJCNN 2011, pp. 314–319 (2011)

Part III

Bio-Inspired and Genetic Optimization Methods

Comparative Study of Particle Swarm Optimization Variants in Complex Mathematics Functions

Juan Carlos Vazquez, Fevrier Valdez, and Patricia Melin

Tijuana Institute of Technology
juancarlos_jcs27@hotmail.com,
fevrier@tectijuana.mx,
pmelin@tectijuana.edu.mx

Abstract. Particle Swarm Optimization (PSO) is one of the evolutionary computation techniques based on the social behaviors of birds flocking or fish schooling, biologically inspired computational search and optimization method. Since first introduced by Kennedy and Eberhart [7] in 1995, several variants of the original PSO have been developed to improve speed of convergence, improve the quality of solutions found, avoid getting trapped in the local optima and so on. This paper is focused on performing a comparison of different PSO variants such as full model, only cognitive, only social, weight inertia, and constriction factor. We are using a set of 4 mathematical functions to validate our approach. These functions are widely used in this field of study.

1 Introduction

Particle Swarm Optimization (PSO) was developed by Kennedy and Eberhart in 1995 [7] [11], and is a stochastic search method based on population. The idea behind this algorithm was inspired by the social behavior of animals, such as bird flocking or fish schooling. The process of the PSO algorithm in finding optimal values follows the work of this animal society. In a PSO system, a swarm of individuals (called particles) fly through the search space. Each particle represents a candidate solution to the optimization problem. The performance of each particle is measured using a fitness function that varies depending on the optimization problem. The PSO has been applied successfully to a number of problems, including standard function optimization problems [1] [12] [20] [21], solving permutation problems [15] [19] and training multi-layer neural networks [13] [14] [20]. The basic PSO has problems with consistently converging to good solutions, so, there are several modifications that have been proposed from the original PSO. It modifies to accelerate achieving of the best conditions, improving convergence of the PSO and increasing the diversity of the swarm. A brief description of some PSO approaches is presented below. Benchmark functions were used to measure the performance of the PSO algorithm with the different approaches. This

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 223–235.
springerlink.com

paper is organized as follows: section 2 describes the standard PSO algorithms, section 3 describe the different variants of standard PSO algorithms used in this paper, section 4 presents simulation results for functions mathematic and finally, conclusions are summarized in sections 5.

2 Standard PSO Algorithm

The basic equations are usually given as follow:

$$v_{ij}(t+1) = vi_{ij}(t) + c_1 r_{1j}(t)[y_{ij}(t) - x_{ij}(t)] + c_2 r_{2j}(t)[\hat{y}_j(t) - x_{ij}(t)] \quad (1)$$

$$x_i(t+1) = x_{ij}(t) + v_{ij}(t+1) \quad (2)$$

The social network employed by the PSO reflects the star topology. For the star neighborhood topology, the social component of the particle velocity update reflects information obtained from all the particles in the swarm, referred to as $\hat{y}(t)$, where $v_i(t)$ is the velocity of particle i in dimension $j = 1, \ldots, n_x$ at time step t, $x_{ij}(t)$ is the position of particle i in dimension j at time step t, c_1 and c_2 are positive acceleration constants used to scale the contribution of the cognitive and social components respectively, and $r_{1j}(t)$, $r_{1j}(t) \sim U(0,1)$ are random values in the range [0, 1], sampled from a uniform distribution. These random values introduce a stochastic element to the algorithm.

Let $x_i(t)$ denote the position of particle i in the search space at time step t, which denotes discrete time steps. The position of the particle is changed by adding a velocity, $v_i(t)$ to the current position. It is the velocity vector that drives the optimization process, and reflects both the experiential knowledge of the particle and socially exchanged information from the particle's neighborhood.

3 Variants of PSO

Several variants of the PSO algorithm have been developed [16] [17] [10]. It has been shown that the question of convergence of the PSO algorithm is implicitly guaranteed if the parameters are adequately selected [18] [4].

3.1 Inertia Weight

This variation was introduced by Shi and Eberthart [17] as a mechanism to control the exploration and exploitation abilities of the swarm, and as a mechanism to eliminate the need for velocity clamping [9]. The inertia weight was successful in addressing the first objective, but could not completely eliminate the need for velocity clamping. The inertia weight, w, basically works by controlling how much

memory of the previous flight direction will influence the new velocity. The velocity equation is changed from equation (1) to:

$$v_{ij}(t+1) = wvi_j(t) + c_1 r_{1j}(t)[y_{ij}(t) - x_{ij}(t)] + c_2 r_{2j}(t)[\hat{y}_j(t) - x_{ij}(t)] \quad (3)$$

The value of *w* is extremely important to ensure convergent behavior, and to optimally tradeoff exploration and exploitation. For $w \geq 1$, velocities increase over time, accelerating towards the maximum velocity (assuming velocity clamping is used), and the swarm diverges. For $w < 1$, particles decelerate until their velocities reach zero (depending on the values of the deceleration coefficients). Large values for *w* facilitate exploration, with increased diversity. Very small values eliminate the exploration ability of the swarm.

3.2 Constriction Coefficient

Clerc developed an approach very similar to the inertia weight to balance the exploration-exploitation trade-off, where the velocities are constricted by a constant *X,* referred to as the constriction coefficient [5]. The velocity update equation changes to:

$$v_{ij}(t+1) = x[vi_j(t) + c_1 r_{1j}(t)[y_{ij}(t) - x_{ij}(t)] + c_2 r_{2j}(t)[\hat{y}_j(t) - x_{ij}(t)] \quad (4)$$

where

$$x = \frac{2k}{\left|2 - \phi - \sqrt{\phi(\phi-4)}\right|} \quad (5)$$

with

$$\phi = \phi_1 + \phi_2$$
$$\phi_1 = c_1 r_1$$
$$\phi_2 = c_2 r_2$$

Equation (5) is used under the constraints that $\phi \geq 4$ and $k \in [0,1]$.

The constriction equations above were derived from a formal eigenvalue analysis of swarm dynamics [6].

The constriction approach was developed as a natural, dynamic way to ensure convergence to a stable point, without the need for a velocity clamping. Under the conditions that $\phi \geq 4$ and $k \in [0,1]$, the swarm is guaranteed to converge. The constriction coefficient, *x*, evaluates to a values in the range [0, 1] which implies that the velocity is reduced at each time step.

The parameter, *k*, in equation (5) controls the exploration and exploitation abilities of the swarm. For $k \approx 0$, fast convergence is obtained with local exploitation. On the other hand, $k \approx 1$, results in slow convergence with a high degree of exploration. Usually, k is set to a constant value. However, an initial high degree of exploration with local exploitation in the later search phases can be achieved using an initial value close to one, decreasing it to zero.

The constriction approach is effectively equivalent to the inertia weight approach. Both approaches have the objective of balancing exploration and exploration, and in doing so of improving convergence.

3.3 Cognition-Only Model

The cognition-only model excludes the social components from the original velocity equation as given in equation (1). For the cognition-only model, the velocity update changes to:

$$v_{ij}(t+1) = vi_{ij}(t) + c_1 r_{1j}(t)(y_{ij}(t) - x_{ij}(t)) \tag{6}$$

The behavior of the particle within the cognition-only model can be composed to nostalgia. The cognition-only model provides a stochastic tendency for particles to return toward their previous best position. From empirical work, Kennedy reported that the cognition-only model is slightly more vulnerable to failure than the full model. It tends to locally search in areas where particles are initialized. The cognition-only model is slower in the number of iterations it requires to reach a good solution, and fails when velocity clamping and the acceleration coefficient are small. The poor performance of the cognitive model is confirmed by Carlisle and Dozier [19], but with respect to dynamic changing environments.

The cognition-only model was, however, successfully used within niching algorithms [2].

3.4 Social-Only Model

The social-only model excludes the cognitive component from the velocity equation:

$$v_{ij}(t+1) = vi_{ij}(t) + c_2 r_{2j}(t)(\hat{y}_j(t) - x_{ij}(t)) \tag{7}$$

For the social-only model, particles have no tendency to return to previous best positions. All particles are attracted towards the best position of their neighborhood.

4 Simulation Results for Mathematical Functions

In this section four mathematical functions are tested under the different reviewed variants of PSO.

4.1 F1 Function

$$|x| + \cos(x) \tag{8}$$

where $-100 < x_n < 100$ n=1, n is the number of the variable to be optimized. The objective is to find the minimum of F1 function and the related variable

locations. Fig.1 shows the surface plot of the F1 function in one variable. There is a unique minimum point in the figure with location $f(0)=1$.

For the simulations we used a population size of 20, 40, 60, 80 and 100. $c_1 = c_2 = 2,$ Inertia weight (W) = [0.2, 0.4, 0.6, 0.8], 100000 generations and 5 experiments.

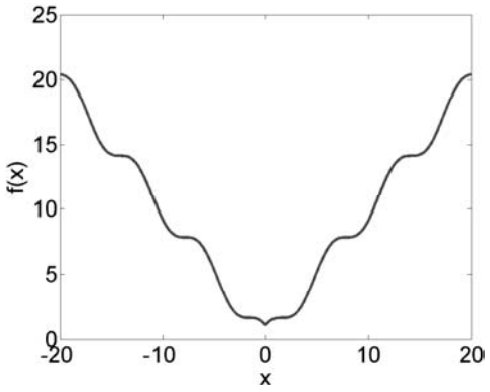


Fig. 1. Surface plot of the F1 function in n=1 variable

Table 1 shows the optimization results of F1 function with different variants, Basic PSO, Only Cognitive and Only Social, where *Tswarm* is the population size, *G* is average of generations, *Success rate* is the reached minimum in the generation *G* and the number of variables is 1.

Table 1. Optimization results of F1 function with the different variants

Number of variables	Tswar	Basic PSO G	Success rate	Only Cognitive G	Success rate	Only Social G	Success rate
1	20	100000	1.0001	82061.8	1	100000	1.0001
1	40	53746.8	1	49784.6	1	50839.4	1
1	60	38763.4	1	27245.5	1	33767.2	1
1	80	21683.1	1	16440.7	1	24722.7	1
1	100	21398.6	1	16488.3	1	20084.6	1

Table 2 shows the optimization results of F1 function, where *W* is the inertia weight with different values, *X* is the constriction factor, *Tswarm* is the population size, *G* is average of generations, *Success rate* is the reached minimum in the generation *G* and the number of variables is 1.

Table 2. Optimization results of F1 function, inertia weight and factor constriction variants

Number of variables	*W* / Tswar	0.2 G	Success rate	0.4 G	Success rate	0.6 G	Success rate	0.8 G	Success rate	*X*	Success rate
1	20	100000	1.0003	100000	1.0001	296.4	1	482	1	356.6	1
1	40	100000	1.0011	100000	1.0001	271.4	1	451	1	345.8	1
1	60	100000	1.0012	100000	1.0001	289	1	420.2	1	344.4	1
1	80	100000	1.0001	100000	1.0001	278.4	1	433.6	1	340	1
1	100	100000	1.0003	100000	1.0001	267	1	427	1	342	1

4.2 F2 Function

$$x\sin(4x)+1.1y\sin(2y) \tag{9}$$

where $0 < x_n < 10$ n=2, n is the number of variables to be optimized. The objective is to find the minimum of F2 function and the related variable locations. Fig.2 shows the surface plot of the F2 function in one variable. There is a unique minimum point in the figure with location $f(0.9039,0.8668) = -18.5547$.

For the simulations we used a population size of 20, 40, 60, 80 and 100. $c_1 = c_2 = 2,$ Inertia weight (W) = [0.2, 0.4, 0.6, 0.8], 100000 generations and 5 experiments.

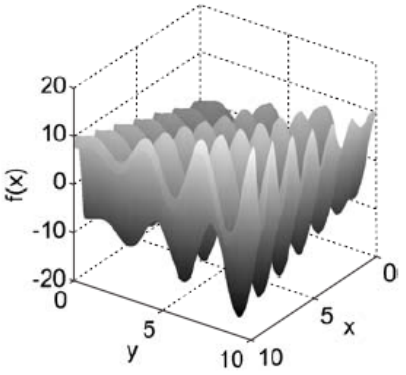


Fig. 2. Surface plot of the F2 function in n=2 variables

Table 3 shows the optimization results of F2 function with different variants, Basic PSO, Only Cognitive and Only Social, where *Tswarm* is the population size, *G* is average of generations, *Success rate* is the reached minimum in the generation *G* and the number of variables is 2.

Table 3. Optimization results of F2 function with the different variants

Number of variables	*W* / Tswar	Basic PSO G	Success rate	Only Cognitive G	Success rate	Only Social G	Success rate
2	20	30262.2	-18.5547	48285.2	-18.5547	70274.4	-16.7476
2	40	26157.6	-18.5547	23359.6	-18.5547	56851.4	-16.6509
2	60	25183.2	-18.5547	17634.8	-18.5547	23282.6	-18.5547
2	80	6255	-18.5547	17237	-18.5547	10195	-18.5547
2	100	11022.4	-18.5547	9430.2	-18.5547	10106.4	-18.5547

Table 4 shows the optimization results of F2 function, where *W* is the inertia weight with different values, *X* is the constriction factor, *Tswarm* is the population size, *G* is average of generations, *Success rate* is the reached minimum in the generation *G* and the number of variables is 2.

Table 4. Optimization results of F2 function, inertia weight and factor constriction variants

Number of variables	*W* / Tswar	0.2 G	Success rate	0.4 G	Success rate	0.6 G	Success rate	0.8 G	Success rate	*X* G	Success rate
2	20	57343.4	-18.5547	738.2	-18.5547	62.4	-18.5547	98.2	-18.5547	35	-18.2406
2	40	6726.4	-18.5547	273.2	-18.5547	72.4	-18.5547	76.4	-18.5547	43.4	-18.5547
2	60	3412.6	-18.5547	288.4	-18.5547	52.6	-18.5547	66.4	-18.5547	14.2	-18.5547
2	80	2222.6	-18.5547	243.8	-18.5547	46.6	-18.5547	71.8	-18.5547	11	-18.5547
2	100	2684.8	-18.5547	152.8	-18.5547	53.4	-18.5547	58.2	-18.5547	10.6	-18.5547

4.3 F3 Function

$$f(x)=\sum_{n=1}^{N-1}\left\{100\left[x_{n+1}-x_n^2\right]^2+\left[1-x_n\right]^2\right\} \tag{10}$$

where $-100 < x_i < 100$, i = 1, 2,…, n, n is the number of variables to be optimized. The objective is to find the minimum of F3 function and the related variable locations. Fig.3 shows the surface plot of the F3 function in one variable. There is a unique minimum point in the figure with location $f(1,1)=0$.

For the simulations we used a population size of 20, 40, 60, 80 and 100. $c_1 = c_2 = 2,$ Inertia weight (*W*) = [0.2, 0.4, 0.6, 0.8], 100000 generations and 5 experiments.

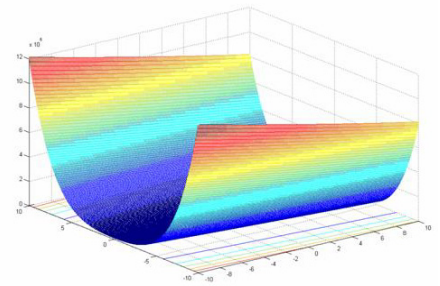


Fig. 3. Surface plot of the F3 function in n=2 variables

Table 5 shows the optimization results of F3 function, where *W* is the inertia weight with different values, *X* is the constriction factor, *Tswarm* is the population size, *G* is average of generations, *Success rate* is the reached minimum in the generation *G* and the number of variables is 2.

Table 5. Optimization results of F3 function, inertia weight and factor constriction variants

Number of variables	*W* / Tswar	0.2 G	Success rate	0.4 G	Success rate	0.6 G	Success rate	0.8 G	Success rate	*X* *G*	Success rate
2	20	100000	56.5	100000	3.2 001	3099.6	0	2654.9	0	100000	18.5
2	40	100000	46.6	100000	0.0204	2222.3	0	2195.6	0	100000	9.2
2	60	100000	27.4	100000	0.0114	2033.5	0	2030.2	0	82486.2	5.2
2	80	100000	32.5	100000	0.0098	1926.7	0	1905.7	0	60060.4	3.1
2	100	100000	30.7	100000	0.0054	1842.8	0	1829.5	0	80137.8	3.2

Table 6 shows the optimization results of F3 function, where *W* is the inertia weight with different values, *X* is the constriction factor, *Tswarm* is the population size, *G* is average of generations, *Success rate* is the reached minimum in the generation *G* and the number of variables is 4.

Table 6. Optimization results of F3 function, inertia weight and factor constriction variants

Number of variables	*W* / Tswar	0.2 G	Success rate	0.4 G	Success rate	0.6 G	Success rate	0.8 G	Success rate	*X* *G*	Success rate
4	20	100000	259811.4	100000	104.9	100000	2.9104e -027	100000	2.3296e -030	100000	1946.6
4	40	100000	250096.4	100000	48.32	100000	1.0354e -030	67592.6	0	100000	128.7
4	60	100000	2685.1	100000	76.5	98088.4	0	73702.4	0	100000	25.6
4	80	100000	5245.8	100000	62.1	73809.6	0	66970.8	0	100000	10.1
4	100	100000	23.5	100000	156.3	77679.4	0	50556.4	0	100000	0.8

Table 7 shows the optimization results of F3 function, where *W* is the inertia weight with different values, *X* is the constriction factor, *Tswarm* is the population size, *G* is average of generations, *Success rate* is the reached minimum in the generation *G* and the number of variables is 6.

Table 7. Optimization results of F3 function, inertia weight and factor constriction variants

Number of variables	W / Tswar	0.2 G	Success rate	0.4 G	Success rate	0.6 G	Success rate	0.8 G	Success rate	X G	Success rate
6	20	100000	3637934.9	100000	504058.8	100000	290.2	100000	2500.5	100000	4193.2
6	40	100000	601490.8	100000	4424.4	100000	22.5	100000	21.6	100000	2321.3
6	60	100000	500383.8	100000	501119.4	100000	64.8	100000	21.6	100000	151.7
6	80	100000	251034.8	100000	750375.2	100000	4.1455e-009	100000	22.5	100000	132.4
6	100	100000	250116.2	100000	783.9	100000	21.6	100000	21.7	100000	3.7

Table 8 shows the optimization results of F3 function, where *W* is the inertia weight with different values, *X* is the constriction factor, *Tswarm* is the population size, *G* is average of generations, *Success rate* is the reached minimum in the generation *G* and the number of variables is 8.

Table 8. Optimization results of F3 function, inertia weight and factor constriction variants

Number of variables	W / Tswar	0.2 G	Success rate	0.4 G	Success rate	0.6 G	Success rate	0.8 G	Success rate	X G	Success rate
8	20	100000	2981228.8	100000	658030.4	100000	250022.1	100000	44.2	100000	128964.3
8	40	100000	1398054.9	100000	634092.0	100000	250000.5	100000	1.9	100000	2604.1
8	60	100000	757957.1	100000	22594.1	100000	43.2	100000	44.2	100000	2375.1
8	80	100000	101143.9	100000	9882.2	100000	17.3	100000	0.2101	100000	85.3
8	100	100000	205438.1	100000	211345.6	100000	86.4	100000	0.7971	100000	478.2

The table 9 shows the optimization results of F3 function, where *W* is the inertia weight with different values, *X* is the constriction factor, *Tswarm* is the population size, *G* is average of generations, *Success rate* is the reached minimum in the generation *G* and the number of variables is 10.

Table 9. Optimization results of F3 function, inertia weight and factor constriction variants

Number of variables	W / Tswar	0.2 G	Success rate	0.4 G	Success rate	0.6 G	Success rate	0.8 G	Success rate	X G	Success rate
10	20	100000	191603992.5	100000	4213320.2	100000	200000.4	100000	2019.5	100000	158087.3
10	40	100000	1717782.9	100000	1733716.8	100000	600035.9	100000	35.4	100000	2778.4
10	60	100000	2153458.6	100000	519313.1	100000	2039.2	100000	2019.7	100000	1147.4
10	80	100000	189513.1	100000	575338.9	100000	400018.2	100000	17.3	100000	147.8
10	100	100000	145306.6	100000	738666.1	100000	400035.5	100000	17.3	100000	433.9

4.4 F4 Function

$$10N + \sum_{n=1}^{N} \left[x_n^2 - 10\cos(2\pi x_n) \right] \tag{11}$$

where $-100 < x_i < 100$, i = 1, 2,…, n, n is the number of variables to be optimized. The objective is to find the minimum of F2 function and the related variable locations. Fig.2 shows the surface plot of the F2 function in one variable. There is a unique minimum point in the figure with location $f(0,0) = 0$.

For the simulations we used a population size of 20, 40, 60, 80 and 100. $c_1 = c_2 = 2,$ Inertia weight (W) = [0.2, 0.4, 0.6, 0.8], 10000 generations and 50 experiments.

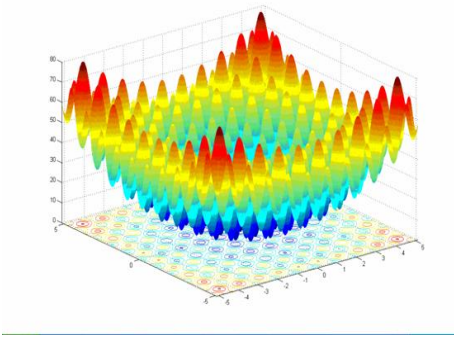


Fig. 4. Surface plot of the F4 function in n=2 variables

Table 10 shows the optimization results of F4 function, where *W* is the inertia weight with different values, *X* is the constriction factor, *Tswarm* is the population size, *G* is average of generations, *Success rate* is the reached minimum in the generation *G* and the number of variables is 2.

Table 10. Optimization results of F4 function, inertia weight and factor constriction variants

Number of variables	*W* / Tswar	0.2 G	Success rate	0.4 G	Success rate	0.6 G	Success rate	0.8 G	Success rate	*X* *G*	Success rate
2	20	100000	6.1	100000	0.0005	323.1	0	323.1	0	18311.6	0
2	40	100000	4.2	100000	0.0003	297.4	0	297.4	0	273.2	0
2	60	100000	1.8	100000	0.0001	279.9	0	279.9	0	53.7	0
2	80	100000	0.9	100000	0.0001	273.9	0	273.9	0	54.7	0
2	100	100000	1.1	100000	0.0002	267.9	0	267.9	0	44.3	0

Table 11 shows the optimization results of F4 function, where *W* is the inertia weight with different values, *X* is the constriction factor, *Tswarm* is the population size, *G* is average of generations, *Success rate* is the reached minimum in the generation *G* and the number of variables is 4.

Table 11. Optimization results of F4 function, inertia weight and factor constriction variants

Number of variables	*W* / Tswar	0.2 G	Success rate	0.4 G	Success rate	0.6 G	Success rate	0.8 G	Success rate	*X* *G*	Success rate
4	20	100000	34.3	100000	12.8	1547.1	0	2087.9	0	100000	9.944
4	40	100000	31.5	100000	9.8	783.9	0	907.5	0	100000	7.164
4	60	100000	16.9	100000	9.9	721.2	0	815.6	0	100000	5.501
4	80	100000	11.7	100000	8.1	669.2	0	805.1	0	100000	4.457
4	100	100000	13.8	100000	6.9	624.0	0	759.3	0	100000	2.654

Table 12 shows the optimization results of F4 function, where *W* is the inertia weight with different values, *X* is the constriction factor, *Tswarm* is the population size, *G* is average of generations, *Success rate* is the reached minimum in the generation *G* and the number of variables is 6.

Table 12. Optimization results of F4 function, inertia weight and factor constriction variants

Number of variables	*W* / Tswar	0.2 G	Success rate	0.4 G	Success rate	0.6 G	Success rate	0.8 G	Success rate	*X* *G*	Success rate
6	20	100000	305.8	100000	47.9	5951.5	0	6396.0	0	100000	204.3
6	40	100000	68.5	100000	57.3	3465.6	0	3676.2	0	100000	31.1
6	60	100000	46.1	100000	47.2	2066.3	0	2036.4	0	100000	31.9
6	80	100000	35.3	100000	38.2	1521.8	0	1404.8	0	100000	25.3
6	100	100000	50.1	100000	33.4	1181.7	0	1355.4	0	100000	12.7

Table 13 shows the optimization results of F4 function, where *W* is the inertia weight with different values, *X* is the constriction factor, *Tswarm* is the population size, *G* is average of generations, *Success rate* is the reached minimum in the generation *G* and the number of variables is 8.

Table 13. Optimization results of F4 function, inertia weight and factor constriction variants

Number of variables	*W* / Tswar	0.2 G	Success rate	0.4 G	Success rate	0.6 G	Success rate	0.8 G	Success rate	*X* *G*	Success rate
8	20	100000	1132.0	100000	198.7	100000	1.5919	100000	2.1889	100000	283.4
8	40	100000	232.9	100000	148.4	100000	0.1989	100000	0.3979	100000	82.1
8	60	100000	149.3	100000	111.1	100000	0.1989	100000	0.3979	100000	60.7
8	80	100000	124.4	100000	84.5	4901.2	0	6534	0	100000	41.5
8	100	100000	95.7	100000	70.2	2038.2	0	2151.8	0	100000	29.9

Table 14 shows the optimization results of F4 function, where *W* is the inertia weight with different values, *X* is the constriction factor, *Tswarm* is the population size, *G* is average of generations, *Success rate* is the reached minimum in the generation *G* and the number of variables is 10.

Table 14. Optimization results of F4 function, inertia weight and factor constriction variants

Number of variables	*W* / Tswar	0.2 G	Success rate	0.4 G	Success rate	0.6 G	Success rate	0.8 G	Success rate	*X* *G*	Success rate
10	20	100000	3725.6	100000	486.2	100000	2.5869	100000	2.7859	100000	347.4
10	40	100000	694.8	100000	278.7	100000	1.5919	80835.4	1.3929	100000	243.6
10	60	100000	391.1	100000	185.1	44431	0.5969	80672.8	0.9949	100000	100.3
10	80	100000	185.6	100000	159.2	42176	0.3979	61146	0.9949	100000	103.5
10	100	100000	138.5	100000	113.7	4658.4	0	12037	0	100000	82.3

5 Conclusions

In this paper, we proposed a comparative study of PSO variants. Simulation tests on four mathematical functions were performed to compare the proposed study. The simulation results show that the standard PSO algorithm with one or more variables used too many generations (mean) to find the minimum. Similar results with variations on only cognitive and only social, thus in some tests do not show results of simulations of these variants. Inertia weight and constriction factor variants show better results to find the minimum of the function, where some of the different values of inertia weight variation showed better results than with constriction factor variation. As shown in the results among more variables has the function the PSO used more generations to find the minimum, therefore results show great difference in both the basic PSO as its variations.

References

1. Angeline, P.J.: Evolutionary Optimization versus Particle Swarm Optimization: Philosophy and Performance Differences. In: Porto, V.W., Waagen, D. (eds.) EP 1998. LNCS, vol. 1447, pp. 601–610. Springer, Heidelberg (1998)
2. Brits, R., Engelbrecht, A.P., van den Bergh, F.: A Niching Particle Swarm Optimizer. In: Proceedings of the Fourth Asia-Pacific Conference on Simulated Evolution and Learning, pp. 692–696 (2002)
3. Carlisle, A., Dozier, G.: Adapting Particle Swarm Optimization to Dynamic Environments. PhD thesis. Auburn University (2002)
4. Cristian, T.I.: The particle swarm optimization algorithm: convergence analysis and parameter selection. Information Processing Letters 85(6), 317–325 (2003)
5. Clerc, M.: The Swarm and the Queen: Towards a Deterministic and Adaptive Particle Swarm Optimization. In: Proceedings of the IEEE Congress on Evolutionary Computation, vol. 3, pp. 1951–1957 (July 1999)
6. Clerc, M., Kennedy, J.: The Particle Swarm-Explosion, Stability, and Convergence in a Multidimensional Complex Space. IEEE Transactions on Evolutionary Computation 6(1), 58–73 (2002)

7. Eberhart, R.C., Kennedy, J.: A New Optimizer using Particle Swarm Theory. In: Procedings of the Sixth International Symposium on MicroMachine and Human Science, pp. 39–43 (1995)
8. Eberhart, R.C., Shi, Y.: Comparing Inertia Weights and Constriction Factors in Particle Swarm Optimization. In: Proceedings of the IEEE Congress on Evolutionary Computation, vol. 1, pp. 84–88 (July 2000)
9. Eberhart, R.C., Shi, Y.: Particle Swarm Optimization: Developments, Applications and Resources. In: Proceedings of the IEEE Congress on Evolutionary Computation, vol. 1, pp. 27–30. IEEE Press (May 2001)
10. Kennedy, J.: The behaviour of particles. Evol. Progr. VII, 581–587 (1998)
11. Kennedy, J., Eberhart, R.C.: Particle Swarm Optimization. In: Proceedings of the IEEE International Joint Conference on Neuronal Networks, pp. 1942–1948. IEEE Press (1995)
12. Kennedy, J., Spears, W.: Matching Algorithms to problems: An Experimental Test of the Particle Swarm and Some Genetic Algorithms on the Multimodal Problem Generator. In: Proceedings of the IEEE Congress on Evolutionary Computation, pp. 78–83. IEEE Press (May 1998)
13. Ribeiro, P.F., Kyle Schlansker, W.: A Hybrid Particle Swarm and Neuronal Network Approach for Reactive Power Control. IEEE (2006)
14. Russell, C., Eberthart, Hu, X.: Human Tremor Analysus Using Particle Swarm Optoimization. Purdue Shool of Engineering and Technology, Indiana University Purdue University Indianapolis, Indianapolis (1999)
15. Salerno, J.: Using the Particle Swarm Optimization Technique to Train a Recurrent Neural Model. In: In Proceedings of the IEEE International Conference on Tools with Artificial Intelligence, pp. 45–49. IEEE Press (November 1997)
16. Shi, Y.H., Eberhart, R.C.: Fuzzy adaptive particle swarm optimization. In: IEEE Int. Conf. on Evolutionary Computation, pp. 101–106 (2001)
17. Shi, Y.H., Eberhart, R.C.: A modified particle swarm optimizer. In: Proceedings of the IEEE International Conference on Evolutionary Computation, pp. 69–73 (May 1998)
18. Shi, Y.H., Eberhart, R.C.: Parameter Selection in Particle Swarm Optimization. In: Porto, V.W., Waagen, D. (eds.) EP 1998. LNCS, vol. 1447, pp. 591–600. Springer, Heidelberg (1998)
19. Tasgetiren, M.F., Liang, Y.C., Sevkli, M., Gencyilmaz, G.: A Particle Swarm Optimization Algorithm for Makespan and Total Flowtime Minimization in the Permutation Flowshop Sequencing Problem. European Journal of Operational Research 177, 1930–1947 (2007)
20. Valdez, F., Melin, P., Castillo, O.: Evolutionary method combining particle swarm optimization and genetic algorithms using fuzzy logic for decision making. In: Proceedings of the IEEE International Conference on Fuzzy Systems, pp. 2114–2119 (2009)
21. Valdez, F., Melin, P., Castillo, O.: An improved evolutionary method with fuzzy logic for combining Particle Swarm Optimization and Genetic Algorithms. Appl. Soft Comput. 11(2), 2625–2632 (2011)

A Method to Solve the Traveling Salesman Problem Using Ant Colony Optimization Variants with Ant Set Partitioning

Evelia Lizárraga, Oscar Castillo, and José Soria

Tijuana Institute of Technology, Tijuana, Mexico

Abstract. In this paper we propose an ant's partition method for Ant Colony Optimization (ACO), a meta-heuristic that is inspired in ant's behavior and how they collect their food. The proposed method equivalently divides the total number of ants in three different subsets and each one is evaluated separately by the corresponding variation of ACO (AS, EAS, MMAS) to solve different instances of The Traveling Salesman Problem (TSP). This method is based on the idea of "divide and conquer" to be applied in the division of the work, as the ants are evaluated in different ways in the same iteration. This method also includes a stagnation mechanism that stops at a certain variation if it's not working properly after several iterations. This allows us to save time performing tests and have less overhead in comparison with the conventional method, which uses just one variation of ACO in all iterations.

1 Introduction

Nowadays there are open NP-complete problems where the complexity to construct the right configuration of a solution lays in finding a possible solution within a great number of possible combinations. In the past it was assumed that this kind of problems where impossible to solve since the feasibility to find the right answer was attached to the idea that an exhaustive search was needed resulting in the impossibility to find an answer in a reasonable frame of time. In many cases even when a super computer was used the amount of time needed to find a solution was monumental causing overhead and great computational complexity. Therefore NP-hard problems are approached finding the solution in a subset of the decision problem; this means finding an optimal solution using approximate algorithms that can give us good solutions efficiently, sacrificing finding very good solutions in a polynomial time. One of the most known examples of this kind of problems is the TSP (Traveling Salesman Problem), which is represented figuratively as a salesman that wants to travel from city to city starting in a original point (city) without passing a city twice before returning home (original city). This is considering the length between cities as the cost that needs to be minimized. This problem can be represented as an undirected weight graph where the cities are the vertices of the graph, paths are graph's edges and the distance of the

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 237–246.
springerlink.com

path is the length of the edge. There are many approaches proposed to solve this problem, one of the most used are meta-heuristics inspired in biological behavior such as Ant Colony Optimization(ACO)[4][5], Genetic Algorithms (GA), Particle Swarm Optimization (PSO) [7][1], among others. These techniques are known to be efficient in the search of a space solution providing optimal values. Given that ACO can be represented as graph, this meta-heuristic is one of the most used to represent and solve TSP. Dorigo and Stützle proposed the solution of TSP in [4][8][9][11] using ACO according to the biological information that an ant provides when a good path is found adding pheromone to the nodes of that specific path. ACO uses the amount of pheromone and heuristic information to find the probability of the next node, were the heuristic information in this case (TSP) is the distance between cities. Owing to the fact that ACO has many variations in how the pheromone it's applied; the way to solve each problem is narrowed to how a designer uses these variations and how good is a specific variation in a specific problem. In this paper we propose a strategy to compare these variations, such as Elitist Ant System (EAS), Ant System (AS), and Max Min Ant System (MMAS) within the same iteration; hence minimizing the number of tests needed. For this paper we performed 30 experiments using the proposed method (P-ACO) compared to 30 experiments for each ACO variation (AS, EAS, MMAS) resulting in a good performance and reduction of time. The main contribution of this work is the proposed P-ACO approach that combines the use of EAS, AS and MMAS in a single method that is more efficient in finding solutions for complex problems, such as the TSP.

1.1 Ant Colony Optimization (ACO)

Meta-heuristic introduced by Dorigo and Stützle that is based on ant's behavior and how they collect their food following biological and heuristic information. Artificially speaking each ant represents a possible solution in which each one of them cooperates to find an optimal global solution. ACO works creating a construction graph Gc(C=L) identical to the problem graph. Each node connects to a corresponding set of arcs (i.e. L=A) and those arcs have a weight that corresponds to the distance d_{ij} between nodes *i* to *j*.

$$\forall (i, j) \in L$$

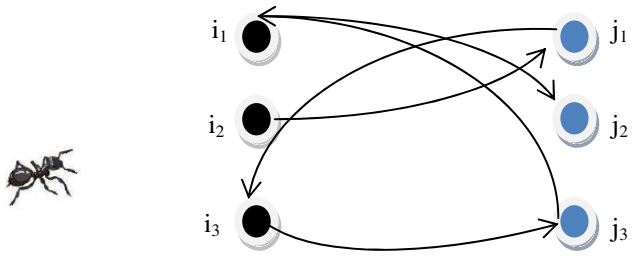


Fig. 1. Shows the nodes and the corresponding arcs

After the graph is constructed, a set of constraints are defined, which depend on the problem. In this case the only constraint is, that a city is visited only once within the feasible neighborhood N_i^k where *k* is the ant and *i* is the node to be visited choosing the next node probabilistically using (1). The ants "memorize" the visited nodes as well as the cost of that node (heuristic information).The evaporation of the pheromone (3), helps to forget bad decisions and follow the best path; also delimitating the maximum amount of pheromone deposited by the ants. The following step defines how the pheromone τ_{ij} is updated (2) and the evaporation rate p that is needed, also the heuristic information η_{ij} that is going to be used; the distance between nodes gives this information. Finally we construct a solution setting the ants randomly in nodes to start searching paths. The algorithm ends when all the cities have been visited.

$$p_{ij}^k(t) = \begin{cases} \dfrac{[\tau_{ij}(t)]^\alpha [\eta_{ij}]^\beta}{\sum_{k \in N_i^k} [\tau_{ik}(t)]^\alpha [\eta_{ik}]^\beta} & \text{si } j \in N_i^k \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

$$\Delta\tau_{ij} \leftarrow \tau_{ij} + \Delta\tau^k \tag{2}$$

$$\begin{aligned} &\tau_{ij} \leftarrow (1-p)\tau_{ij} \\ &p \in (0,1] \end{aligned} \tag{3}$$

1.2 ACO Variations

According to the literature [4] there exist many variations of ACO, the slightly difference among them is in how they update the pheromone and the use of some parameters to perform such update. Some variations are the best for certain problems including the TSP like MMAS, EAS and RankAS. In the same way these variations have different functionality in specific problems. Table 1 shows the difference of the variations that we are going to use, including the parameter values recommended in literature [4][5][11].

Where *avg* is the average number of different choices available to an ant at each step while constructing a solution [10]. The main justification for using these variations is to test the feasibility of the proposed method P-ACO using the variations that have different performance in TSP, like AS that performs badly in comparison with the others, EAS which has proved that can provide good solutions and finally MMAS, which is one of the most used variations in TSP because it gives

Table 1. Different variations of ACO

Variation	Description	τ_0	Parameter values
Elitist Ant System (EAS)	The global best solution deposits pheromone on every iteration along with all the other ants.	$(e+m)/\rho C^{nn}$ (4)	$\alpha=1$ $\beta=2$ to 5 $\rho=0.5$ $m=n$ $e=n$
MaxMin Ant System (MMAS)	Added Maximum and Minimum pheromone amounts $[\tau_{max},\tau_{min}]$ Only global best or iteration best tour deposited pheromone. All edges are initialized to τ_{max} and reinitialized to τ_{max} when nearing stagnation.	$1/\rho C^{nn}$ (5) $\tau_{min} = \tau_{max}(1-\sqrt[n]{0.05})/((avg-1)*\sqrt[n]{0.05})$ (6) $\tau_{max}=1/\rho C^{nn}$ (7)	$\alpha=1$ $\beta=2$ to 5 $\rho=0.5$ $m=n$
Ant System	Updates pheromones after construct solutions	m/C^{nn} (8)	

excellent results. This will show us if the algorithm is working when it compares the different results of the different variations per iteration, expecting the best one (MMAS) as the winner, this means the variation that produces the best global in each test.

2 Proposed Method

This method provides a way to divide the total number of ants in different subsets. This means given a set of *m* ants the method equivalently divides the total number of ants in three different subsets and each one is evaluated separately by the corresponding variation of ACO (AS,EAS and MMAS) to solve different instances of The Traveling Salesman Problem (TSP).

The stagnation mechanism works evaluating the best result per iteration and counting which variation have poor results. Making several experiments we concluded that the feasible number of allowed poor results is five; this means if the variation best result is not the global best in that iteration for five consecutively times then the variation is stopped. This could cause that none of the variations are stopped if there isn't five consecutively bad results in any of them, also could

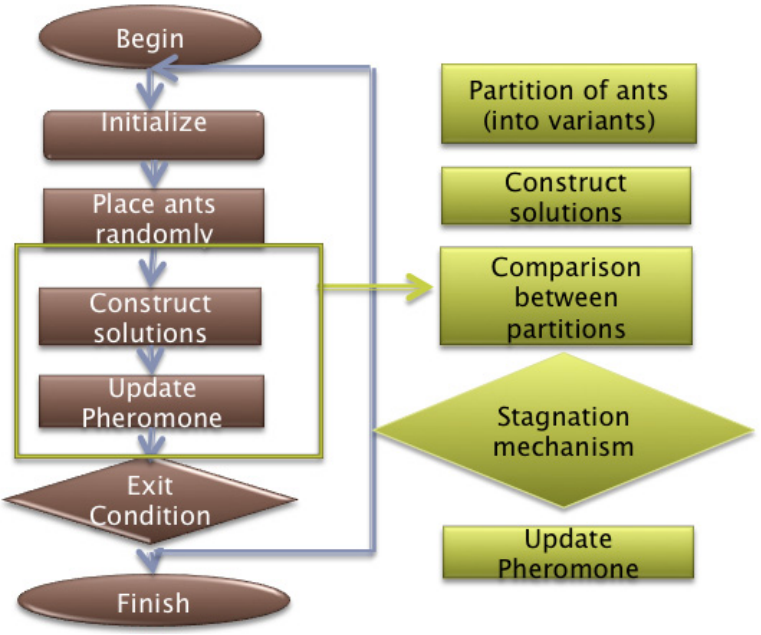


Fig. 2. Shows the proposed variation of the ACO algorithm (P-ACO)

cause that one by one the variations are stopped leaving only one "winner" variation at the end of the maximum number of iterations which will make the algorithm faster because it will perform the best variation(s), not wasting time in evaluating the ants in the variations with poor results.

2.1 Methodology

First we choose the desired parameters. For practical purposes we implemented an interface to choose our parameters and TSP instances.

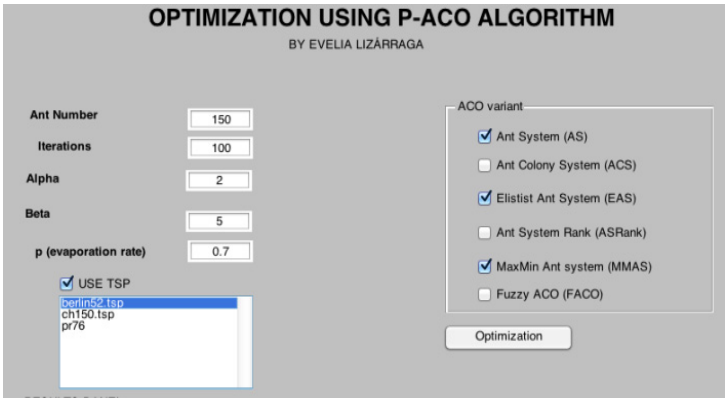


Fig. 3. Shows the mentioned interface

Subsequently the best ant of each partition is compared with each other; obtaining the best global ant *i* (global best *i*).

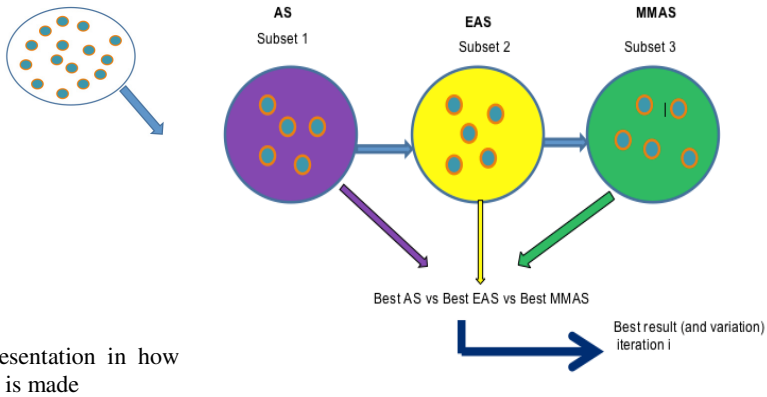


Fig.4. Representation in how the partition is made

The evaluation of the partitions is made sequentially in the same iteration *i*. Therefore in each iteration the best global ant *i* is compared with the best global ant *i*-1 following the conventional ACO algorithm. This allows us to compare in one iteration three different variations, saving time performing tests and having less overhead in comparison with the conventional method, which uses just one variation of ACO in all iterations. This means *n* (iterations) x 3 (variations) total tests in comparison with the proposed method, which only needs *n* tests. Because the proposed method selects one ant per iteration that represents also a variant of ACO; the end result provides the most used variant hence the one with best performance in a particular instance of TSP.

3 Experiments

We performed tests with different instances of TSP to test the algorithm using different parameters for two types of experiments.

- TSP Instances used: Berlin52 (52 cities) and bier127 (127 cities)
- 30 experiments with P-ACO
- 30 experiments with AS
- 30 experiments with EAS
- 30 experiments with MMAS
- We used α=2, β=5 and ρ=0.7 in every experiment.

- Hardware information:
  - Mac OS X 2.3 Ghz Intel Core i5 4GB 1333 Mhz DDR3

- Types of experiments:
  - Experiments type 1: consist in setting the parameters in 120 ants and 150 iterations.
  - Experiments type 2: consist in setting the parameters in 150 ants and 100 iterations.

An Experiment type 1 should take longer to perform than an experiment type 2.

4 Results

Variables used: It(Iterations), G.B. (Global Best), Avg. (Average), T(Time), BRKSF(Best Result Known so far) according to the literature [12], E (Error), N.E.(Normalized Error). The average is calculated with the global best of each experiment dived in the number of experiments (30). The error is calculated as the following:

Error=Best Result so Far-Average
Normalized Error=1-(Best Result so Far/Average)

4.1 Berlin52 (52 Cities)

We performed different tests using a TSP instances with 52 cities. Tables 2 and Table 3 shows the comparison between the proposed method and the conventional method, using experiments type 1. Table 4 and Table 5 are doing the same comparison using experiments type 2.

Table 2. Experiments type 1 using proposed algorithm P-ACO

Ants	It	G.B.	Avg.	V	T(min)	BRKSF	E	N.E
120	150	7549.29	7549.29	P-ACO (MMAS)	**14.2**	7542	7.29	**0.00096**

Table 3. Experiments type 1 using individual algorithms

Ants	It	G.B.	Avg.	V	T(min)	BRKSF	E	N.E
120	150	7713.03	7961.82	AS	37.95	7542	171.03	0.052729
120	150	7544.37	7628.35	EAS	37.53	7542	2.366	0.011321
120	150	7549.29	7944.86	MMAS	839.09	7542	7.29	0.050707
			Total time used		**914.57 min(15.24 hours)**	Average Error		**0.038252**

Table 4. Experiments type 2 using proposed algorithm P-ACO

Ants	It	G.B.	Avg.	V	T(min)	BRKSF	E	N.E
150	100	7544.37	7544.37	P-ACO (MMAS)	**15.2**	7542	2.37	**0.000314**

Table 5. Experiments type 2 using individual algorithms

Ants	It	G.B.	Avg.	V	T(min)	BRKSF	E	N.E
150	100	7857.17	7857.17	AS	23.62	7542	315.16	0.040112
150	100	7544.2	7688.23	EAS	31.44	7542	2.37	0.019020
150	100	7658.96	7941.22	MMAS	451.55	7542	116.96	0.050272
			Total time used		**506.61 min(8.44 hours)**	Average Error		**0.036468**

4.2 Bier127 (127 Cities)

We performed different tests using a TSP instances with 127 cities. Tables 6 and Table 7 shows the comparison between the proposed method and the conventional method, using experiments type 1. Table 8 and Table 9 are doing the same comparison using experiments type 2.

Table 6. Experiments type 1 using proposed algorithm P-ACO

Ants	It	G.B.	Avg.	V	T(min)	BRKSF	E	N.E
120	150	126551	126551.72	P-ACO (EAS)	**69.85 (1.16 hours)**	118282	8269	**0.065**

Table 7. Experiments type 1 using individual algorithms

Ants	It	G.B.	Avg.	V	T(min)	BRKSF	E	N.E
120	150	133400.8	136754.79	AS	149.97	118282	15118.8	0.135
120	150	125788.79	128061.79	EAS	136.20	118282	7506.79	0.076
120	150	126554.00	126781.43	MMAS	3071.89	118282.	8272	0.067
			Total time used		**3357.06 min (55.95hours)**	Average Error		**0.092**

Table 8. Experiments type 2 using proposed algorithm P-ACO

Ants	It	G.B.	Avg.	V	T(min)	BRKSF	E	N.E
150	100	123903	123912.33	P-ACO (EAS)	**50.46**	118282	5621	**0.045**

Table 9. Experiments type 2 using individual algorithms

Ants	It	G.B.	Avg.	V	T(min)	BKSF	E	N.E
150	100	134660.87	137786.55	AS	127.21	118282	16378.87	0.142
150	100	123903.00	123915.46	EAS	101.3	118282	5621.00	0.045
150	100	123903.00	123921.21	MMAS	1463.34	118282.	5621.00	0.045
			Total time used		**1691.85 min (28.19 hours)**	Average Error		**0.077**

5 Conclusions

We proposed a method for ant set partitioning in ACO. The proposed method is to divide the total number of ants into different partitions for a corresponding variant of ACO. Simulation results show that the proposed approach is working appropriately, in some cases 64 times faster than the conventional methodology and also providing better results. Performing two types of experiments varying the number of ants and iterations we made a comparison between the proposed algorithm P-ACO and the conventional methodology. In experiment of type 1 we used 120 ants and 150 iterations and in the type 2 we used 150 ants and 100 iterations for each instance of TSP. Using 52 cities (Berlin52) with the type 1 experiment we compared Table 2 with Table 3, in which the P-ACO reached a better result in global best and 64 faster, with a much lower average normalized error of 0.00096. In experiment of type 2 the comparison between Table 4 and Table 5 shows that P-ACO reached a better result 33 times faster, with a normalized error of 0.000314. Using 127 cities (bier127) with type 1 experiment we compared Table 6 and Table 7, where P-ACO showed a better global best 48 times faster with a average normalized error of 0.065. In experiment type 2 we compared Table 8 and Table 9 obtaining a better global best of 0.045, 33.5 times faster than the conventional method. These results clearly demonstrate the feasibility of divide the work between the ants for the evaluation, also providing a stagnation mechanism that controls the waste of computational calculations reducing only the work in the variations that shows good results.

References

[1] Engelbrecht, A.: Fundamentals of Computational Swarm Intelligence, pp. 85–131. Wiley, England (2005)

[2] Naredo, E., Castillo, O.: ACO-Tuning of a Fuzzy Controller for the Ball and Beam Problem. In: Batyrshin, I., Sidorov, G. (eds.) MICAI 2011, Part II. LNCS, vol. 7095, pp. 58–69. Springer, Heidelberg (2011)

[3] Castillo, O., Martinez-Marroquin, R., Soria, J.: Parameter Tuning of Membership Functions of a Fuzzy Logic Controller for an Autonomous Wheeled Mobile Robot Using Ant Colony Optimization. In: IEEE International Conference on Systems, Man and Cybernetics (2009)
[4] Dorigo, M., Stützle, T.: Ant Colony Optimization, Massachusetts Institute of Technology (2004)
[5] Dorigo., D.M., Birattari, M., Stützle, T.: Ant Colony Optimization. IEEE Computational Intelligence Magazine, 28–39 (November 2006)
[6] Ginidi-Ginidi, A.R., Kamel, A.M.A.M., Dorrah, H.T.: Development of New Fuzzy Logic-based Ant Colony Optimization Algorithm for Combinatorial Problems. In: 14th International Middle East Power Systems Conference (MEPCON 2010), December 19-21, Cairo University, Egypt (2010), Paper ID 294
[7] Eberhart, R.C., Kennedy, J.: A new optimizer using particle swarm theory. In: Proceedings of the Sixth International Symposium on Micro Machine and Human Science, pp. 39–43. IEEE service center, Piscataway (1995)
[8] Stützle, T., Hoos, H.H.: Improving the Ant System: A detailed report on the MAX-MIN Ant System. Technical report AIDA-96-12, FG Intellektik, FB Informatik, TU Darmstadt, Germany (1996)
[9] Stützle, T., Hoos, H.H.: MAX-MIN Ant System and local search for combinatorial optimization problems. In: Voss, S., Martello, S., Osman, I., Roucairol, C. (eds.) Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization, pp. 137–154. Kluwer Academic Publishers, Dordrecht (1999)
[10] Stüttzle, T., Hoos, H.H.: MAX-MIN Ant System. Future Generation Computer Systems 16(8), 889–914 (2000)
[11] Stützle, T., Linke, S.: Experiments with variants of ant algorithms. Mathware and Soft Computing 9(2-3), 193–207 (2002)
[12] TSPLIB Official page of TSP, `http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/STSP.html`

Particle Swarm Optimization with Dynamic Parameter Adaptation Using Fuzzy Logic for Benchmark Mathematical Functions

Frumen Olivas and Oscar Castillo

Tijuana Institute of Technology,
Unidad Tomas Aquino, Tijuana, México
ocastillo@tectijuana.mx

Abstract. In this paper a new method for dynamic parameter adaptation in particle swarm optimization (PSO) is proposed. PSO is a metaheuristic inspired in social behaviors, which is very useful in optimization problems. In this paper we propose an improvement to the convergence and diversity of the swarm in PSO using fuzzy logic. Simulation results show that the proposed approach improves the performance of PSO.

Keywords: Fuzzy Logic, Particle Swarm Optimization, Dynamic parameter adaptation.

1 Introduction

Fuzzy logic or multi-valued logic is based on fuzzy set theory proposed by Zadeh [9], which helps us in modeling knowledge, through the use of if-then fuzzy rules.

The fuzzy set theory provides a systematic calculus to deal with linguistic information, and that improves the numerical computation by using linguistic labels stipulated by membership functions [5].

Particle swarm optimization (PSO) that was introduced by Kennedy and Eberhart in 1995 [6, 7], maintains a swarm of particles and each particle represents a possible solution. These particles "fly" through a multidimensional search space, where the position of each particle is adjusted according to your own experience and that of its neighbors [3].

PSO has had many proposed improvements and applications. Most of the modifications to PSO are to improve convergence and to increase the diversity of the swarm [3]. So in this paper we propose an improvement to the convergence and diversity of PSO through the use of fuzzy logic. Basically, fuzzy rules are used to control the key parameters in PSO to achieve the best possible dynamic adaptation of these parameter values.

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 247–258.
springerlink.com

The rest of the paper is organized as follows. Section 2 describes the proposed methodology. Section 3 shows how the experiments were performed with the proposed method and the simple method using the benchmark functions defined in section 2. Section 4 shows how to perform the statistical comparison with all its parameters and analysis of results. Finally, the conclusions of this paper are presented.

2 Methodology

The dynamics of PSO is defined by Equations 1 and 2, which are the equations to update the position and velocity of the particle, respectively.

$$x_i(t+1) = x_i(t) + v_i(t+1) \tag{1}$$

$$v_{ij}(t+1) = v_{ij}(t) + c_1 r_1(t)[y_{ij}(t) - x_{ij}(t)] + c_2 r_{2j}(t)[\hat{y}_j(t) - x_{ij}(t)] \tag{2}$$

Parameters c1 and c2 were selected to be adjusted using fuzzy logic, since those parameters account for the movement of the particles.

The parameter c1 or cognitive factor represents the level of importance given the particle to its previous positions.

The parameter c2 or social factor represents the level of importance that the particle gives the best overall position.

Based on the literature [3] the recommended values for c1 and c2 must be in the range of 0.5 and 2.5, plus it is also suggested that changing the parameters c1 and c2 dynamically during the execution of this algorithm can produce better results.

In addition it is also found that the algorithm performance measures, such as: diversity of the swarm, the average error at one point in the execution of the algorithm, the iterations themselves, need to be considered to run the algorithm, among others. In our work all the above are taken in consideration for the fuzzy systems to modify the parameters c1 and c2 dynamically changing these parameters in each iteration of the algorithm.

For measuring the iterations of the algorithm, it was decided to use a percentage of iterations, i.e. when starting the algorithm the iterations will be considered "low", and when the iterations are completed it will be considered "high" or close to 100%. To represent this idea we use Equation 3.

$$Iteration = \frac{Current\ Iteration}{Maximum\ of\ Iterations} \tag{3}$$

The diversity measure is defined by Equation 4, which measures the degree of dispersion of the particles, i.e. when the particles are closer together there is less diversity as well as when particles are separated then diversity is high. As the reader will realize the equation of diversity can be considered as the average of the Euclidean distances between each particle and the best particle.

$$Diversity(S(t)) = \frac{1}{n_s}\sum_{i=1}^{n_s}\sqrt{\sum_{j=1}^{n_x}(x_{ij}(t) - \bar{x}_j(t))^2} \tag{4}$$

The error measure is defined by equation 5, which measures the difference between the swarm and the best particle, by averaging the difference between the fitness of each particle and the fitness of the best particle.

$$Error = \frac{1}{n_s}\sum_{i=1}^{n_s}(Fitness(x_i) - MinF) \tag{5}$$

Therefore for designing the fuzzy systems, which dynamically adjust the parameters of c1 and c2, the three measures described above were considered as inputs. It is obvious that for each fuzzy system the outputs are c1 and c2.

In regards to the inputs of the fuzzy systems, the iteration variable has by itself a defined range of possible values which range from 0 to 1 (0 is 0% and 1 is the 100%), but with the diversity and the error, we perform a normalization of the values of these to have values between 0 and 1. Equation 6 shows how the normalization of diversity is performed and Equation 7 shows how the normalization of the error is obtained.

$$DiverNorm = \begin{cases} if\ MinDiver = MaxDiver\ \{DiverNorm = 0 \\ if\ MinDiver \neq MaxDiver\ \{DiverNorm = FnNorm \end{cases}$$

$$FnNorm = \frac{Diversity - MinDiver}{MaxDiver - MinDiver} \tag{6}$$

Equation 6 shows two conditions for the normalization of diversity, the first provides that where the maximum Euclidean distance is equal to the minimum Euclidean distance, this means that the particles are exactly in the same position so there is no diversity. The second condition deals with the cases with different Euclidean distances.

$$ErrorNorm = \begin{cases} if\ MinF = MaxF\ \{ErrorNorm = 1 \\ if\ MinF \neq MaxF\ \{ErrorNorm = \frac{Error - MinF}{MaxF - MinF} \end{cases} \tag{7}$$

Equation 7 shows two conditions to normalize the error, the first one tells us that when the minimum fitness is equal to the maximum fitness, then the error will be 1; this is because the particles are close together. The second condition deals with the cases with different fitness.

The design of the input variables can be seen in Figures 1, 2 and 3, which show the inputs iteration, diversity, and error respectively, each input is granulated into three triangular membership functions.

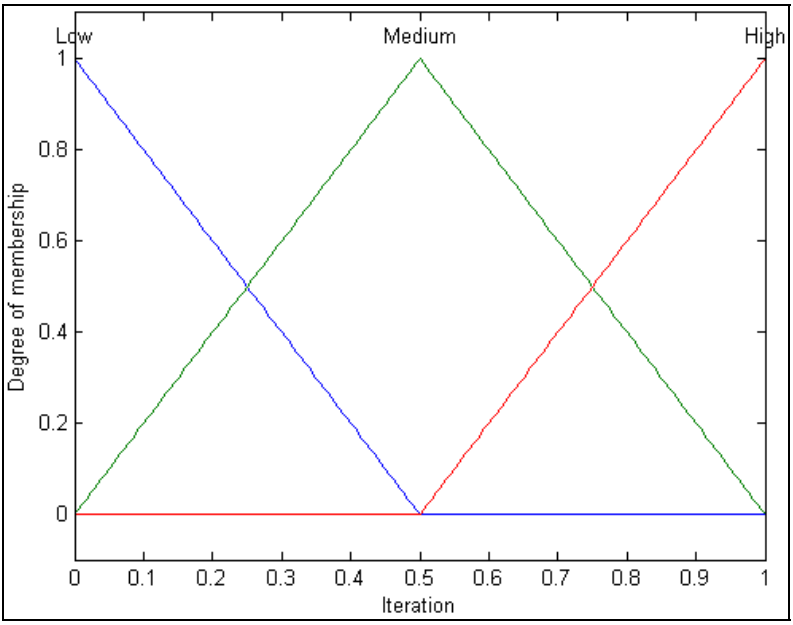


Fig. 1. Input 1: iteration

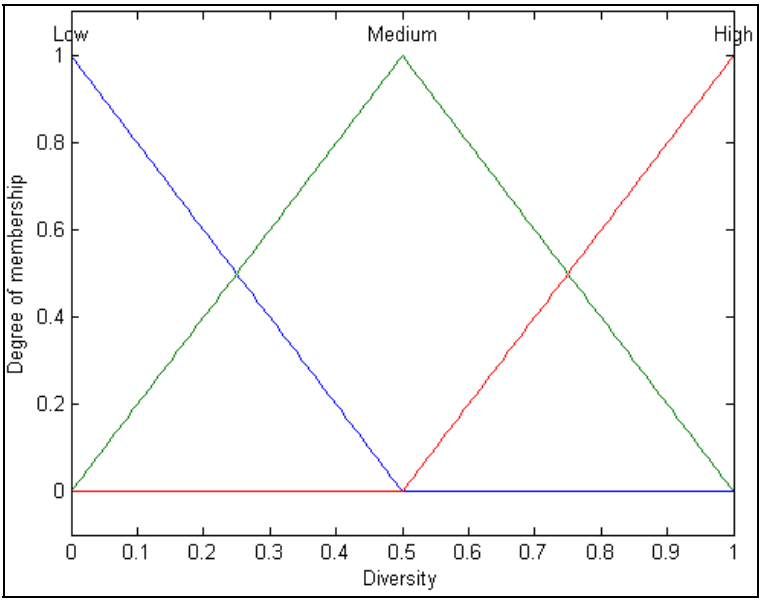


Fig. 2. Input 2: diversity

For the output variables, as mentioned above, the recommended values for c1 and c2 are between 0.5 and 2.5, so that the output variables were designed using this range of values. Each output is granulated in five triangular membership functions, the design of the output variables can be seen in Figures 4 and 5, c1 and c2 respectively.

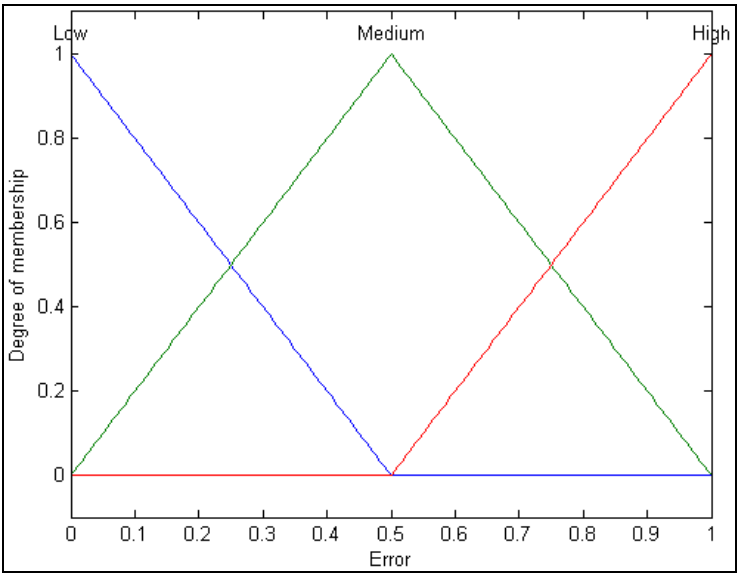


Fig. 3. Input 3: error

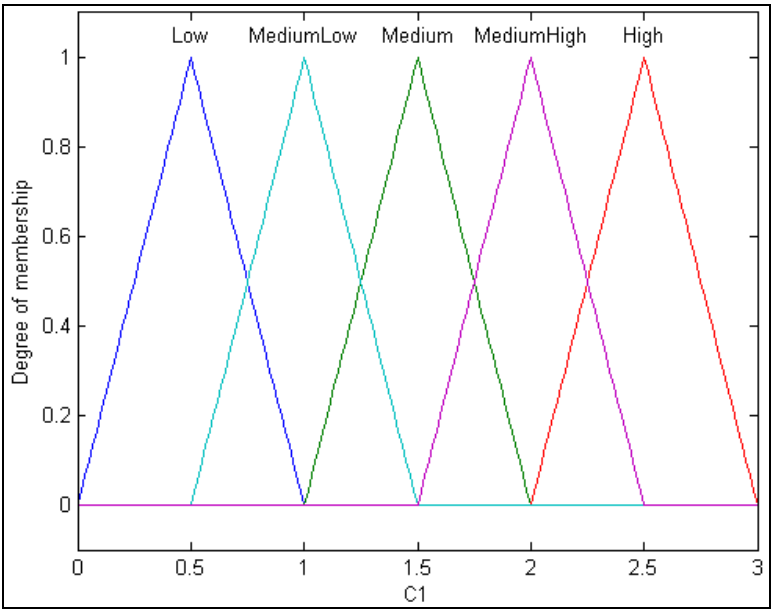


Fig. 4. Output 1: c1

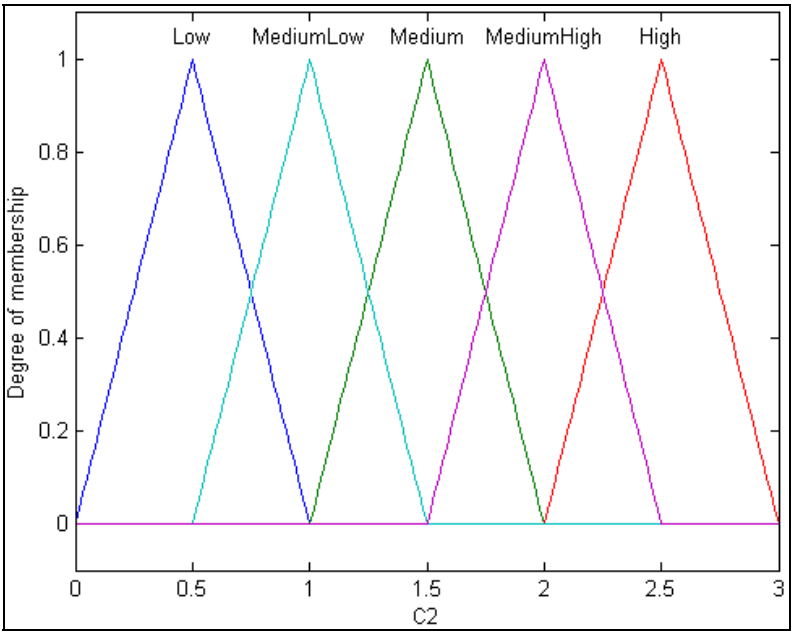


Fig. 5. Output 2: c2

Having defined the possible input variables, it was decided to combine them to generate different fuzzy systems for dynamic adjustment of c1 and c2. Based on the combinations of possible inputs, there were seven possible fuzzy systems, but it was decided to consider only the systems that have more inputs (since we previously considered fuzzy systems with only a single input), so that eventually there were three fuzzy systems which are defined below.

The first fuzzy system has iteration and diversity as inputs, which is shown in Figure 6. The second fuzzy system has iteration and error as inputs and is shown in Figure 7. The third fuzzy system has iteration, diversity, and error as inputs, as shown in Figure 8.

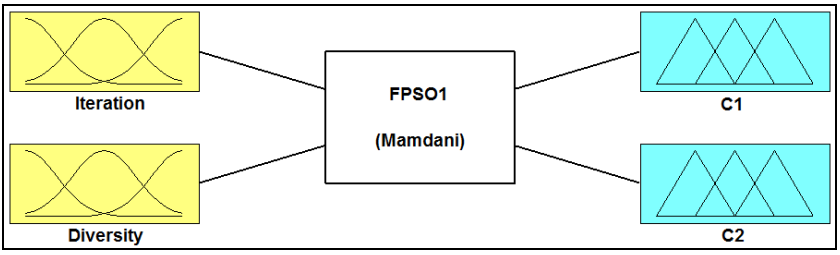


Fig. 6. First fuzzy system

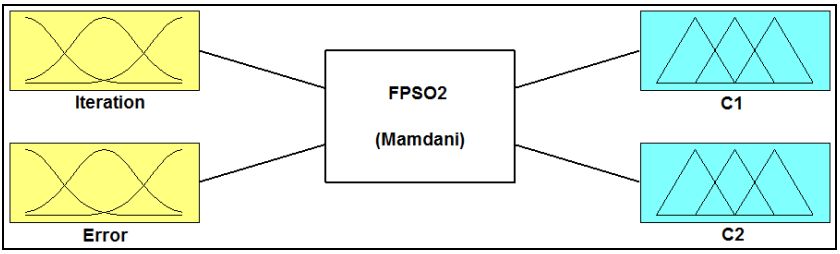


Fig. 7. Second fuzzy system

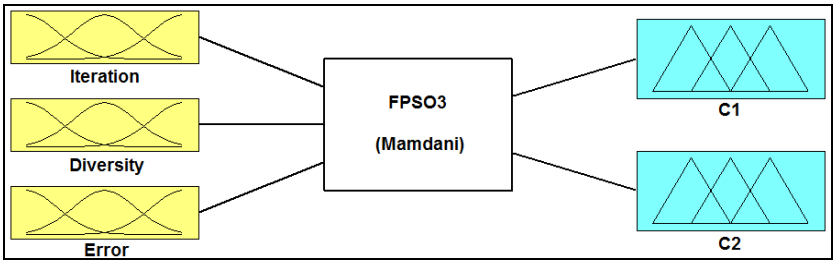


Fig. 8. Third fuzzy system

To design the rules of each fuzzy system, it was decided that in early iterations the PSO algorithm must explore and eventually exploit. Taking into account other variables such as diversity, for example, when diversity is low, that is, that the particles are close together, we must use exploration, and when diversity is high we must use exploitation.

The rules for each fuzzy system are shown in Figures 9, 10 and 11, for the fuzzy systems 1, 2 and 3, respectively.

1. If (Iteration is Low) and (Diversity is Low) then (C1 is High)(C2 is Low)
2. If (Iteration is Low) and (Diversity is Medium) then (C1 is MediumHigh)(C2 is Medium)
3. If (Iteration is Low) and (Diversity is High) then (C1 is MediumHigh)(C2 is MediumLow)
4. If (Iteration is Medium) and (Diversity is Low) then (C1 is MediumHigh)(C2 is MediumLow)
5. If (Iteration is Medium) and (Diversity is Medium) then (C1 is Medium)(C2 is Medium)
6. If (Iteration is Medium) and (Diversity is High) then (C1 is MediumLow)(C2 is MediumHigh)
7. If (Iteration is High) and (Diversity is Low) then (C1 is Medium)(C2 is High)
8. If (Iteration is High) and (Diversity is Medium) then (C1 is MediumLow)(C2 is MediumHigh)
9. If (Iteration is High) and (Diversity is High) then (C1 is Low)(C2 is High)

Fig. 9. Rules for fuzzy system 1

Also for the comparison of the proposed method with respect to the PSO without parameter adaptation, we considered benchmark mathematical functions, defined in [4, 8], which are 27 in total, and in each we must find the parameters that give us the global minimum of each function. In Figure 12 there is a sample of the functions that are used.

1. If (Error is Low) and (Iteration is Low) then (C1 is Low)(C2 is Medium)
2. If (Error is Low) and (Iteration is Medium) then (C1 is MediumLow)(C2 is MediumHigh)
3. If (Error is Low) and (Iteration is High) then (C1 is Low)(C2 is High)
4. If (Error is Medium) and (Iteration is Low) then (C1 is MediumLow)(C2 is MediumHigh)
5. If (Error is Medium) and (Iteration is Medium) then (C1 is Medium)(C2 is Medium)
6. If (Error is Medium) and (Iteration is High) then (C1 is Medium)(C2 is High)
7. If (Error is High) and (Iteration is Low) then (C1 is High)(C2 is MediumLow)
8. If (Error is High) and (Iteration is Medium) then (C1 is MediumHigh)(C2 is Medium)
9. If (Error is High) and (Iteration is High) then (C1 is High)(C2 is Low)

Fig. 10. Rules for fuzzy system 2

1. If (Iteration is Low) and (Diversity is Low) and (Error is Low) then (C1 is MediumLow)(C2 is Low)
2. If (Iteration is Low) and (Diversity is Low) and (Error is Medium) then (C1 is MediumHigh)(C2 is Low)
3. If (Iteration is Low) and (Diversity is Low) and (Error is High) then (C1 is High)(C2 is Low)
4. If (Iteration is Low) and (Diversity is Medium) and (Error is Low) then (C1 is Medium)(C2 is Medium)
5. If (Iteration is Low) and (Diversity is Medium) and (Error is Medium) then (C1 is MediumHigh)(C2 is MediumLow)
6. If (Iteration is Low) and (Diversity is Medium) and (Error is High) then (C1 is MediumHigh)(C2 is Low)
7. If (Iteration is Low) and (Diversity is High) and (Error is Low) then (C1 is Low)(C2 is MediumLow)
8. If (Iteration is Low) and (Diversity is High) and (Error is Medium) then (C1 is Medium)(C2 is Medium)
9. If (Iteration is Low) and (Diversity is High) and (Error is High) then (C1 is MediumLow)(C2 is Low)
10. If (Iteration is Medium) and (Diversity is Low) and (Error is Low) then (C1 is Medium)(C2 is Medium)
11. If (Iteration is Medium) and (Diversity is Low) and (Error is Medium) then (C1 is MediumHigh)(C2 is MediumLow)
12. If (Iteration is Medium) and (Diversity is Low) and (Error is High) then (C1 is MediumHigh)(C2 is Low)
13. If (Iteration is Medium) and (Diversity is Medium) and (Error is Low) then (C1 is MediumLow)(C2 is MediumHigh)
14. If (Iteration is Medium) and (Diversity is Medium) and (Error is Medium) then (C1 is Medium)(C2 is Medium)
15. If (Iteration is Medium) and (Diversity is Medium) and (Error is High) then (C1 is MediumHigh)(C2 is MediumLow)
16. If (Iteration is Medium) and (Diversity is High) and (Error is Low) then (C1 is Low)(C2 is MediumHigh)
17. If (Iteration is Medium) and (Diversity is High) and (Error is Medium) then (C1 is MediumLow)(C2 is MediumHigh)
18. If (Iteration is Medium) and (Diversity is High) and (Error is High) then (C1 is Medium)(C2 is Medium)
19. If (Iteration is High) and (Diversity is Low) and (Error is Low) then (C1 is Low)(C2 is MediumLow)
20. If (Iteration is High) and (Diversity is Low) and (Error is Medium) then (C1 is Medium)(C2 is Medium)
21. If (Iteration is High) and (Diversity is Low) and (Error is High) then (C1 is MediumLow)(C2 is Low)
22. If (Iteration is High) and (Diversity is Medium) and (Error is Low) then (C1 is Low)(C2 is MediumHigh)
23. If (Iteration is High) and (Diversity is Medium) and (Error is Medium) then (C1 is MediumLow)(C2 is MediumHigh)
24. If (Iteration is High) and (Diversity is Medium) and (Error is High) then (C1 is Medium)(C2 is Medium)
25. If (Iteration is High) and (Diversity is High) and (Error is Low) then (C1 is Low)(C2 is High)
26. If (Iteration is High) and (Diversity is High) and (Error is Medium) then (C1 is Low)(C2 is MediumHigh)
27. If (Iteration is High) and (Diversity is High) and (Error is High) then (C1 is Low)(C2 is MediumLow)

Fig. 11. Rules for fuzzy system 3

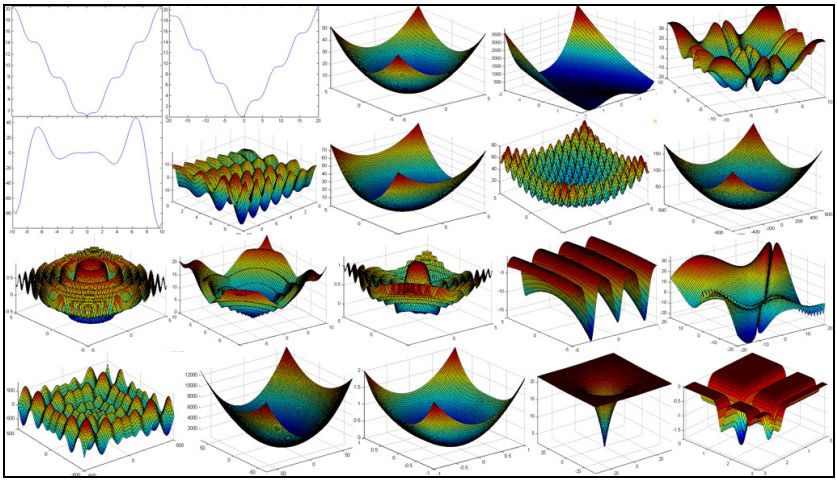


Fig. 12. Benchmark mathematical functions

As indicated in Figure 12 we only considered functions of one or two dimensions for the experiments.

So that once defined the fuzzy systems that dynamically adjust the parameters of PSO, and defined the problem in which it applies (Benchmark mathematical functions), the proposal is as shown in Figure 13, where we can notice that c1 and c2 parameters are adjusted by a fuzzy system, and in turn this "fuzzy PSO" searches for the optimal parameters for the Benchmark mathematical functions.

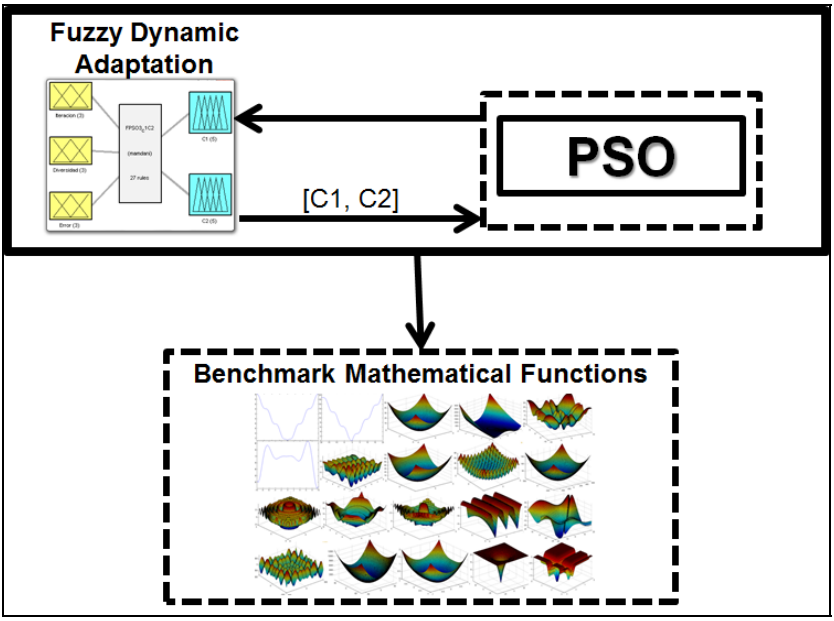


Fig. 13. Proposal

3 Experimentation with the Fuzzy Systems and the Benchmark Mathematical Functions

For the experiments we used the parameters contained in Table 1. Table 1 shows the parameters of the methods to be compared; in this case, we perform a comparison of the proposed method and its variations against the simple PSO algorithm.

Table 1. Parameters for each method

Parameter	Simple PSO	Fuzzy PSO 1	Fuzzy PSO 2	Fuzzy PSO 3
Population	10	10	10	10
Iterations	30	30	30	30
C1	1	Dynamic	Dynamic	Dynamic
C2	3	Dynamic	Dynamic	Dynamic

Since functions do not have the same global minimum, for comparison it was decided to normalize the results of each function, for this it is used equation 8, which gives results between 0 and 1, which means that a number close to 0 is better than a number close to 1.

$$ExprerimentNorm = \left| \frac{Experiment - GlobalMin}{GlobalMax - GlobalMin} \right| \quad (8)$$

To normalize the results with equation 8, we need the maximum and the minimum of each benchmark mathematical function; in our case these data are known. Also, the absolute value is needed, because we want to know how much difference between the results of the experiment and the minimum value of the function.

Therefore, Table 2 shows some experimental results of each method with each function.

Table 2. Simulation

Function	Minimum	Simple PSO	Fuzzy PSO 1	Fuzzy PSO 2	Fuzzy PSO 3
1	1	0.0005	0.0000	0.0001	0.0003
2	0	0.0000	0.0000	0.0000	0.0000
3	0	0.0000	0.0009	0.0000	0.0000
4	0	0.0000	0.0000	0.0000	0.0000
5	-20	0.1042	0.0665	0.0728	0.0743
6	-100.2238	0.1275	0.1277	0.0000	0.0000
7	-18.5547	0.1929	0.2484	0.2645	0.1253
8	0	0.0000	0.0000	0.0003	0.0000
9	0	0.0017	0.0039	0.0157	0.0019
10	0	0.0000	0.0000	0.0001	0.0000

4 Statistical Comparison

To perform the statistical comparison, we have:

3 methods to compare against the simple PSO, (FPSO1, FPSO2, FPSO3).

27 Benchmark mathematical functions.

10 experiments were performed for each method by each function, so it has a total of 270 experiments for each method. Of this total, we took a random sample of 50 experiments for each method for statistical comparison.

The statistical test used for comparison is the z-test, whose parameters are defined in Table 3.

Table 3. Parameters for statistical z-test

Parameter	Value
Level of Significance	95%
Alpha	0.05%
Ha	$\mu 1 < \mu 2$
H0	$\mu 1 \geq \mu 2$
Critical Value	-1.645

With the parameters in Table 3, we applied the statistical z-test, giving the following results:

Table 4. Results of applying statistical z-test

Our Method	Simple Method	Z Value	Evidence
FPSO1	Simple PSO	-2.1937	Significant
FPSO2	Simple PSO	-0.6801	Not Significant
FPSO3	Simple PSO	-2.1159	Significant

In applying the statistic z-test, with significance level of 0.05, and the alternative hypothesis says that the average of the proposed method is lower than the average of simple PSO, and of course the null hypothesis tells us that the average of the proposed method is greater than or equal to the average of simple PSO, with a rejection region for all values fall below -1.645. So the statistical test results are that: for the fuzzy PSO 1, there is significant evidence to reject the null hypothesis, as in the fuzzy PSO 3. But in the fuzzy PSO 2, there is no significant evidence to reject the null hypothesis. In conclusion, two of the proposed variants of PSO were significantly better than simple PSO.

5 Conclusions

We proposed a method for dynamic adaptation of the parameters of PSO to improve the quality of results. With the results of the statistic test, we can conclude that there is significant evidence to say that the proposed approach could help in the adaptation of parameters in PSO.

Future work includes experiments with functions with more than two dimensions, comparison with other approaches of PSO, for example, PSO with inertia weight and PSO with constriction. Also try to achieve better results for the PSO with fuzzy system 2, more specifically with the input error and the rules of the fuzzy system. In future work also we try to apply the proposed method to other types of problems, for example, optimization of fuzzy systems.

References

[1] Abdelbar Ashraf, M., Abdelshahid, S., Wunsch, D.C.: Fuzzy PSO: A Generalization of Particle Swarm Optimization. In: Proceedings 2005 IEEE International Joint Conference on Neural Networks, vol. 2, pp. 1086–1091 (2005)
[2] Juana, A.S.: Optimización por nube de partículas (PSO) de controladores difusos para robots autónomos móviles. Master's thesis at Tijuana Institute of Technology (2011)
[3] Engelbrecht Andries, P.: Fundamentals of Computational Swarm Intelligence. University of Pretoria, South Africa (2005)
[4] Haupt Randy, L., Ellen, H.S.: Practical Genetic Algorithms, 2nd edn. A Wiley-Interscience publication, New York (2004)
[5] Jang, J.-S.R., Sun, C.-T., Mizutani, E.: Neuro-fuzzy and soft computing: a computational approach to learning and machine intelligence. Prentice-Hall, Upper Saddle River (1997)
[6] Kennedy, J., Eberhart, R.C.: Swarm Intelligence. Morgan Kaufmann, San Francisco (2001)
[7] Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: Proc. IEEE Int. Conf. on Neural Networks, IV, pp. 1942–1948. IEEE Service Center, Piscataway (1995)
[8] Molga, M., Smutnicki, C.: Test functions for optimization needs (2005)
[9] Zadeh, L.: Fuzzy sets. Information & Control 8, 338–353 (1965)
[10] Valdez, F., Melin, P., Castillo, O.: An improved evolutionary method with fuzzy logic for combining Particle Swarm Optimization and Genetic Algorithms. Appl. Soft Comput. 11(2), 2625–2632 (2011)

Dynamic Fuzzy Logic Parameter Tuning for ACO and Its Application in TSP Problems

Héctor Neyoy, Oscar Castillo, and José Soria

Tijuana Institute of Technology, Tijuana, México
ocastillo@tectijuana.mx

Abstract. Ant Colony Optimization (ACO) is a population-based constructive metaheuristic that exploits a form of past performance memory inspired by the foraging behavior of real ants. The behavior of the ACO algorithm is highly dependent on the values defined for its parameters. Adaptation and parameter control are recurring themes in the field of bio-inspired algorithms. The present paper explores a new approach of diversity control in ACO. The central idea is to avoid or slow down full convergence through the dynamic variation of the alpha parameter. The performance of different variants of the ACO algorithm was observed to choose one as the basis to the proposed approach. A convergence fuzzy logic controller with the objective of maintaining diversity at some level to avoid premature convergence was created. Encouraging results on several travelling salesman problem (TSP) instances are presented with the proposed method.

1 Introduction

ACO is inspired by the foraging behavior of ant colonies, and targets discrete optimization problems [1].

The behavior of the ACO algorithm is highly dependent on the values defined for its parameters and has an effect on its convergence. Often these are kept static during the execution of the algorithm. Changing the parameters at runtime, at a given time or depending on the search progress may improve the performance of the algorithm [3], [4], [5].

Control the dynamics of convergence to maintain a balance between exploration and exploitation is critical for good performance.

Early convergence leaves large sections of the search space unexplored. Slow convergence does not concentrate its attention on areas where good solutions were found.

This paper explores a new method of diversity control in ACO. The central idea is to prevent or stop the total convergence through the dynamic adjustment of the parameter alpha.

The rest of the paper is organized as follows. Section 2 presents an overview of ACO. Section 3 describes a performance analysis on several TSP instances. Section 4 presents a new method of parameter tuning through fuzzy logic, finally section 5 and section 6 shows some simulation results and conclusions respectively.

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 259–271.
springerlink.com

2 Ant Colony Optimization (ACO)

The first ACO algorithm was called Ant System (AS) and its objective was to solve the traveling salesman problem (TSP), whose goal is to find the shortest route to link a number of cities. In each iteration, each ant keeps adding components to build a complete solution, the next component to be added is chosen with respect to a probability that depends on two factors. The pheromone factor that reflects the past experience of the colony and the heuristic factor that evaluates the interest of selecting a component with respect to an objective function. Both factors weighted by the parameters α and β respectively (1).

$$P_{ij}^{k} = \frac{\left[\tau_{ij}\right]^{\alpha}\left[\eta_{ij}\right]^{\beta}}{\sum_{l} \in N_i^k [\tau_{il}]^{\alpha}[\eta_{il}]^{\beta}}, \qquad if\ j \in N_i^k \tag{1}$$

After all ants have built their tours, pheromone trails are updated. This is done by lowering the pheromone value on all arcs by a constant factor (2), which prevents the unlimited accumulation of pheromone trails and allows the algorithm to forget bad decisions previously taken.

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij}, \qquad \forall (i,j) \in L \tag{2}$$

And by depositing pheromone on the arcs that ants have crossed in its path (3). The better the tour the greater the amount of pheromone that their arcs receive.

$$\begin{gathered} \tau_{ij} \leftarrow \tau_{ij} + \sum_{k=1}^{n} \Delta\, \tau_{ij}^{k}, \qquad \forall (i,j) \in L \\ \Delta\tau_{ij}^{k} = \begin{cases} \dfrac{1}{C^k}, if\ arc\ (i,j)\ belongs\ to\ T^k; \\ 0, otherwise; \end{cases} \end{gathered} \tag{3}$$

A first improvement on the initial AS, called the elitist strategy for Ant System (EAS). The idea is to provide strong additional reinforcement to the arcs belonging to the best tour found since the start of the algorithm (4)[1].

$$\begin{gathered} \tau_{ij} \leftarrow \tau_{ij} + \sum_{k=1}^{n} \Delta\, \tau_{ij}^{k} + e\Delta\tau_{ij}^{bs}, \ \ \forall (i,j) \in L \\ \Delta\tau_{ij}^{bs} = \begin{cases} \dfrac{1}{C^{bs}}, if\ arc\ (i,j)\ belongs\ to\ T^{bs}; \\ 0, otherwise; \end{cases} \end{gathered} \tag{4}$$

Another improvement over AS is the rank-based version of AS (ASRank). In ASrank each ant deposits an amount of pheromone that decreases with its rank. Additionally, as in EAS, the best-so-far ant always deposits the largest amount of pheromone in each iteration [1].

$$\tau_{ij} \leftarrow \tau_{ij} + \sum_{r=1}^{w-1} (w-r)\Delta\,\tau_{ij}^{r} + \Delta\tau_{ij}^{bs} \tag{5}$$

3 Performance Analysis of ACO

To observe the performance of the AS, EAS and ASRank ACO variants 30 experiments were performed by method for each instance of the examined TSP (Table 1), which are in the range of 14 to 100 cities, all extracted from TSPLIB [6], using the parameters recommended by the literature (Table 2) [1].

Table 1. TSP instances considered

TSP	Number of cities	Best tour length
Burma14	14	3323
Ulysses22	22	7013
Berlin52	52	7542
Eil76	76	538
kroA100	100	21282

Table 2. Parameters used for each ACO algorithm

ACO	α	β	ρ	m	τ_0
AS	1	2	0.5	n	m/C^{nn}
ASRank	1	2	0.1	n	$0.5r(r-1)/\rho C^{nn}$
EAS	1	2	0.5	n	$(e+m)/\rho C^{nn}$

m = n
C^{nn} = 20 for each tsp except burma14 where C^{nn} = 10.
EAS: e = 6
ASRank: r = w – 1; w = 6.

The behavior of AS and EAS was very similar in all experiments (Tables 3, 4, 5, 6, 7), the performance of the three variants began to worsen by increasing the problem complexity, however ASRank performance decreased to a lesser extent than their counterparts when the number of cities was greater than 50 (Tables 5, 6, 7).

Table 3. Performance obtained for the TSP instance Burma14

ACO	Best	Average	Successful runs
AS	3323	3323	30/30
ASRank	3323	3329	19/30
EAS	3323	3323	30/30

Table 4. Performance obtained for the TSP instance Ulysses22

ACO	Best	Average	Successful runs
AS	7013	7022	30/30
ASRank	7013	7067	19/30
EAS	7013	7018	30/30

Table 5. Performance obtained for the TSP instance Berlin52

ACO	Best	Average	Successful runs
AS	7542	7557	2/30
ASRank	7542	7580	17/30
EAS	7542	7554	6/30

Table 6. Performance obtained for the TSP instance Eil76

ACO	Best	Average	Successful runs
AS	547	556	0/30
ASRank	538	543	1/30
EAS	544	555	0/30

Table 7. Performance obtained for the TSP instance KroA100

ACO	Best	Average	Successful runs
AS	22305	22483	0/30
ASRank	21304	21549	0/30
EAS	22054	22500	0/30

Since ASRank had more success finding the minimum and scored lower averages with more complex TSP instances than the other approaches discussed (Figure 1, 2). It can be concluded that AS and EAS have better performance when the number of cities is low unlike ASRank that works best when the number of cities is not so small due to the pheromone deposit mechanism of this approach, where only the w-1 ants with the shorter tours and the ant with the best so far tour are allowed to deposit pheromone. This strategy can lead to a stagnation situation where all the ants follow the same path and construct the same tour [1] as a result of excessive increase in the pheromone trails of suboptimal routes (Figures 3, 4).

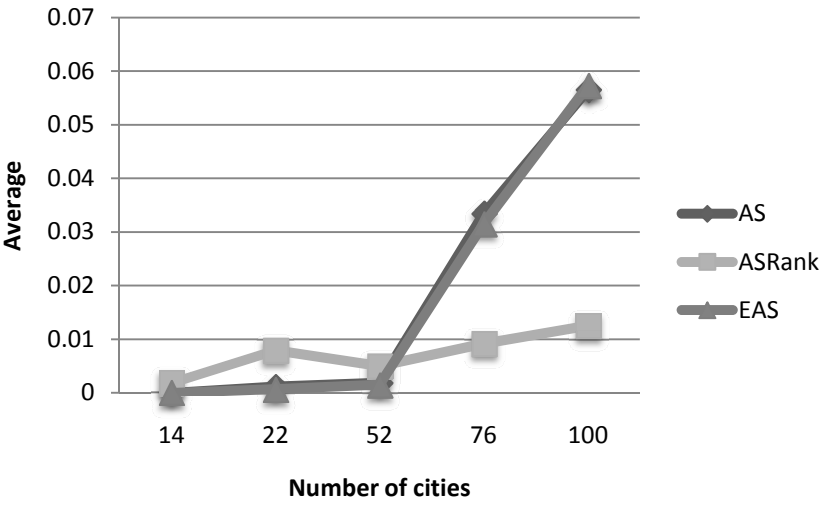


Fig. 1. Average results of each approach discussed

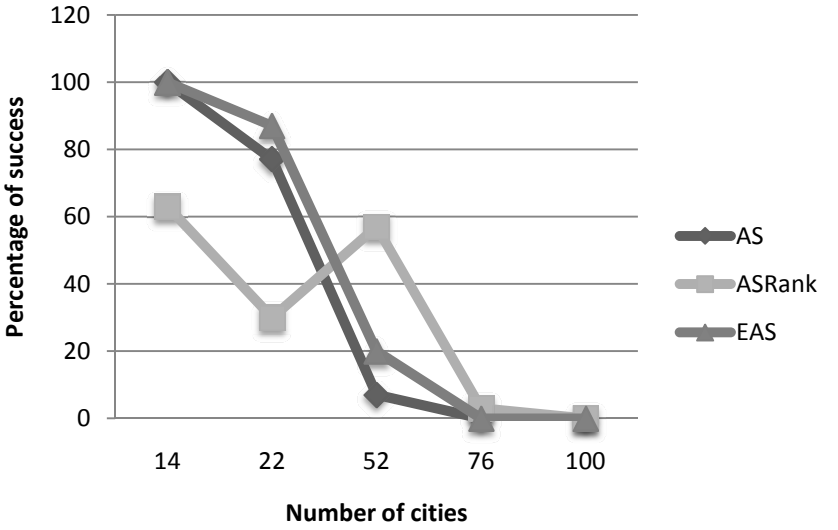


Fig. 2. Percentage of success in finding the global minimum of each approach discussed

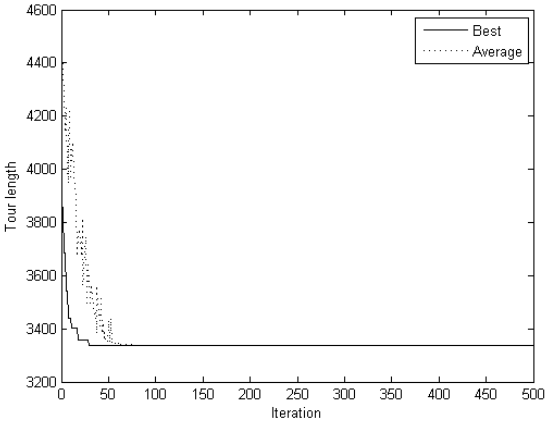


Fig. 3. Convergence plot of the ACO algorithm variant ASRank

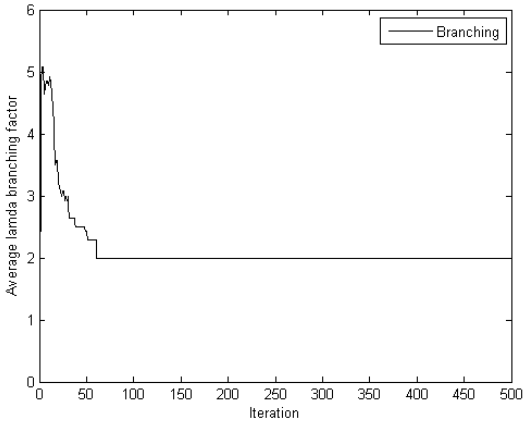


Fig. 4. Behavior of the average lambda branching factor during the execution of the algorithm ACO variant ASRank

4 Fuzzy Logic Convergence Controller

Due to the obtained results it was decided to use ASRank as the basis for our proposed ACO variant. The central idea is to prevent or stop the total convergence through the dynamic variation of the alpha parameter.

Alpha has a big effect in the diversity. Is recommended to keep α in the range of $0< \alpha< 1$[1].

A value closer to 1 will emphasize better paths but reduce diversity, while lower α will keep more diversity but reduce selective pressure [6].

However, it appears impossible to fix a universally best α. In most approaches it is taken to be 1, so that the selection probability is linear in the pheromone level.

An adaptive parameter control strategy was used; this takes place when there is some form of feedback from the search that is used to determine the direction and/or magnitude of the change to the strategy parameter [8]. In our case, the average lambda branching factor, this factor measures the distribution of the values of the pheromone trails and provides an indication of the size of the search space effectively explored [1].

A convergence fuzzy controller to prevent or delay the full convergence of the algorithm was created (Figure 5). Fuzzy control can be seen as the translation of external performance specifications and observations of a plant behavior into a rule based linguistic control strategy [7].

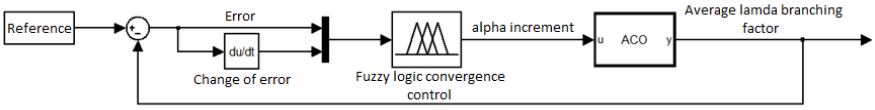


Fig. 5. Block diagram of the proposed system to control the convergence of the ACO algorithm variant ASRank

The objective of the controller is maintain the average lambda branching factor at a certain level to avoid a premature convergence, so its rules were made to fulfill this purpose (Figure 6).

If (error is P) and (error_change is P) then (alpha increment is N)
If (error is N) and (error_change is N) then (alpha increment is P)
If (error is P) and (error_change is Z) then (alpha increment is N)
If (error is N) and (error_change is Z) then (alpha increment is P)
If (error is P) and (error_change is N) then (alpha increment is Z)
If (error is N) and (error_change is P) then (alpha increment is Z)
If (error is Z) and (error_change is Z) then (alpha increment is Z)
If (error is Z) and (error_change is N) then (alpha increment is P)
If (error is Z) and (error_change is P) then (alpha increment is N)

Fig. 6. Rules of the proposed fuzzy system to control the convergence of the ACO algorithm

The controller uses as inputs the error (Figure 7) and change of error (Figure 8) with respect to an average lambda branching factor reference level and provides as output an increase in the value of parameter alpha (Figure 9).

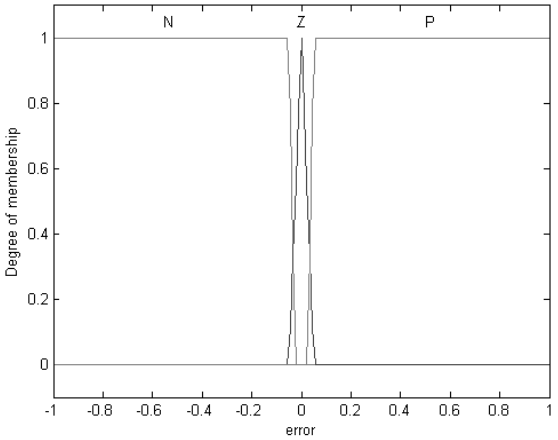


Fig. 7. Membership functions of the input variable error of the fuzzy system proposed to control the convergence of the ACO algorithm

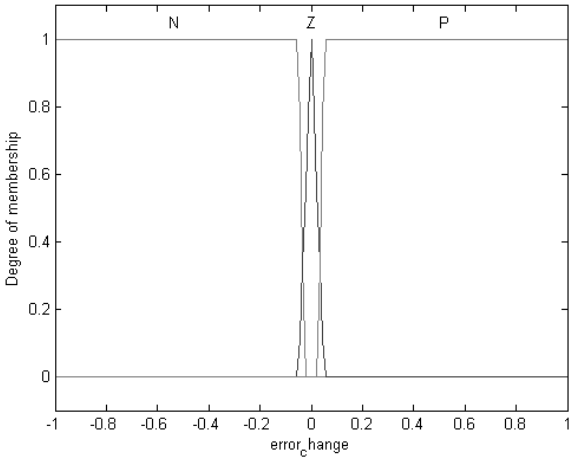


Fig. 8. Membership functions of the input variable error change of the fuzzy system proposed to control the convergence of the ACO algorithm

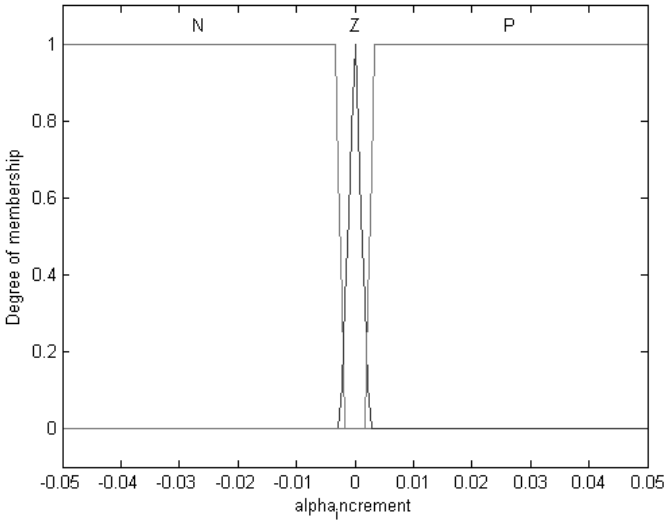


Fig. 9. Membership functions of the output variable alpha increment of the fuzzy system proposed to control the convergence of the ACO algorithm

5 Simulation

The controller was able to maintain diversity in a more appropriate level, avoiding the full convergence of the algorithm. The same number of experiments that in the above analysis were performed and obtained the following results:

Table 8. Performance obtained by the strategy proposed in the instances discussed above

TSP	Best	Average	Successful runs
Burma14	3323	3323	30/30
Ulysses22	7013	7013	30/30
Berlin52	7542	7543	26/30
Eil76	538	539	21/30
KroA100	21292	21344	0/30

It was found that the proposed method was able to improve the results of the strategies studied, obtaining lower averages (Figure 10) and reaching the global minimum on more occasions than the analyzed variants (Figure 11).

To verify the above in a more formal way a Z test for means of two samples was performed.

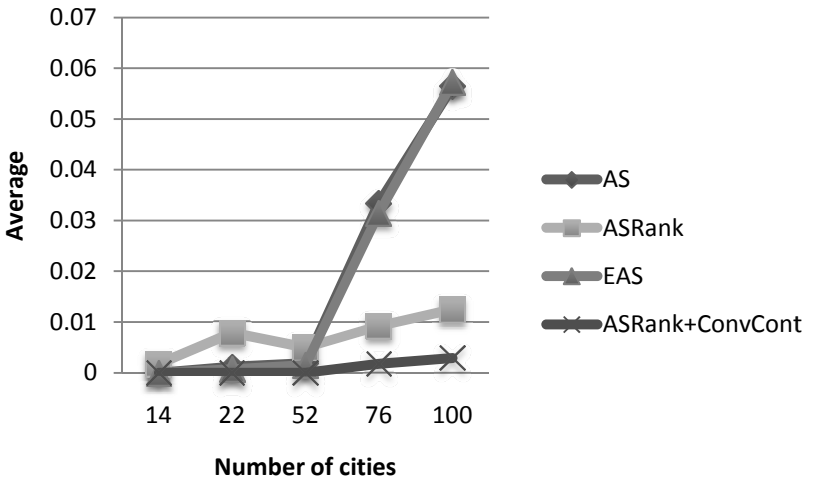


Fig. 10. Average of the results obtained by the proposal and each approach under review

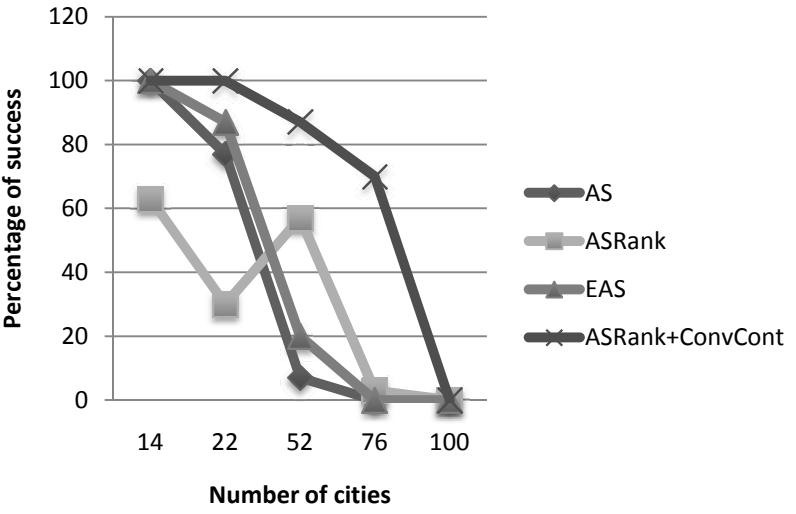


Fig. 11. Percentage of success in finding the global minimum of the proposal and each approach under review

The 3 ACO variants mentioned above were analyzed in addition to the approach developed in 5 instances of the TSP, 30 experiments were performed for each instance, 150 experiments were made in total of we extracted a 30 data random sample for each method.

It started by comparing the average of AS with ASRank+ConvCont, which is the name of the approach developed in this work, the null hypothesis states that the average of AS is less than or equal to ASRank+ConvCont.

With a significance level of 5% we found sufficient statistical evidence to claim that the average of AS is higher than the obtained for ASRank+ConvCont in the experiments (Figure 12).

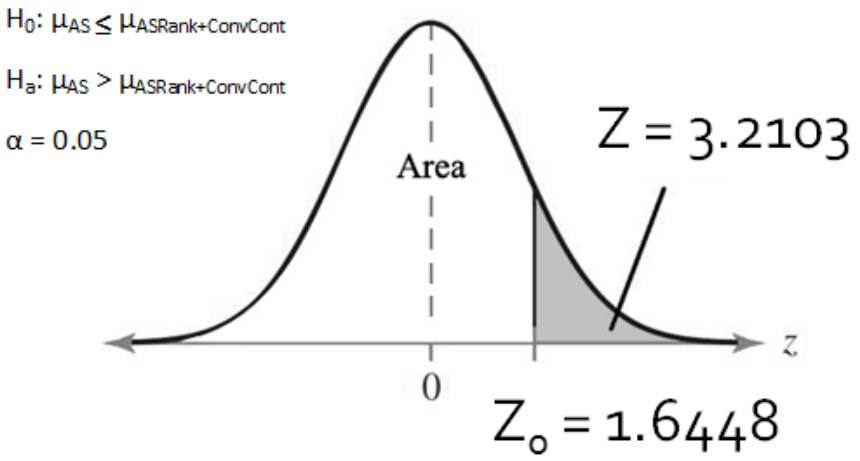


Fig. 12. Statistical hypothesis testing for AS vs. ASRank+ConvCont

Following the average of EAS with ASRank+ConvCont was compared, the null hypothesis claims that the average of EAS is less than or equal to ASRank+ConvCont mean.

Again with a significance level of 5% enough statistical evidence were found to show that the average of EAS is greater than ASRank+ConvCont mean since Z (test statistic) is in the rejection region (Figure 13).

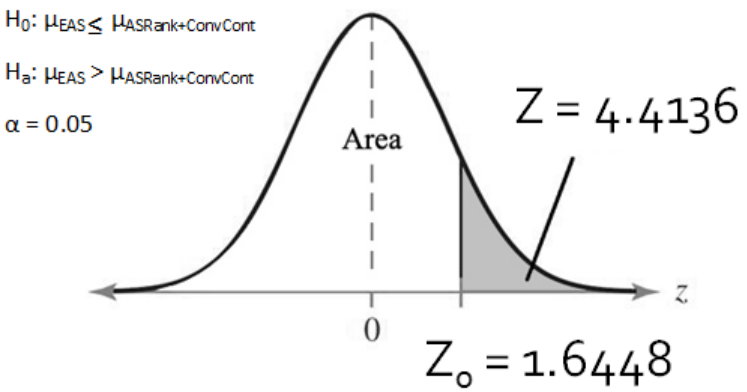


Fig. 13. Statistical hypothesis testing for EAS vs. ASRank+ConvCont

Finally the average of ASRank against ASRank+ConvCont was compared, the null hypothesis claims that the average of ASRank is less or equal to the ASRank+ConvCont mean.

Since Z is located in the rejection region with a significance level of 5% (Figure 14) there is enough statistical evidence to show that the average ASRank is also higher than ASRank+ConvCont mean.

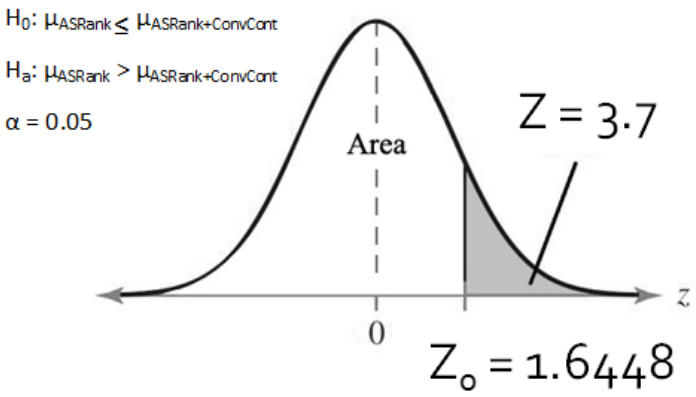


Fig. 14. Statistical hypothesis testing for ASRank vs. ASRank+ConvCont

In the three cases the average were lower, this means that our approach improved the performance of the ACO algorithm on the studied problems, as had been observed in the first analysis.

6 Conclusions

Maintaining diversity is important for good performance in the ACO algorithm. An adaptive control strategy of the parameter alpha for this purpose was used, which was embodied in a diversity fuzzy controller which allows to avoid or delay the total convergence and thereby control the exploration and exploitation capabilities of the algorithm.

The strategy was compared with 3 variants of the ACO algorithm on several instances of the TSP taken from TSPLIB. An improvement was observed by dynamically changing the parameter alpha value, as was seen in the statistical analysis performed, where our approach outperforms the classical strategies.

This strategy looks promising, so it is important to use the proposed variant in more complex optimization problems to observe its efficiency and consider modifying another parameter in conjunction with the above.

References

1. Dorigo, M., Stutzle, T.: Ant Colony Optimization. MIT Press, Cambridge (2004)
2. Meyer, B.: Convergence control in ACO. In: Genetic and Evolutionary Computation Conference (GECCO), Seattle, WA (2004)
3. Merkle, D., Middendorf, M.: Prospects for dynamic algorithm control: Lessons from the phase structure of ant scheduling algorithms. In: Heckendorn, R.B. (ed.) Proceedings of the 2000 Genetic and Evolutionary Computation Conference -Workshop Program. Workshop “The Next Ten Years of Scheduling Research". Morgan Kaufmann Publishers (2001)
4. Merkle, D., Middendorf, M., Schmeck, H.: Ant colony optimization for resource-constrained project scheduling. IEEE Transactions on Evolutionary Computation 6, 333–346 (2002)
5. Li, Y., Li, W.: Adaptive ant colony optimization algorithm based on information entropy, Foundation and application. Fundamenta Informaticae (2007)
6. Reinelt, G.: TSPLIB. University of Heidelberg, `http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/`
7. Yen, J., Langari, R.: Fuzzy Logic: Intelligence, Control and Information. Prentice Hall (1999)
8. Eiben, A.E., Michalewicz, Z., Schoenauer, M., Smith, J.E.: Parameter control in evolutionary algorithms. In: Lobo et al. (2007)
9. Stützle, T., López-Ibañez, M., Pellegrini, P., Maur, M., Montes de Oca, M.A., Birattari, M., Dorigo, M.: Parameter adaptation in ant colony optimization. In: Hamadi, Y., Monfroy, E., Saubion, F. (eds.) Autonomous Search. Springer, Berlin (2012)
10. Valdez, F., Melin, P., Castillo, O.: An Improved Evolutionary Method with Fuzzy Logic for Combining Particle Swarm Optimization and Genetic Algorithms. Applied Soft Computing 11, 2625–2632 (2011)
11. Chusanapiputt, S., Nualhong, D., Jantarang, S., Phoomvuthisarn, S.: Selective self-adaptive approach to ant system for solving unit commitment problem. In: Cattolico, M., et al. (eds.) GECCO 2006, pp. 1729–1736. ACM Press, New York (2006)
12. Hao, Z.-F., Huang, H., Qin, Y., Cai, R.: An ACO Algorithm with Adaptive Volatility Rate of Pheromone Trail. In: Shi, Y., van Albada, G.D., Dongarra, J., Sloot, P.M.A. (eds.) ICCS 2007, Part IV. LNCS, vol. 4490, pp. 1167–1170. Springer, Heidelberg (2007)

Using Immunogenetic Algorithms for Solving Combinatorial Optimization Problems

Francisco Javier Díaz-Delgadillo, Oscar Montiel-Ross, and Roberto Sepúlveda

Centro de Investigación y Desarrollo de Tecnología Digital del
Instituto Politécnico Nacional (CITEDI- IPN)
diaz@citedi.mx, {o.montiel,r.sepulveda}@ieee.org

Abstract. We present a novel approach for reducing the computational time of Combinatorial Optimization Problems (COPs). The approach is inspired by the use of Artificial Vaccines, a concept that is classified as being part of a set of algorithms called Artificial Immune Systems (AIS). Artificial Vaccines are able to reduce the computational time and quality of the solution of any existing COP solving algorithm by reducing the size of the problem set. To demonstrate the usefulness of this proposal we provide comparative results obtained with a Genetic Algorithm (GA) solving the Traveling Salesman Problem (TSP) for three large instances of 423, 737 and 1583 cities.

Keywords: Artificial Vaccines, AIS, COP, TSP.

1 Introduction

Obtaining the optimal solution of Combinatorial Optimization Problem has been known ever since the term was coined by mathematicians; it consists of exploring and evaluating every possible combination in order to find the solution.

However, simple this solution might sound, in reality, the problem becomes much more complex. When a COP involves more than just a few numbers of elements the complexity of the problem quickly grows out of hand than even the fastest computers will not be able to solve it by means of brute force. Hence why even until this day COPs are considered an open problem and finding better and faster algorithms that can provide the solution to them has received great emphasis by the scientific community and has proven to be a very important field of research [1].

The quest to develop better algorithms that outperform current algorithms have drawn inspiration from various sources, including physical and biological phenomenon. The presented algorithm is no exception, and in this case, we take inspiration from the natural functions of the human immune system, being its extrapolation into the mathematical and computational world the Artificial Immune Systems (AIS) [2]. The presented algorithm works quite different than most, as its meant to be a tool that can be applied to existing COP solving algorithms in order to provide better performance.

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 273–288.
springerlink.com

The COP to be studied in this work is the Traveling Salesman Problem (TSP) being a classic benchmarking example of COP. The problem is modeled after a traveling salesman that is presented with a dilemma, in which he wants to find the shortest route in order to visit a set of cities only once, maintaining the cost at a minimum. The problem also states that he wishes to start and end at his headquarters. Fig. 1 shows an example of the solution to a 423 TSP instance.

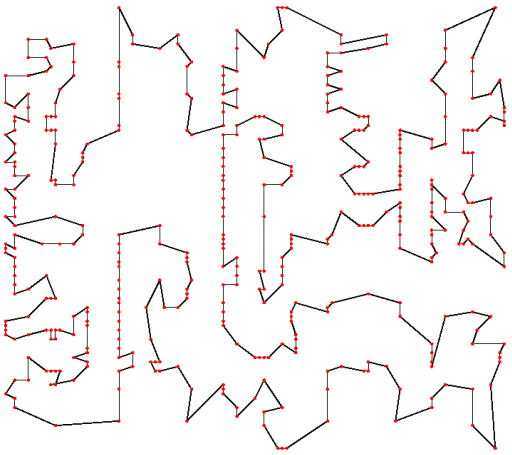


Fig. 1. Example of the optimal tour for the TSP instance pbn423 from TSPLIB

Although one can be inclined to think this problem is merely theoretical, even the simplest formulation of it has several real world applications, such as path planning, scheduling, logistics and manufacturing of Printed Circuit Boards, or even in Control Numeric Computer Machining.

The complexity of the TSP relies in the fact that as the number of cities grows, finding the solution by means of brute, or exhaustive force becomes a NP-Complete problem, since the number of possible combination, which needs to be generated and evaluated, becomes $(n\text{-}1)!$, this quickly gets out of hand.

Many different techniques have been developed in order to provide with solutions to large instances of TSP, and new methods are presented every year. In general, there are two ways for of solving the TSP: by the exact solution or by a metaheuristics algorithm.

Some of the most popular exact algorithms for solving the TSP are branch-and-bound and lineal optimization or programming [3][1], although these types of algorithms can be very effective at solving some instances of TSP, they usually fall short when the problem sets become very large, and they usually fall back to other type of techniques, in addition, to the exact approach.

Metaheuristics are currently the most popular way to solve very large instances of TSP; some of the main algorithms are Genetic Algorithms (GA) [4], Ant Colony Optimization (ACO) [5], and Neural Networks (NN) [6]. Also in the field of

metaheuristics there are several works inspired in AIS that have been used successfully to solve the TSP, they use Clonal Selection [7] or Cooperation Mechanisms of Th cells [8] or even hybrid immune-genetic attempts [9].

There have also being exact and metaheuristics hybrid approaches that have proven to be very effective. One of the most popular benchmarking and readily available algorithm is presented in a program named Concorde, which is considered by many as one of the best TSP solvers [10][11]. Concorde's algorithm has proven to be quite effective when compared to other methods (exact, metaheuristics or otherwise) and has been able to handle a very large number of cities providing excellent results [11].

In this work, we explore the concept of Artificial Vaccines to reduce the complexity of TSP instances in order to provide better solutions and lower computational time, we are improving our previous results presented in [12].

The main goal was to create a methodology or tool that can be used with existing TSP solving algorithms without compromising the algorithm itself. The proposed methodology is supported with several comparative experiments of TSP instances of 423, 737 and 1583 cities. A Genetic Algorithm was utilized as the test algorithm for our methodology.

The organization of the work is as follows: In section 2, the Human and Artificial Immune Systems are explained. In section 3, Artificial Vaccines are defined mathematically for COPs as well as the main algorithms for their generation. In Section 4, the experimental process is explained. In section 5, the results are presented. Finally, in section 6 the conclusions are given.

2 Human and Artificial Immune Systems

The function of the Human Immune System is to keep our body healthy. It manages to do this by constant monitoring of our body by means of search and identification of abnormalities or any foreign agents named pathogens.

The process of identification and resolving these issues starts with what is called an immune response. Our body is capable of generating two types of immune responses called innate and adaptive [13].

The innate immune response is the ability of our immune system to identify a threat and handle it even if it's never been encountered before. This process is very important because once it's started; the cells responsible for identifying the original threat undergo a series of changes that include cloning and hyperosmotic mutation with the goal that through these changes they will eventually be able to exterminate the threat. This process is one of the most important functions of our immune system, and it serves as the main concept behind the Clonal Selection Algorithm of AIS [2].

The ability to recall previously encountered pathogens is called the adaptive immune response. The way this is achieved is by keeping alive for a period of time the cells that have successfully eliminated a pathogen in a previous innate response. In this way, future attacks will be handled in a better way, since the immune system will have the appropriate tools to respond faster and more efficient.

In reality, the complexity of the human immune system is much greater than the previous description; however it provides us with a very interesting view into why the human immune system is a great paradigm for new and exciting ways to obtain data and information and for the creation of algorithms. These attributes are in part because the human immune system, as well as other sources of inspiration in nature, demonstrate very sophisticated capabilities for adaptation, search, identification and learning, which are valuable characteristics to extrapolate into solving engineering problems.

The AIS are the extrapolation of the works of human immune system into the fields of mathematics, computer science and engineering [14]. The concept behind them is to model the main properties of the immune system and use them as a foundation to solve real world problems.

The following characteristics from the human immune system stand out when looking to extrapolate them into real-world problems and applications [2]:

- Pattern recognition
- Singularity
- Diversity
- Autonomy
- Multilayer system
- Fault detection
- Adaptive capabilities
- Robustness
- Immune learning and memory
- Distributed system

Even though, we can identify these attributes of the immune system, the exact way that they are achieved is still a main part of research for immunologists. As our knowledge of the immune system expands so will the extrapolation into the AIS will grow, eventually finding their way to our daily scientific and engineering lives.

There are some types of problems where AIS are particularly good at solving because of their relationship between a natural function and a computational task. Some of the main applications of the AIS are [2]:

- Pattern recognition
- Fault detection and anomalies
- Data mining and classification
- Agent based systems
- Scheduling
- Machine learning
- Autonomous control and navigation
- Optimization and search
- Artificial life

AIS has served as the main source of inspiration for the proposed methodology. Taking into attention that the handling and solution of COPs can take into consideration Pattern Recognition and Optimization we ventured into the field of Artificial Vaccination. This field has not been explored by AIS even when it has been considered in the literature [2]. The main concepts we want to extrapolate from the way vaccination is as follows: given the immune system which has a very specific task of protecting ourselves, inject an external element or agent that allows reinforcing this function. In this sense, we extrapolate this into having an algorithm that already provides a function – solving the TSP – and by means of artificial vaccination allow this algorithm to have better performance.

3 Artificial Vaccine Definition and Implementation

The main issue with every COP solving algorithm is that as the number of elements grows, so does the problem's complexity and this usually leads to degradation in performance either by quality of the solutions themselves, by a substantial increase of the execution required to find an optimal solution or by simply becoming unsolvable in some cases.

This situation has become a very important challenge for every researcher in the field of algorithms for solving COP and a constant search for a methodology that alleviates this issue is exactly the focus of this work. The Artificial Vaccine is our proposed solution to this.

In the following sections, we will discuss the main concept of the Artificial Vaccine and how to generate them – by two different methods – and apply them to a problem set.

3.1 Artificial Vaccine

The *Artificial Vaccine (AV)* is the foundation of the proposed algorithm, we have defined it as a 7-tuple as is specified in (1)

$$AV = (VID, NL, NN, IE, FE, GC, L) \tag{1}$$

where,

- **Vaccine Identifier (VID):** Every AV requires a unique identifier in order to complete the Vaccination Process (described at a later section).
- **Node List (NL):** Each AV holds either the pointer to the data that represent each node or the original node index.
- **Number of nodes (NN):** Each AV must have a parameter that states how many elements it holds from the original problem set.
- **Initial Element (IE):** The Initial Element or node of the AV.
- **Final Element (FE):** The Final Element or node of the AV.

- **Geometrical Center (GC):** Because the Vaccination Process requires the subtraction of all of the implicated nodes of an AV from the problem set, a substitute for these nodes is reinserted as a placeholder. The Geometrical Center represents this point. For a 2D problem, set the coordinates of the GC, they are calculated using (2) and (3).

$$\text{Center in x: } X_c = \frac{\sum_{i=0}^{n} x_i}{n} \tag{2}$$

$$\text{Center in y: } y_c = \frac{\sum_{i=0}^{n} y_i}{n} \tag{3}$$

- **Length (L):** The length of the implicated nodes in the AV, calculated by Euclidian distance using (4),

$$L = \sum_{i=0}^{n-1} \sqrt{(x_i - x_{i+1})^2 + (y_i - y_{i+1})^2} \tag{4}$$

Fig. 2 shows the conceptual illustration of an Artificial Vaccine, and it allows us to visualize how it would be implemented in software. It consists primarily of an array of node indexes that represent a possible combination of values.

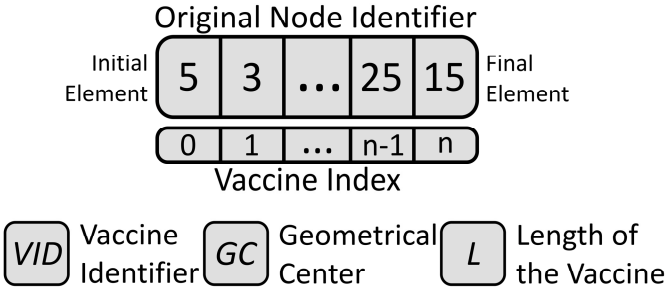


Fig. 2. Conceptual representation of an Artificial Vaccine

3.2 Vaccination Process

Vaccination Process is the methodical generation, application and removal of vaccines from a COP set.

It consists of three basic steps. The first step is the generation and application of the AVs to the problem set or Original Node List (ONL) in order to create the Reduced Node List (RNL). For this step, we propose two distinct algorithms called Vaccination by Random Selector (VRS) and Vaccination by Elitist Selector (VES), both of these algorithms are discussed in depth on the following sections.

The second step consists of the solution of the RNL COP with an already established algorithm. Once the algorithm presents a solution we are not done, as it will be in terms of the RNL and not the ONL.

This brings us to step three in which we take the optimized tour, and we exchange the AV for their corresponding original values in order to bring the solution to terms of the ONL. This whole process is illustrated in the flowchart shown in Fig. 3.

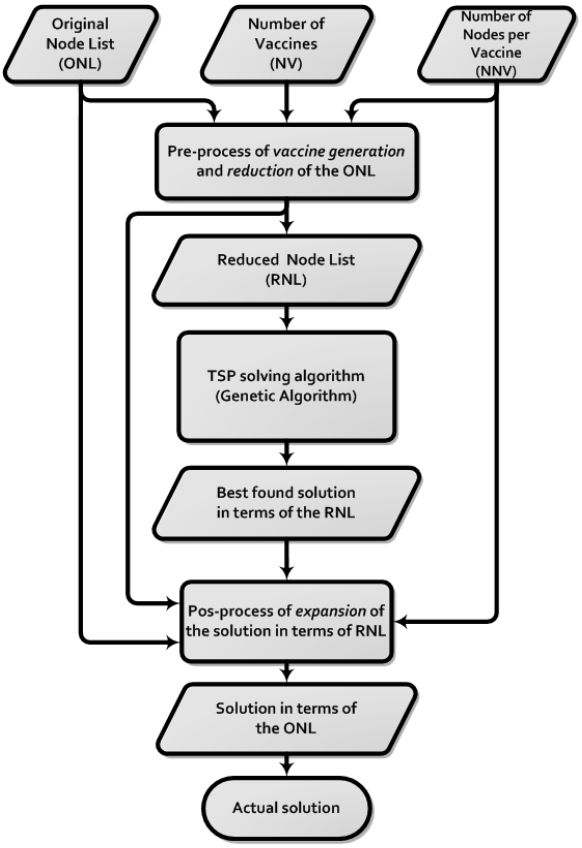


Fig. 3. Flowchart illustrating the Vaccination Process

3.3 Vaccination by Random Selector (VRS)

Our first method for generating AV is called Vaccination by Random Selector (VRS). This method is particularly useful when information about the problem set is scarce, nonexistent or simply when this information costs too much to generate.

We first randomly select the Initial Element of the AV from which the rest of the AV is constructed. Once the element is selected, the VRS algorithm requires calculating the distance from the chosen node to the rest of nodes with the aim of choosing the nearest one. This method is based on the Nearest Neighbor algorithm but does not calculate the entire problem set, only for small subsets.

In the TSP, we utilize Euclidean distance in order to determine the nearest neighbor. VRS is an iterative process for each individual AV that finishes when the AV reaches the amount of nodes required per AV; i.e., when the number NN indicated at (1) has been reached. This whole process is repeated as many times as needed to fill the Vaccine List (VL). Fig. 4 shows the algorithm for this process.

Procedure VRS. Inputs: Original Node List (ONL), Number of Vaccines (NV), Number of Nodes per Vaccine (NNV).
Output: Vaccine List (VL), Reduced Node List (RNL).

```
Procedure VaccineVRS(ONL, NV, NNV)
copy ONL to Available Node List (ANL)
copy ONL to Reduced Node List (RNL)
while (NV is not met) do
  randomly select a node which is named Random Node (RN) from ANL
  calculate the route of the nearest nodes from RN. The route length is
  equal to NNV
  generate a vaccine which spans from RN to each of the involved Nodes
  calculate the Geometrical Center (GC) for the Vaccine
  add Vaccine to Vaccine List (VL)
  remove the implicated nodes that form the vaccine from the RNL
  add a new Node with the vaccine's GC to the RNL
  remove from the ANL the randomly selected node RN
end while
return VL and RNL
```

Fig. 4. Generation of Artificial Vaccines using VRS

3.4 Vaccination by Elitist Selector (VES)

In contrast with VRS, Vaccination by Elitist Selector (VES) requires full information from the problem set, specifically the distance matrix in the case of the TSP in order to generate each AV.

First we must iterate through each one of the nodes in the problem set and calculate as many nearest neighbors as stated in NN for that particular node. After this is done, each one of these calculations is an AV, however, now we perform a

sorting operation based on the length of this subset of nodes, and we choose as many as we need to complete our Vaccine List (VL), effectively selecting the shortest connections in our problem set. Fig. 5 shows the algorithm for this process.

```
Procedure VES. Inputs: Original Node List (ONL), Number of Vaccines (NV),
               Number of Nodes per Vaccine (NNV).
               Output: Vaccine List (VL), Reduced Node List (RNL).
Procedure VaccineVES(ONL, NV, NNV)
copy ONL to Reduced Node List (RNL)
generate distance matrix for ONL
{by taking each node and calculating as many nearest neighbors are NNV
states}
sort the ONL by shortest distance to the nearest neighbor
for each Node in ONL do
  if NV is not met then
    select the corresponding nearest neighbors for the node until reach NNV
    generate the two possible routes (from left-right and from right-left)
    if (left-right and right-left route) are not in VL then
      generate a Vaccine with the given route
      calculate Geometrical Center (GC) of the Vaccine
      add Vaccine to VL
      remove the implicated Nodes that form the Vaccine from the RNL
      add a new Node with the vaccine's GC to the RNL
    else
      do nothing
    end if
```

Fig. 5. Generation of Artificial Vaccines using VES

3.5 Expansion of Vaccinated Route

As explained before, after the COP solving algorithm gives a solution we must perform an additional step in order to obtain the real value for this solution. We have defined this method as the expansion algorithm and basically what it takes the solution that is in terms of the RNL (vaccinated by either VRS or VES) and convert it in terms of the ONL (remove the vaccines).

Procedure Expand (EXP). Inputs: Original Node List (ONL), Reduced Node List (RNL), Vaccine List (VL), Optimal Route (OR). Output: Expanded Route (ER). Variable list: Initial Node (IN), Final Node (FN), Vaccine (V).

```
Procedure Expand(ONL, RNL, VL, OR)
select first Link L from OR
select first node from L as Selected Node (SN)
for each Link in the Optimum Route OR do
  if SN is not Vaccine then
    add SN to ER
  else
    find the V in VL equal to SN
    calculate distance d1 and d2 {Distance from the SN to the IN and FN
    of V}
    if d1 ≤ d2 then
      add to the ER the IN {index 0, IN=Initial Node}
      add rest of the Nodes from V starting from index 1 to n-1
    else
      add to ER the FN {index n, FN=Final Node}
      add rest of the Nodes from V starting from index n-1 to 0
    end if
  end if
  select SN based on the following Node defined by the Link
end for
output ER
{order of elements is the expanded route}
```

Fig. 6. Algorithm to expand the Reduced Node List to the Original Node List

When talking about the TSP, the previously removed cities are reinserted as sub-paths where the placeholder AV is located and the total tour length is recalculated taking this into consideration. Fig. 6 shows the algorithm for this process.

4 Experiments

For testing the proposed method, we have mentioned that we decided to use the TSP because there is extensive research with known results that have been stored in public repositories; this is an advantage since it facilitates comparison and duplication of experimental results.

We proposed test is aimed to demonstrate that by using the Vaccination method is possible to outperform the solutions provided by the Genetic Algorithm for large TSP instances.

For this test, we used three different TSP instances, obtained from the public library TSPLIB pbn423, rbu737, and rbv1583, with city count of 423, 737 and 1583 respectably. These problem sets are shown in Table 1, where the problem name, city count and the known optimal route are given. In accordance with the results, we have considered that this set is enough large for evaluating the Vaccination method.

Table 1. TSP Instances used in experimentation. All problems were obtained from TSPLIB

TSP name	City Count	Optimal Route Length
pbn423	423	1365
rbu737	737	3314
rbv1583	1583	5387

For each TSP problem of Table 1, three experiments that embrace a big diversity of comparative situations were conducted; they will help to conclude about the usefulness of the application of the Vaccination method when it is applied to a GA. Next, we describe each set of experiments.

Experiment 1. Obtaining benchmarks: This set of experiments consisted in obtaining the control data to be used as benchmarks in order to achieve comparisons. The behavior of the GA optimizing the TSP for a different amount of cities were marked using the symbol "□" in Figs. 7 to 9.

Experiment 2. Vaccination with Random Selector (VRS): In this set of experiments vaccination using VRS on the City List was used; i.e., we applied the Algorithm 1 shown in Fig. 4. The behavior of the VRS algorithm for a different amount of cities were marked using the symbol "◦" in Figs. 7 to 9.

Experiment 3. Vaccination using the Elitist Selector (VES): The algorithm 2 described in Fig. 5 was applied to obtain a reduced set of cities. Similarly to Experiment 2, after reducing the set of cities, the GA was used to obtain the shortest path, and then optimal route was using the Algorithm 2 was obtained. The behavior of the VES algorithm for a different amount of cities were marked using the symbol "⋄" in Figs. 7 to 9.

For the three sets of experiments, the results were recorded every 500,000 generations, we let the TSP solver algorithms run 10 million generations; the results are shown from Figs. 7 to 9. For the GA, the parameters are as follows: population size of 1000 chromosomes, mutation rate of 3%, random population generation with a 75% greedy selection rate and a PMX crossover operation. For Experiment 2 and 3, the vaccination parameters were as follows: Number of Vaccines is 40% of the Original Node List and Number of Nodes per Vaccine was 2.

5 Experimental Results and Analysis

The experimental results were organized into two sections: the first considers optimal values and the second execution times. All the experiments were achieved several times, and results are consistent and replicable.

Results of the experiments are shown in Table 2; the graphs of the collected data are shown from Figs 7 through 9. In these graphs the GA always outperforms the final tour value when the Vaccination method is applied. The best suboptimal values were obtained using the VES algorithm, although they converge soon for TSP instances with a small number of cities. Note that, with a noticeable small number of generations the Vaccination method is providing acceptable suboptimal values, this is more evident for large TSP instances, where the GA is far from reaching the optimal value.

Table 2. Experimental results for the three TSP instances and the three respective experiments

	GA	VRS		VES	
TSP	L	L	%	L	%
pbn423	1892.4	1738.0	8.16	1622.9	14.24
rbu737	7031.2	5002.2	28.86	4783.3	31.97
rbv1583	23781.8	15540.1	34.66	12491.9	47.47

Being more specific, for the case of the TSP with 423 cities (pbn423), from Table 2 the GA reached the value of 1892.4 which is 38.64% worse solution than the known global optimum of 1365 showed in Table 1. Similarly, for the same problem, the VRS and VES with values of 1738 and 1622.9 are local optimal values in a 27.33% and 18.89% far from the global optimum respectively. Significantly, the VRS and VES provided 8.16% and 14.24% better solutions respectively than the GA.

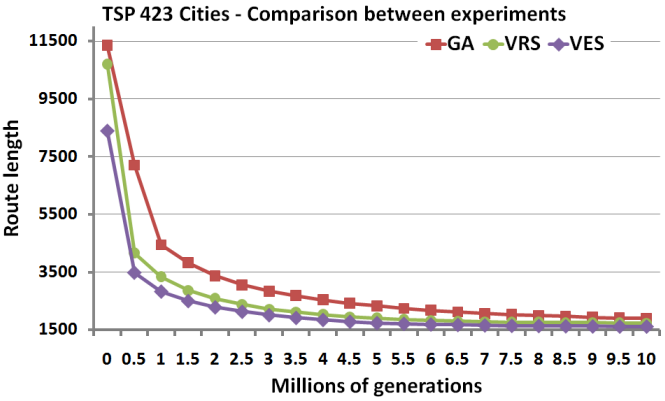


Fig. 7. The Vaccination method improved the performance of a GA in the TSP instance with 423 cities. VES gave the best values with fewer generations; VRS is computationally less expensive because it does not require to sort the original city list.

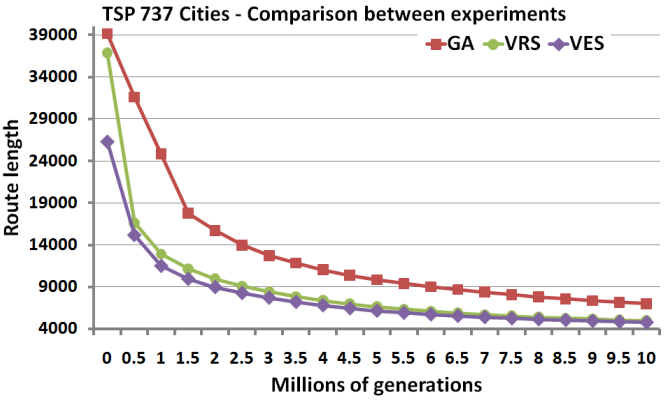


Fig. 8. The Vaccination method outperformed the GA every generation milestone that was recorded along the TSP instance with 737 cities

For the TSP with 737 cities (rbu737), from Table 2 the GA reached the value of 7031.2 which is 112.17% worse solution than the global optimum of 3314. The VRS and VES with values 5002.2 and 4783.3 are local optima values in a 50.94% and 44.34% far from the global optimum respectively. However, with the VRS and VES we obtained better routes than the GA alone in a 28.86% and 31.97% respectively.

For the TSP with 1583 cities (rbv1583), from Table 2 the GA alone reached the value of 23781.8 is 341.47% worse solution than the known global optimum of 5387. The VRS and VES with values 15540.1 and 12491.9 are local optima values in a 188.47% and 131.89% far from the global optimum respectably. Notably, the solutions provided by VRS and VES are 34.66% and 47.47% better than those provided by the GA alone respectively.

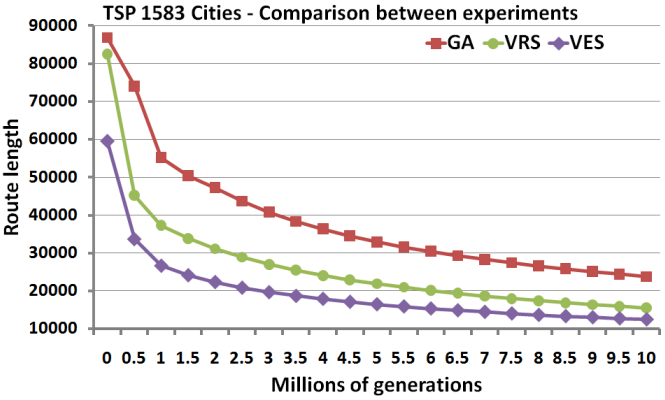


Fig. 9. The benefits of applying the Vaccination method become more evident as the TSP instance grows in size

Regarding execution time, there was no substantial difference between using VRS or VES, see Fig. 10; however, the use of AV speeds up notoriously the solving process in comparison with the GA alone, as it can be seen in Fig. 10.

The improvements in execution time are attributed to the effective reduction of nodes that the optimization algorithm has to work in order to provide a solution. The improvements are determined by (5) that produce a number of Reduced Node List (nRNL) that will depend on the parameters selection.

$$nRNL = nONL - (NV * NNV) + NV \tag{5}$$

Where *nONL* is the number of Original Node List, *NV* is Number of Vaccines and *NNV* is Number of Nodes per Vaccine. With this formula in Table 2 we present the effective reductions for each TSP instanced with 40% vaccination and 2 Number of Nodes per Vaccine. Table 3 shows the nRNL for each of the TSP instances.

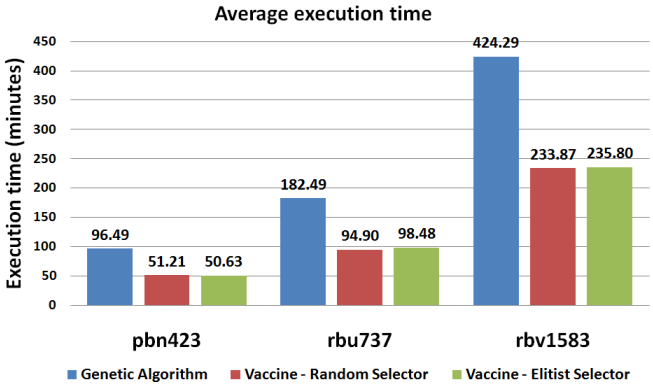


Fig. 10. Comparison of average execution time of the GA vs. VRS and VES for the TSP instances pbn423, rbu737 and rbv1583

Table 3. Number of elements after vaccination. NV = nRNL * 0.4 and NNV = 2

Problem	Formula	nRNL
pbn423	423-((423*.4)*2)+(423*.4)	254
rbu737	737-((737*.4)*2)+(737*.4)	442
rbv1583	1583-((1583*.4)*2)+(1583*.4)	950

6 Conclusions

A novel methodology that allows the reduction of computational complexity of COPs based on the concept of immunization by vaccination was presented. As a case of study, we present the widely known TSP as well as problem instances from the library TSPLIB pbn423, rbu737 and rbv1583.

The main issue with every COP solving algorithm is that as the number of elements grows, the solutions that can be provided are degraded by either creating worst solutions, taking a substantial amount of time to generate them or simply being unable to produce one.

The Artificial Vaccine is presented to alleviate this by providing a methodology where the problem set for a COP is reduced in size by the systematic generation, application and removal of the Artificial Vaccines.

The methodology consists of three steps. The first step is the generation and application of the Artificial Vaccines to the problem set. For this step, we proposed two distinct algorithms called Vaccination by Random Selector (VRS) and Vaccination by Elitist Selector (VES).

The second step consists of the solution of the TSP by any algorithm. Once this is done, the solution must be converted back into the original problem set values.

In step three, we take the previous solution and return it to terms of the original problem set in order to obtain the real final value for the solution.

In order to test our proposal, we utilized a Genetic Algorithm and conducted three experiments to three different problems instances of the TSP. These tree experiments were meant to provide benchmarking of the GA with and without our proposal of VRS and VES.

For the three problem instances, the application of Artificial Vaccines provided for better overall solutions and in the largest instance with 1583 nodes a better solution by 28.86% and 31.97% for the VRS and VES were obtained respectively.

Additional to the better solution values, the computational time for each one of the experiments was reduced substantially between 40% and 55%. This is due the fact that the reduction in nodes necessarily reduces the number of calculations that the TSP solving algorithm must perform.

A formula to obtain the number of Reduced Node List (RNL) was developed; it provides a tool that allows estimating the improvements before performing the actual reduction in the number of nodes; therefore, the reduction in execution time can be known before testing which can be tedious and time consuming. The formula takes into consideration vaccination parameters such as the Number of Vaccine (NV) and Number of Nodes per Vaccine (NNV) which provides flexibility.

Further study into the effects of modifying the parameters NV and NNV on the problem set and solutions is left as future work.

As a last note, when comparing VRS and VES, VRS requires less computational time to create the Artificial Vaccines while VES provides the better Artificial Vaccines that lead to better solutions. The difference in the execution time for the same problem set for both VRS and VES none existing as the number of reduced nodes for both is the same.

Acknowledgment. The authors thank Instituto Politécnico Nacional, Comision de Operación y Fomento de Actividades Académicas del Instituto Politécnico Nacional, and CONACYT for supporting our research activities.

References

1. Vazirani, V.: Approximation Algorithms. Springer (2004)
2. de Castro, L., Timmis, J.: Artificial Immune Systems: A New Computational Intelligence Approach, pp. 55–60. Springer, Great Britain (2002)
3. Ausiello, G.: Complexity and approximation: combinatorial optimization problems and their approximability properties. Springer (1999)
4. Larrañaga, P., Kuijpers, C.M.H., Murga, R.H., Inza, I., Dizdarevic, S.: Genetic algorithms for the travelling salesman problem: A review of representations and operators. Artificial Intelligence Review, 129–170 (1999)
5. Gambardella, L.M., Dorigo, M.: Ant colonies for the traveling salesman problem. IRIDIA (1997)
6. Budinich, M.: A self-organizing neural network for the travelling salesman problem that is competitive with simulated annealing. Neural Computation 8 (1996)
7. Zhao, M.Y., et al.: A novel clonal selection algorithm and its application. In: Apperceiving Computing and Intelligence Analysis, pp. 385–388 (2008)
8. Zhu, Y., Tang, Z., Dai, H., Gao, S.: Cooperation artificial immune system with application to traveling salesman problem. ICIC Express Letters, 143–148 (2008)
9. Jiao, L., Wang, L.: A novel genetic algorithm based on immunity. Systems, Man and Cybernetics. IEEE Transactions on Part A: Systems and Humans 8 (2000)
10. Gutin, G., Jakubowicz, H., Ronnen, S., Zverovitch, A.: Seismic vessel problem, pp. 13–20 (2005)
11. Mulder, S.A., Wunsch II, D.C.: Million city traveling salesman problem solution by divide and conquer clustering with adaptive resonance neural networks. Neural Networks 16, 827–832 (2003)
12. Diaz-Delgadillo, F.J., Montiel-Ross, O., Sepúlveda, R., Melin, P.: Automatic Handling of Expert Knwoledge Using Artificial Immune System Applied to Combinatorioal Optimization Problems. Engineering Letters 20(1), 81–87 (2012)
13. Dasgupta, D., Niño, L.F.: Immunological Computation: Theory and Applications, pp. 8–9. CRC Press Taylor & Francis Group (2009)
14. Chen, J., Mahfouf, M.: Handbook of Research on Artificial Immune Systems and Natural Computing: Applying Complex Adaptive Technologies Chapter II Artificial Immune Systems as a BioInspired Optimization Technique and Its Engineering Applications, pp. 22–45. IGI Global (2009)

Comparison of Metaheuristic Algorithms with a Methodology of Design for the Evaluation of Hard Constraints over the Course Timetabling Problem

A. Soria-Alcaraz Jorge, Carpio Martín, Puga Héctor,and A. Sotelo-figueroa Marco

Division de Estudios de Posgrado e Investigacion, Instituto Tecnologico de Leon, Leon Guanajuato, Mexico
{Soajorgea,masotelof}@gmail.com, jmcarpio61@hotmail.com, pugahector@yahoo.com

Abstract. The Course Timetabling problem is one of the most difficult and common problems inside an university. The main objective of this problem is to obtain a timetabling with the minimum student conflicts between assigned activities. A Methodology of design is a strategy applied before the execution of an algorithm for timetabling problem. This strategy has recently emerged, and aims to improve the obtained results as well as provide a context-independent layer to different versions of the timetabling problem. This methodology offers to an interested researcher the advantage of solving different set instances with a single algorithm which is a new paradigm in the timetabling problem state of art. In this paper the proposed methodology is tested with several metaheuristic algorithms over some well-know set instances such as Patat 2002 and 2007. The main objective in this work is to find which metaheuristic algorithm shows a better performance in terms of quality, used together with the Design Methodology. The algorithms chosen are from the area of evolutionary computation, Cellular algorithms and Swarm Intelligence. Finally our experiments use some non-parametric statistical test like Kruskal-Wallis test and wilcoxon signed rank test.

1 Introduction

The timetabling problem is one of the most difficult, common and diverse problems inside an university. This problem tries to assign several activities into a *Timelsots* making a *Timetabling*. The main objective of this problem is to obtain a timetabling with the minimum conflicts between assigned activities [16]. This paper focus on the *Course timetabling problem* (CTTP). This problem assigns subjects to individual students with a minimum conflicts, usually time-conflicts, between the assigned events.

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 289–302.
springerlink.com

Like most timetabling problems, the *Course timetabling* is NP-Complete [6] [21]. The reason is the combinatorial explosion of possible events assigned into timeslots, as well as the constraints that each university uses in the course timetabling creation. Due to this complexity and the fact that, even now, many of these course timetabling constructions are making by hand, it is necessary to automate the construction process improving the performance of the solutions reached by the human expert [11].

If we consider that every university usually needs to implement a new course timetabling algorithm in order to achieve a good solution (basically due its internal policies) then it exists an important obstacle; for example an algorithm that solves the problem in a university may not be able to provide at least a feasible solution for another university.

In this context a new methodology of solution has appeared, the methodology of design [16] [17] [18]. This design layer offers a new start up method where it translates several university policies from the original course timetabling input into a set of generic structures for its treatment by means of metaheuristic algorithms. This design provides a context-independent layer allowing metaheuristic algorithms to work and solve several course timetabling problems without using any explicit university constraints.

The metaheuristics algorithms has been characterized for offer good results in a reasonable time. There exist a huge variety of metaheuristics applicable to a wide range of problems, but the no-free lunch theorem [22] indicate us that there no exist such a metaheuristic capable to give a good solution for every possible problem. The selection of the best possible metaheuristic algorithm has a great importance over the final performance for the generic proposed approach. This paper shows a comparative study between several different metaheuristics and its performance over a design approach for the course timetabling problem.

The paper is organized as follows. Section 2 presents the design methodology for the course timetabling, The metaheuristics chosen for comparison and its justification. Section 3 contains the experimental setup, results, analysis and discussion. Finally Section 4 include some conclusions and future work.

2 Comparative Approach

2.1 Problem Definition

A clear and concise definition of the CTTP is given by Lewis Rhydian [11]: A set of events(courses or subjects) $E = e_1, e_2, \ldots, e_n$ is the basic element of a CTTP. Also there are a set of periods of time $T = t_1, t_2, \ldots, t_s$, a set of places (classrooms) $P = p_1, p_2, \ldots, p_m$, and a set of agents (students registered in the courses) $A = a_1, a_2, \ldots, a_o$. Each member $e \in E$ is a unique event that requires the assignment of a period of time $t \in T$, a place $p \in P$ and a set of students

$S \subseteq A$, so that an assignment is a quadruple (e, t, p, S). A timetabling solution is a complete set of n assignment, one for each event, which satisfies the set of hard constraints defined usually by each university of college. This problem is documented to be at least a NP-complete problem [6] [21].

2.2 Methodology of Design for the Course Timetabling Problem

In the literature it can be seen that there exists a problem with the diversity of course timetabling instances due different university policies. This situation directly impacts in the reproducibility and comparison between course timetabling algorithms[15]. The state of art indicates some strategies to avoid this problem. For example, a more formal problem formulation [15] as well as the construction of benchmark instances [11]. These schemes are useful for a deeper understanding of the university timetabling complexity, but the portability and the reproducibility of a timetabling solver in another educational institution is still in discussion[16]. In this sense, we use a context-independent layer for the course timetabling resolution process. This new layer integrates timetabling constraints into three basic structures *MMA matrix, LPH list and LPA list*.

MMA matrix contains the number of students in conflict between subjects i.e. the number of conflicts if two subjects are assigned in the same timeslots. An example of this matrix can be seen in the Figure 1.

	ACM0403	SCM0414	SCB0421	SCE0418	ACH0408	ACM0401	ACM0404
ACM0403	4	1	1				
SCM0414	1	10	3	3			6
SCB0421	1	3	3	2			2
SCE0418		3	2	3			3
ACH0408							
ACM0401						4	1

Fig. 1. MMA matrix

LPH List: This structure have in its rows the subjects offered. In its columns have the offered timeslots, So this list gives information about the allowed timeslots per subject. one example of this list can be seen on table 1.

Table 1. LPH List

	Day 1	Day 2
e_1	$<t_3>$	$<t_2>$
e_2	$<t_2>$	$<t_2 or\ t_1>$

LPA list: This list shows in its rows events and the classrooms available to be assigned to it without conflict. An example of this table can be seen on table 2.

Table 2. LPA List

Events	Classrooms
e_1	$<P_1,P_2>$
e_2	$<Lab_1,Lab_3>$
e_3	$<P_3,Lab_2>$
e_4	$<Lab_4>$
⋮	⋮
e_n	$<P_2>$

These structures are constructed with the natural/original inputs of the CTTP problem.This process ensures *by design* the non-existence of violations in the selection of any values showned in LPH and LPA. Sothe main problem is how to deal with students conflicts. A proposal is to work with these conflicts by means of the next minimization function:

$$Min(FA) = \sum_{i=1}^{k} FA_{V_i} \tag{1}$$

$$FA_{V_j} = \sum_{s=1}^{(M_{V_j})-1} \sum_{l=1}^{M_{V_i}-s} \left(A_{j,s} \wedge A_{j,s+l}\right) \tag{2}$$

Where: FA = Student conflicts of current timetabling. V_i = Student conflicts from "Vector" i of the current Timetabling. $A_{j,s} \wedge A_{j,s+l}$ = students that simultaneously demand subjects s and $s+1$ inside the "Vector" j. A = student that demands subject s in a timetabling j.

The concept of vector is the most important element in the design methodology. This vector is a binary representation of an event.[17][16] It can be constructed as seen on table 3 where each v_i is a vector that represents event e_i.

The vectors can be easily added and subtracted to construct sets. the symbols used for these sets of vectors are *V1, V2 … V_i*. One characteristic is that the number of vectors sets is related with the number of timeslots offered by the current timetabling. The main idea about vectors is to have a space where the events can be move without assigned them to a fixed timeslot. This independent layer of context generalizes in the solution process of the CTTP problem.

The main objective is to construct a fixed number of vectors sets (usually the cardinality of timeslots set) in order to obtain zero conflict on MMA, LPH and LPA. It is precisely for the vector sets construction that a metaheuristic algorithm is needed, but if any other CTTP problem can be expressed by means of the Methodology of design then the same algorithm can be applied without any modification.

Table 3. Vector Construction

	e_1	e_2	…	e_{n-1}	e_n
V_1	1	0	…	0	0
V_2	0	1	…	0	0
⋮	⋮	⋮	⋯	⋮	⋮
V_{i-1}	0	0	…	1	0
V_i	0	0	…	0	1

2.3 Metaheuristics

As it can be seen from section 2.2 the generic approach needs to construct a particular vector set where the number of student conflicts between the assigned subjects be the minimum or idealy zero. The construction of this set can be seen as a combinatorial problem, despite of the wide variety of metaheuristics that can be applied to this kind of problem. The chosed metaheuristics for its comparison have been tested over similar problems with a good reported performance for its respective instances. It is important to say that this is the first time that these algorithms will be tested with an generic approach like the methodology of desing, and by the No-free lunch theorem [22] the fact that these algorithms have showed a good performance with its particular approaches does not guarantee that it can be expected a similar behaviour for all metaheuristics in the proposed approach.

Classic GA (sGA). The genetic algorithm is a search heuristic that mimics the process of natural evolution. Developed by Holland 1971. This algorithm is usually the first option among similar heuristic tools that a researcher has to solve a combinatorial problem like the CTT problem. In the Course timetabling state of art this algorithm has reported a good results for early works like [25] [8] and [5]. The legacy of this algorithm and its ease of implementation allows us to selected it as the first metaheuristic tool for our generic approach. This GA uses direct representation previously reported by Soria et al [17] where each gene represents an event to be assigned into a timeslot or vector. This GA uses single point cross-over. Probabilistic Roulette for re-selection operator and simple elitism.

Frequency GA (fGA). Genetic Algorithm with non direct and dynamical representation. GA Developed by Soria et. al. 2011, This algorithm selects a subset of events based on its domain size to solve it, afterwards selects other subset of events until every event is assigned to an timeslot. This GA share the same operators with the classic GA, but changes the representation. This algorithms uses in each gene an indirect value of the possible timeslot to be chosen, in the practice this allows the conformation of irregular matrix that minimizes the computational operations needed to achieve the number of generations given by the user.

Eclectic GA (eGA). Genetic Algorithm with Vasconcelos selection and auto-adaptation in its parameters. This Genetic algorithm was developed by Morales and Quezada 1998 [13], and has shown a good performance over high-constrained problems [12]. The auto-tunning allows this algorithm to escape from local optima by itself. This GA shares the same operators with the fGA and Classic GA.

Cellular GA (cGA). Genetic Algorithm with high parallelism developed by Alba et.al [2] [1]. This GA limits each individual to a specific neighbourhood (NEWS neighbourhood in this paper), also each individual is placed in a toroidal grid. This kind of algorithm admits sub-populations that work at the same time in different regions of the search space, but gathers information with a migration operator. Several adaptations of common GA operators are made in order to use them in this cGA, For example the selection operator only selects neighbours for each individual/cell. The Elitism operator is changed as well, making only possible to choose the best individual for each subpopulation, i.e in each subpopulation the best cell cannot be modify ,but this cell can modify (cross-over) others.

Differential Evolution (DE). The Differential Evolution (DE) is a evolutionary strategy designed for problems of continuous nature. This algorithm has been reported [19] [20] [14] as a good algorithm capable to work with high constrained problems in a small time. Developed by Storn and Price, It is a vector-based algorithm and can be considered as a further development to GA. This stochastic search algorithm with self-organizing tendency do not uses information of derivatives (as a GA). Unlike GA, DE carries out operations over each component of the vector (in our case each variable coded in an individual). This kind of operators can be expected more efficient when the optimal solution is near to the current point coded in an individual [24]. Despite of the discrete nature of the CTT problem, the DE can work over a representation like our chromosome where each *component* is an event whit a well-defined pool of timeslot choices.

PSO. Particle Swam Optimization is based on the swarm behaviour was developed by Kennedy and Eberhart(1995) [9]. Since then the PSO has been applied to almost every area in optimization, computational intelligence and design/scheduling applications [24]. This algorithm searches the space of an objective function by adjusting the trajectories coded inside each particle (in our case the time values of each variable/event) in a quasi-stochastic manner. Each particle is attracted toward the position of the current global best g^* and its best location x^*_i in history, while at the same time it has a tendency to move randomly. This algorithm can be seen adapted to CTT problem with the same considerations seen on the DE algorithm.

Great Deluge Algorithm (GDA). The Great Deluge Algorithm was developed by Dueck, 1993 [7] based on simulated annealing. This algorithm uses only one parameter *time execution*. It has been observed [3] that a enough big execution

time impacts positively in the final solution granted by this algorithm. The GDA strategy works with an quasi-stochastic search, looking for the best possible solution in the fitness landscape but its constrained to: Do not chose a worse solution that current and Do not take a solution with a fitness smaller that the remain time execution units. This strategy ensures for example that if GDA has 100 remain execution units(time) no solution with a fitness bigger that 100 will be accepted. This algorithm have been tested over CTT problems with good results.[23]

2.4 Test Instances

The methodology of design allows to solve several different set instances as long as these instances can be expressed in term of the generic structures (MMA, LPH and LPA), That is the principal advantage of this generic approach. Two well know and referenced set instances are taken to make the comparison experiment over our generic approach, these set instances PATAT 2002 and PATAT 2007 were made for the first and second International Timetabling Competition respectively.

There are 20 test instances for patat 2002 and 24 for patat 2007, the main characteristics are share between these sets like the main data as well as some constraints. The last two hard constraints marked by (*) are only utilized in ITC 2007.

Patat 2002 and Patat 2007

These instances consist in:

- A set of n events that are to be schedule into 45 timeslots.
- A set of r rooms, each which has a specific seating capacity.
- A set *features* that are satisfied by rooms and required by events.
- A set of s students who attend various different combination of events.

The hard constraints are:

- No student should be required to attend more that one event at the same time
- Each case the room should be big enough for all the attending students
- Only one event is put into each room in any timeslot.
- Events should only be assigned to timeslots that are pre-defined as available *
- Where specified, events should be scheduled to occur in the correct order. *

The Soft constraints are:

- Students should not attend an event in the last timeslot of a day.
- Students should not have to attend three or more events in successive timeslots.
- Student should not be required to attend only one event in particular day.

3 Experiment Design

The comparison between the selected metaheuristics was made with PATAT 2002 an 2007. Basically once each metaheuristic is adapted to the proposed generic approach, that adaptation is used to solve both test instances. It should be noted that exists previous works [10] [4] where some metaheuristics were tested for patat 2002 or patat 2007, however this is the first time where a single generic algorithm is capable to solve both instance sets with no special adaptation for each case. For the present comparison each metaheuristic execute 100 independent experiments in order to assume statistical normality as well as 1000 functions points per independent run. The parameters used for each metaheuristic can be seen on table 4. These parameters were taken from the literature and empirical evidence obtained in this paper.

Table 4. Parameters for each metaheuristic

Algorithm	Parameter	Value
sGA	Elitism	0.3
	Cross-over	0.85
	Mutation	0.15
	Population	256
fGA	Elitism	0.3
	Cross-over	0.95
	Mutation	0.1
	Selection	Harmonic
	Population	256
eGA	Elitism	0.3
	Cross-over	Auto adaptable
	Mutation	Auto adaptable
	Population	256
cGA	Elitism	1 per sub-population
	Cross-over	0.937
	Mutation	0.1
	sub-populations	16
	individuals	256
	Neighbourhood	NEWS
DE	f	0.9
	Cr	0.5
	Population	256
PSO	gBest	0.8
	lBest	0.4
	Inertia	0.95
	Population	256
GDA	Time	Until it finish

3.1 Results

The results achieved for each metaheuristic can be seen on table 5 and 6. The presented results only shows the first 10 instances per set, if the reader is interested in detailed results please contact the authors.

Table 5. Results for PATAT 2002 instances

Instance	Result	sGA	fGA	eGA	cGA	DE	PSO	GDA
2002-1	Mean Fitness	324	305	308	188	303	288	279
	Stddeviation	18.8	19.8	20.4	14.9	12.2	19.6	15.5
2002-2	Mean Fitness	305.5	215	260	179	184	252	204
	Stddeviation	18.2	15.5	19.9	14	13.5	17.6	18.4
2002-3	Mean Fitness	330.4	278.58	248.04	202.7	295.9	239.6	252.5
	Stddeviation	15.3	16.8	15.6	13	12.5	16.8	18.4
2002-4	Mean Fitness	485.5	451.2	397.2	304	438.2	349.8	390.5
	Stddeviation	23.9	24	22.8	22.9	19.5	26.5	28.4
2002-5	Mean Fitness	480.18	360	423.6	293.3	348.6	440	310.2
	Stddeviation	26.57	25.5	28.5	21.4	23.3	22.4	21.9
2002-6	Mean Fitness	481.8	374.8	416.5	294.8	280.5	439.1	402.7
	Stddeviation	26.56	25.4	26.85	21.06	25.4	30	27.87
2002-7	Mean Fitness	503.4	371.2	282.3	287.9	268.5	400.1	339.9
	Stddeviation	30.55	29.3	27.6	24.9	12.45	18.66	24.4
2002-8	Mean Fitness	371.94	228.6	316.5	210.2	180.6	350.2	237.6
	Stddeviation	25.14	28.4	22.65	18.65	17.8	28.1	24.6
2002-9	Mean Fitness	346.8	254.1	288.6	207.9	274.5	266.5	281.5
	Stddeviation	20.54	19.5	17.65	16.56	12.45	16.4	20.2
2002-10	Mean Fitness	335.15	277.9	202.5	201.9	213.7	271.1	218.5
	Stddeviation	21.24	19.8	16.5	14.2	10.4	17.4	15.4

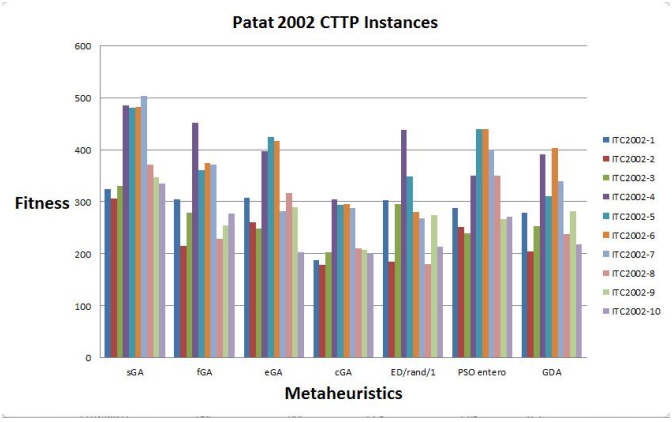


Fig. 2. Performance over Patat 2002 Instances

Table 6. Results for Patat 2007 instaces

Instance	Result	sGA	fGA	eGA	cGA	DE	PSO	GDA
2007-1	Mean Fitness	1362.50	1315.60	1160.20	975.30	1064.15	1281.45	1279.50
	Stddeviation	55.00	52.73	47.10	45.96	54.76	54.59	47.93
2007-2	Mean Fitness	1388.50	1123.23	1189.30	999.00	1068.32	1315.20	986.23
	Stddeviation	56.87	48.38	47.84	42.78	55.28	44.00	50.08
2007-3	Mean Fitness	556.80	445.30	378.60	286.75	336.80	366.50	359.20
	Stddeviation	57.80	57.58	38.89	27.00	26.45	39.43	27.84
2007-4	Mean Fitness	619.20	488.30	334.50	342.80	426.30	329.50	567.20
	Stddeviation	44.00	36.18	42.99	30.55	41.72	39.23	39.82
2007-5	Mean Fitness	805.30	697.50	689.23	564.4	520.51	576.30	650.04
	Stddeviation	35.70	34.39	35.52	32.28	34.61	33.78	33.54
2007-6	Mean Fitness	794.81	623.50	666.56	552.18	550.36	526.30	713.20
	Stddeviation	36.72	33.91	35.54	33.15	34.77	36.63	35.03
2007-7	Mean Fitness	334.90	284.50	200.69	187.8	272.27	196.52	230.13
	Stddeviation	28.96	17.56	17.15	15.5	27.48	20.13	21.14
2007-8	Mean Fitness	375.00	284.50	280.42	196.30	320.40	316.50	220.50
	Stddeviation	36.14	18.86	21.23	18.72	29.50	27.28	21.30
2007-9	Mean Fitness	1403.10	1206.50	1163.20	979.23	1488.50	1154.30	994.50
	Stddeviation	69.77	67.04	58.40	52.67	57.95	67.15	62.59
2007-10	Mean Fitness	1400.30	1305.60	1068.20	1011.60	1159.50	1235.40	1056.20
	Stddeviation	53.58	51.12	50.59	47.38	56.50	47.95	50.61

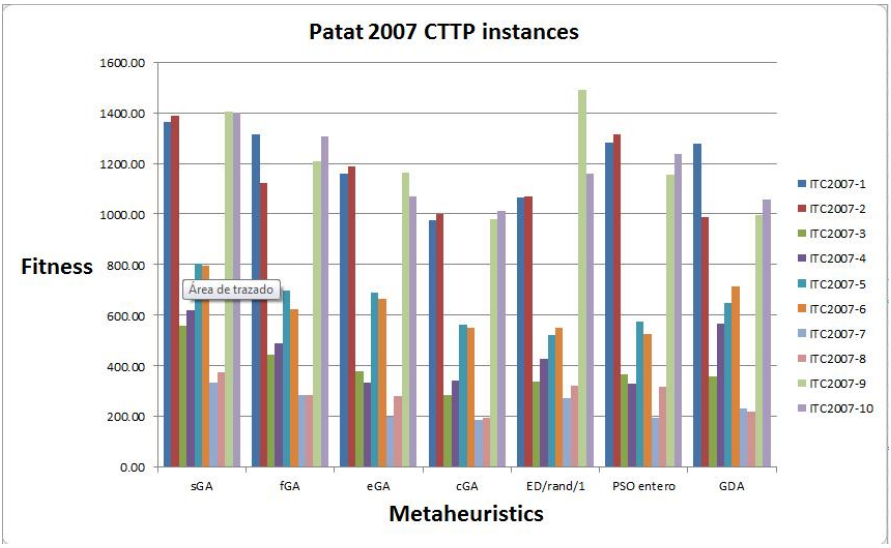


Fig. 3. Performance over Patat 2007 instances

3.2 Discussion

Once obtained the results shown in table 5 and 6 the non-parametric Kruskal-Wallis test is applied. This test is a method for testing whether samples originate from the same distribution. It is used for comparing more than two samples that are independent, or not related. The factual null hypothesis is that the populations from which the samples originate have the same median (in our case this means that each metahuristic have the same median performance). When the Kruskal-Wallis test leads to significant results, then at least one of the samples is different from the other samples. The test does not identify where the differences occur or how many differences actually occur.

The Kruskal-Wallis test rejects the null hypothesis and accept the alternative hypothesis: at least one of the distributions have different mean. This means that at least one of the algorithms have a different performance with the generic approach over the test instances. it can be seen some evidence that reinforces this result on figures 2 and 3, the figures shows different performance between the proposed algorithms.

Once it is known that the algorithms present different performance, it is needed to identify which metaheuristic present the best performance (Minimum conflicts), in order to do that Wilcoxon signed rank test is applied to every possible pair or metaheuristics. The Wilcoxon signed-rank test is a non-parametric statistical hypothesis test used when comparing two related samples, matched samples, or repeated measurements on a single sample to assess whether their population mean ranks differ. In this the wilcoxon signed rank test is applied in order to

evaluate which metahuristic has the best performance in terms of minimun conflicts.

By means of Wilcoxon signed rank test the cGA algorithm has shown a better performance in term of quality solution (conflicts) over our of test instances. The second best algorithm was the ED/Rand/1 also, this algorithm was also the fastest in execution. The second fastest algorithm was PSO however, this algorithm has not shown a good performance in terms of quality over the test instances. On the figures 2 and 3 it can be seen that the cGA algorithm has the best performance, this evidence support the results with Wilconxon Signed Rank test.

4 Conclusions and Future Work

This paper has shown a comparison between several different meta heuristics over an generic approach for the course timetabling problem. This generic approach has been use to solve both well known an referenced international test instances PATAT 2002 and PATAT 2007. The Design methodology was capable to solve both set of instances with a single algorithm.

This paper has gathered evidence about the good performance of the cGA algorithm as an metaheuristic tool to solve the CTT problem by means of a generic approach. The cGA algorithm uses an overlapped neighbourhood as well as a fixed toroidal structure, these concepts allows the cGA algorithm to diversify the genetic material in its individuals and preserve the best traits and characteristics of the best solutions founded. The cGA algorithm also utilizes a parallel scheme that accelerates the time needed to achive a solution. this algorithm uses a sub-population approach in order to search in different areas of the fitness landscape at the same time. This parallelism and sub-population techniques have shown a positive impact in the solution of CTT problem over an generic approach like the Design methodology.

For future work is proposed to analyse the performance of the cGA algorithm over a different set of instances like UNITIME.org with the same generic approach, also to make more test over different neighbourhood schemes for the cGA. The integration of the migration concept could be benefit for the cGA since this operator can be implemented in a parallel scheme, more test over this idea are suggested.

Acknowledgment. The authors thanks Consejo Nacional de Ciencia y Tecnologia (CONACYT) for the obtained support for this research.

References

1. Alba, E., Dorronsoro, B.: Introduction to Cellular Genetic Algorithms. Cellular Genetic Algorithms, 3–20 (2008)
2. Alba, E., Dorronsoro, B.: The State of the Art in Cellular Evolitionary Algorithms. Cellular Genetic Algorithms 1, 21–34 (2008)

3. Burke, E., Bykov, Y., Newall, J., Petrovic, S.: A time-predefined local search approach to exam timetabling problems. Computer Science TEchnical Report No. NOTTCS-TR-2001-6, 1 (2001)
4. Cambazard, H., Hebrard, E., O'Sullivan, B., Papadopoulos, A.: Submission to ICT: Track 2. In: International Timetabling Compertition 2007 (2008)
5. Colorni, A., Dorigo, M., Maniezzo, V.: Genetic algorithms and highly constrained problems: The time-table case. In: Schwefel, H.-P., Männer, R. (eds.) PPSN 1990. LNCS, vol. 496, pp. 55–59. Springer, Heidelberg (1991)
6. Cooper Tim, B., Kingston, J.K.: The Compexity of Timetable Construction Problems. PhD thesis, The University of Sydney (1995)
7. Dueck, G.: New Optimization Heuristics: The Great Deluge Algorithm and the Record-to-Record Travel. Journal of Computational Physics 104, 86–92 (1993)
8. Erben, W.: A Grouping Genetic Algorithm for Graph Colouring and Exam Timetabling. In: Burke, E., Erben, W. (eds.) PATAT 2000. LNCS, vol. 2079, pp. 132–158. Springer, Heidelberg (2001)
9. Kennedy, J., Eberhart, R.: Particle Swarm Optimization. In: Proccedings of IEEE International Conference on Neural Networks, vol. 1, pp. 1942–1948 (1995)
10. Kostuch. P. Timetabling Competition-SA-based Heuristic. Metaheuristics Network (2003)
11. Rhydian, L.: Metaheuristics for University Course Timetabling. PhD thesis. University of Notthingham (August 2006)
12. Angel, K.M.: A solution to the prisioner's dilemma using an eclectic genetic algorithm. In: IPN, vol. 1 (2000)
13. Angel, K.M., Quezada, C.V.: A universal Eclectic Genetic Algorithm for constrained optimization. ITAM, 1 (1998)
14. Price, K., Storn, R., Lampinen: Differential Evolution: A pratical approach to global optimization. Springer (2005)
15. Andrea, S., Di Gaspero, L.: Measurability and Reproducibility in University Timetabling Research: Discussion and Proposals. In: Burke, E.K., Rudová, H. (eds.) PATAT 2007. LNCS, vol. 3867, pp. 40–49. Springer, Heidelberg (2007)
16. Soria-Alcaraz Jorge, A.: Diseño de horarios con respecto al alumno mediante técnicas de cómputo evolutivo. Master's thesis, Instituto Tecnologico de León (2010)
17. Soria-Alcaraz Jorge, A., Carpio, M., Puga, H.: Diseño de Horarios mediante algoritmos géneticos. Décima Primera Reunión de Otoño de Potencia. In: Electrónica y Computación del IEEE, XI ROPEC, Morelia, vol. 1, pp. 24–35 (2009)
18. Soria-Alcaraz Jorge, A., Terashima-Marin, H., Carpio, M.: Academic Timetabling Design using Hyper-heuristics. In: Advances in Soft Computing, ITT, vol. 1, pp. 158–164. Springer (2010)
19. Storn, R.: On the usage of differential evolution for function optimization. In: Biennial Conference of the North America Fuzzy Information Processing Society (NAFIPS), pp. 519–523 (1996)
20. Storn, R., Price, K.: Differential evolution - a simple and efficient heuristic for gobal optimization over continuous spaces. Journal of Global Optimization 11, 341–359 (1997)
21. Willemen Robertus, J.: School Timetable Constructrion: Algorithms and complexity. PhD thesis. Institutefor Programming research and Algorithms (2002)
22. Wolpert, H., Macready, G.: No free lunch Theorems for Search. Technical report The Santa Fe Institute, 1 (1996)

23. Yang, Y., Petrovic, S.: A Novel Similarity Measure for Heuristic Selection in Examination Timetabling. In: Burke, E.K., Trick, M.A. (eds.) PATAT 2004. LNCS, vol. 3616, pp. 247–269. Springer, Heidelberg (2005)
24. Yang, X.-S.: Nature-Inspired Metaheuristics Algorithms, 2nd edn. Luniver Press (2010)
25. Yu, E., Sung, K.S.: A Genetic Algorithm for a University Wekly Courses Timetabling Problem. International Transactions in Operational Research 9, 703–717 (2002)

Path Planning Using Clonal Selection Algorithm

Christian Huizar, Oscar Montiel-Ross, Roberto Sepúlveda,
and Francisco Javier Díaz Delgadillo

Instituto Politécnico Nacional - CITEDI. Av. del Parque 1310, Tijuana, B., C., México
{chuizar,diaz}@citedi.mx, {o.montiel,r.sepulveda}@ieee.org

Abstract. We used the clonal selection algorithm (CSA) to generate optimal or nearly optimal solutions for solving combinatorial optimization problems, the algorithm was applied to generate quality G-Code sequences for a CNC Mill machine tool problem. We validated the software using the Traveler Salesman Problem (TSP) and demonstrated that the algorithm can optimize the travel path by reducing manufacturing time and costs. Different optimization experiments using the CSA of manufacturing codes of application CAD/CAM software are illustrated.

1 Introduction

The responsible system of the defense of a vertebrate being from multiple external microorganisms and molecules is the immune system. This natural system inspired scientists to create a computational model, calling Artificial Immune Systems (AIS) to those systems based on the theoretical immune principles [13-15]. The studies of AIS became strong at late 90s, [17] formulated a background for the AIS and their applications on real world problems. The works [2][18] have had a great impact on the knowledge of the AIS to the scientific community. In [20], a review of different applications is shown and how diverse they are, addressing suggestions to the development of AIS area.

De Castro and Timmis present basic information of models including clonal selection, immune network, positive selection and negative selection [1]. According to the model of AIS, the resulting algorithms are developed to tackle a variety of problems. Much AIS literature has based their work on the clonal selection theory, [22] highlights the impact of the Clonal Selection Algorithm (CSA) over the years and gives up-to-date applications.

Comparing the CSA with the Genetic Algorithms (GA), features such as affinity proportional reproduction and hypermutation are unique to the CSA [19]. Differing from the GA, the CSA has been compared in tasks [8] such as path planning showing good performance.

The manufacturing industry uses a machine tool CNC to perform machining processes faster than the average human can do. Drilling a bore can be a time consuming if the person in charge does not analyze a short route when programming the machine tool CNC. Modern software CAD/CAM is used to

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 303–312.
springerlink.com

generate the tool path by describing it with G-Code, which is a sequence of commands for automating operations of CNC machines, with the path, tools, and anything needed to produce the figure drawn on CAD software.

Some works [9] [10] [16], have used the AIS for path planning giving the authors a good step to apply this concept to path planning for the problem of optimizing the trajectory of the tool traveling for the points of a drilling operation. We developed an implementation of the CSA to generate the optimal sequence of G-code obtained from 2D CAD archives for drilling process, the aim is to reduce time and costs of performing the drilling operation on a Mill/Router CNC. The challenge is to reduce the manufacturing time which is achieve applying the CSA to optimize the path.

The organization of this work is as follows: Section 2 provides the necessary concepts of optimization, AIS and the CSA. In section 3, we present experiments to illustrate the CSA application, and results. Finally, in section 4 are the conclusions and future work.

2 Methods

The aim of this section is to provide a brief description of combinatorial optimization, as well as to present the basis of AIS emphasizing on the CSA.

2.1 Optimization

Finding the best element from a set of available options, can be written mathematically [11] as:

$$\begin{aligned} &\text{Minimize } f(x) \\ &\text{Subject to } x \in \Omega \end{aligned} \tag{1}$$

Where x is a vector satisfying f to result the best value, in this case the minimum from all possible values. The points that are the center of bores of a drilling operation can be seen as discrete values of a function. An example of this kind of function is the Travelling Salesman Problem (TSP), which is a combinatorial optimization problem. For a list of points (cities) and their distances, the task is to search for a minimal route, starting at some point travelling through every other only once and returning to the starting point.

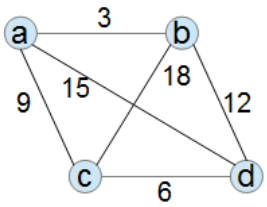


Fig. 1. Graph Representation of TSP

The optimization of paths has been done by other methods in the area of computational intelligence like Ant Colony Optimization [3], GAs, Bacteria Colony, and Particle Swarm Optimization [7]; comparisons of GAs and AIS are in [8].

The representation of the TSP using a graph is shown at Fig. 1 where all the nodes are the cities to visit and the connections are the weights of traveling between the connected nodes. The computational complexity of the TSP is O($n!$), where n is the total number of nodes on the problem.

2.2 Artificial Immune System

The system that is in charge of defending against unknown elements is called the immune system. The vertebrates have a dynamic changing immune system allowing them to recognize many antigens (nonself elements) and eliminate most of them even when the body has never been in contact with it previously. The system has been explained by theories of immunology [4] [6], one of them is about the proliferation of antibody molecules when an infectious element enters the body. Creating clones and mutating them to search good antibody-antigen recognition. The AIS are metaphors based on the immune system of vertebrates; they are adaptive systems, inspired by theoretical immunology and observed immune functions, principles and models, which are applied to problem solving [1]. They are needed to encode the actual problem to those principles that reign the interaction of all the immune molecules and cells, including antibodies and antigens. To have a graphical, and mathematical demonstration of those interactions there was proposed Shape-Spaces [12].

All the features about the immune molecules, their interactions with other molecules, their length, are a set called generalized shape of the molecule. Considering some length L for all parameters of a molecule describing the generalized shape of the molecule; one point over the L-dimensional space, the shape-space, can specify the generalized shape of a molecule showing the antigen-antibody (Ag-Ab) mutual interaction. A molecule m, be an antigen or antibody, can be represented by a string of length L as: $m=<m_1,m_2,...,m_L>$, where $m \in \Re^L$. The string can be of different kinds of attributes depending on the domain of the problem to solve.

The type of attributes considered in this work make a Hamming shape-space, each string of a finite alphabet will be the ordered way to visit the points. It is possible to have an interaction from different length antibody and antigens since the shape-space is for describing all those Ag-Ab interacting processes. Once the shape-space is coded, as with the attributes used for the representation of the domain problem, all interactions need to be measured, this measurement is called the affinity measure. Affinity measure is used to map the interaction between two strings, the molecules, into a non-negative number. Depending on the kind of attributes an affinity can be evaluated, some real-valued shape-spaces can use a distance measure like Euclidean and Manhattan. Euclidean distance as the affinity D of a pair Ag-Ab can be computed it by the equation (1).

$$D = \sqrt{\sum_{i=1}^{L} (Ab_i - Ag_i)^2} \tag{1}$$

$$D = \sum_{i=1}^{L} | Ab_i - Ag_i | \tag{2}$$

On the case of Manhattan distance, the affinity can be calculated using (2). It is not very common to use it, but as an alternative for real-valued shape-spaces. As an alternative, there are Hamming shape-spaces, which use the Hamming distance between two strings. Represented by the differences of between every element:

$$D = \sum_{i=1}^{L} \delta_i \qquad \text{where} \qquad \delta = \begin{cases} 1 & Ab_i \neq Ag_i \\ 0 & Otherwise \end{cases} \tag{3}$$

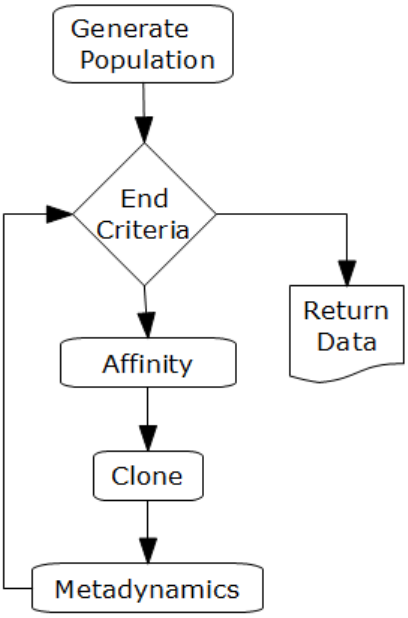


Fig. 2. Clonal Selection Algorithm

2.3 Clonal Selection Algorithm

The CSA is one computational model based on the Clonal Selection principle [6]. Fig. 2 shows the CSA proposed in [2]. The evolutionary procedure of the CSA is explained in [1].

3 Experiments

The commercial CAM (Computer Aided Manufacturing) software support DXF files generated by other CAD software applications. The implementation can read the DXF files to extract the points and generate the G-code as an output allowing to save as NGC files. Some basic figures to test the performance of the algorithm were used. In the experiments, we considered closed paths in order to illustrate the similitude to the TSP; however, in real-life applications it is not necessary to return to the original point.

3.1 Experiment 1

In Fig. 3, a 2D 10-points pattern arranged in a five-point mirror at each side of the y-axis with the points marked as circles is shown. It was obtained from a DXF file from Solidworks.

Fig. 3. 10-point mirror pattern

Fig. 4(a) shows the 10 points as they were captured by the designer; note, that they do not follow any optimal order, on Fig. 4(b) the optimal route is shown. Fig. 5 shows the algorithm evolution through iterations where the optimal route is found after six iterations. The total distance found on the original route is 47.03, after the optimization, we found a final distance of 32.92. The optimization percentage is 30%.

3.2 Experiment 2

In Fig. 6, a pentagon circumscribed, is shown. It has a total of 21 points to be drilled. Fig. 7(a) illustrate the original route and in Fig. 7(b) the optimized route is shown. Note that a suboptimal route is obtained in 39 iterations. The convergence plot is shown in Fig. 8. The original closed path total distance is 78.56; after the optimization, the final distance is 37.11. The optimization percentage is a 52.76%.

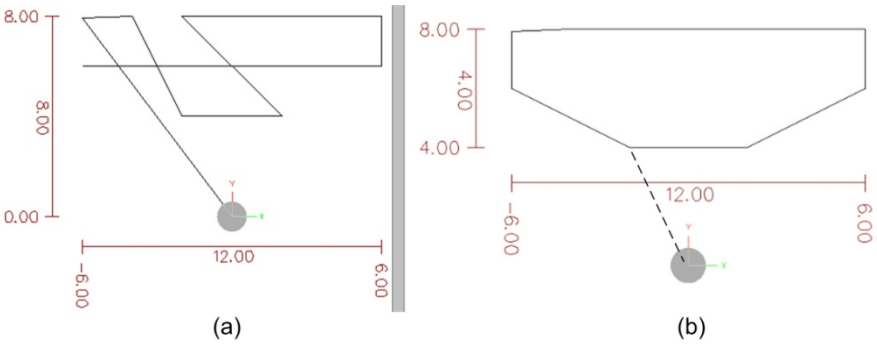


Fig. 4. 10-Point – (a) original route, (b) optimal route

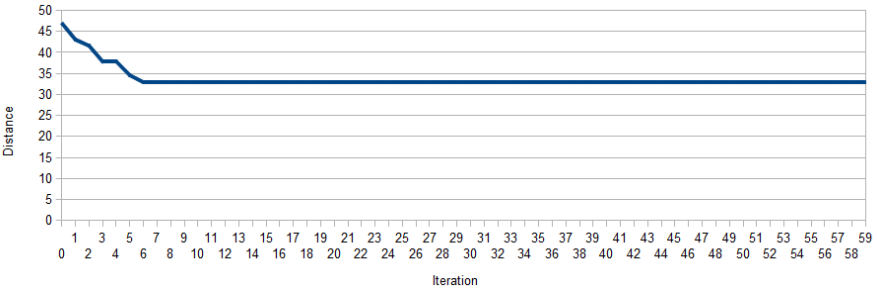


Fig. 5. 10-Point pattern convergence graph

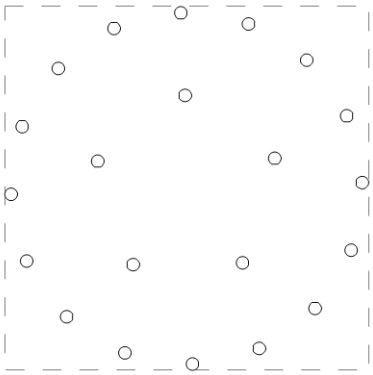


Fig. 6. Circumscribed pentagon pattern. DXF file from Solidworks

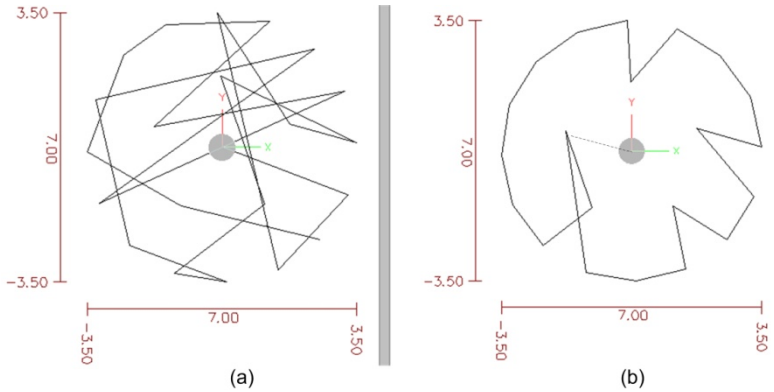


Fig. 7. Circumscribed pentagon – (a) original route, (b) suboptimal route.

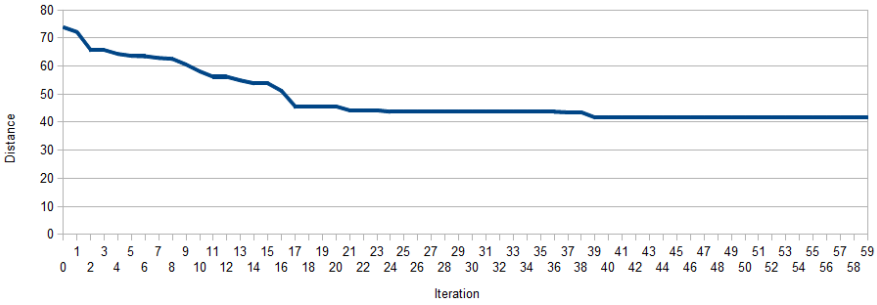


Fig. 8. Circumscribed pentagon convergence graph

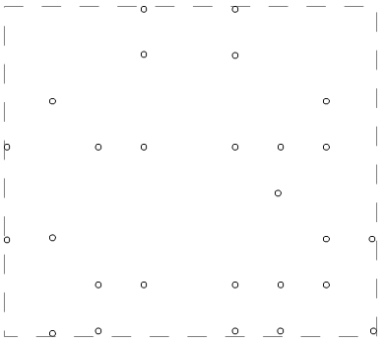


Fig. 9. 27-points test pattern. DXF file from Solidworks

3.3 Experiment 3

In Fig. 9, a 27-point pattern, is shown. Fig. 10(a) shows the original route from the CAD file; whereas, Fig. 10(b) shows the suboptimal route obtained after 43 iterations; it can be seen in Fig. 11. The original close path distance is 194.78; after the optimization, the optimal distance found is 112.09. The optimization percentage is 42.45%.

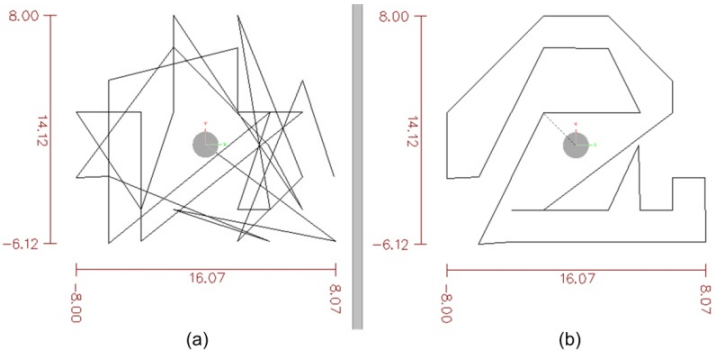


Fig. 10. 27-Point – (a) original route, (b) optimal route

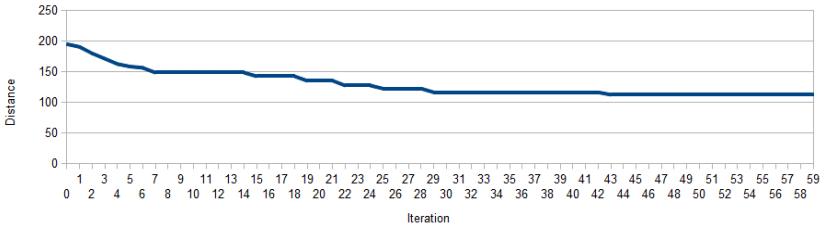


Fig. 11. 27-Point – Convergence graph

4 Conclusions

The CSA has been applied successfully in many applications. In this work, this basic implementation of the generic CSA was applied with the idea of generating optimal solutions in short times.

In this preliminary work, we used a basic generic implementation of the CSA. We achieve many runs of the three experiments obtaining optimal routes most of the times. Indisputably, they can improve the performance of many of the available application CAD software since the obtained routes are better, hence the manufacturing times are improved. For this basic implementation, we reduced manufacturing time in a range of 30% to 53%.

There still much work to do using AIS to solve the same problem since we achieved only a basic implementation of this computational paradigm, a suggestion is to use vaccines in order to reduce the computational cost of large problems [23]. Another interesting work to do is to compare the results obtained with this proposal, against previous results obtained with the Ant Colony Optimization Algorithm (ACO) [4][24].

For demonstrative proposes the authors added an extra distance, connecting the final point and the initial one. This practice is not necessary for field work, since the tool will have to drill again at the point where there is a bore. Common machining practice will require to extract at a safe point, and return to the home of the tool holder.

Acknowledgment. The authors would like to thank the Instituto Politécnico Nacional (IPN), Comisión de Operación y Fomento de Actividades Académicas (COFAA), and the Mexican Consejo Nacional de Ciencia y Tecnología (CONACYT) for supporting our research activities.

References

1. de Castro, L.N., Timmis, J.: Artificial Immune Systems: A New Computational Intelligence Approach. Springer (2002)
2. de Castro, L.N., Von Zuben, F.J.: The Clonal Selection Algorithm with Engineering Applications. In: Workshop Proceedings of GECCO, Workshop on Artificial Immune Systems and Their Applications, Las Vegas, USA, pp. 36–37 (July 2000)
3. Medina-Rodriguez, N., Montiel-Ross, O., Sepulveda, R., Castillo, O.: Tool Path Optimization for Computer Numerical Control Machines based on Parallel ACO. Engineering Letters 20, 101–108 (2012)
4. Janeway, C., Travers, P., Walport, M., Capra, J.: Immunobiology: The Immune System in Health and Disease, Garland (1999)
5. Farmer, J.D., Packard, N.H., Perelson, A.S.: The Immune System, Adaptation, and Machine Learning. In: International Symposium on Physical Design (1986)
6. Burnet, F.M.: The Clonal Selection Theory of Acquired Immunity. Cambridge Press (1959)
7. Sierakowski, C.A., dos S. Coelho, L.: Study of Two Swarm Intelligence Techniques for Path Planning of Mobile Robots. In: 16th IFAC World Congress, Prague, July 4-8 (2005)
8. Ojha, U., Chow, M.-Y.: An Analysis of Artificial Immune System and Genetic Algorithm in Urban Path Planning. In: IECON 2010 - 36th Annual Conference on IEEE Industrial Electronics Society, pp. 1064–1069 (2010)
9. Li, X., Cao, J., Chang, C.: Path Planning of Soccer Robot Based on Artificial Immune. In: 2010 International Conference on Computational Aspects of Social Networks Mechanics, pp. 543–546 (2010)
10. Mrwczynska, B.: Optimal routes scheduling for municipal waste disposal garbage trucks using evolutionary algorithm and artificial immune system. Transport Problems: an International Scientific Journal 6(4), 5–12 (2011)
11. Chong, E.K.P., Zak, S.H.: An Introduction to Optimization, 2nd edn. John Wiley and Sons (2001)

12. Perelson, A.S., Oster, G.F.: Theoretical Studies of Clonal Selection: Minimal Antibody Repertoire Size and Reliability of Self-Nonself Discrimination. Journal of Theoretical Biology 81, 645–670 (1979)
13. Hofmeyr, S.A., Forrest, S.: Architecture for an Artificial Immune System. Evolutionary Computation 7, 45–68 (2000)
14. Tarakanov, A., Dasgupta, D.: A Formal Model of An Artificial Immune System. BioSystems 55, 151–158 (2000)
15. Zak, M.: Physical Model of Immune Inspired Computing. Information Sciences 129, 61–79 (2000)
16. Li, X., Kwong, K.-C., Li, J., Jiang, L., Zeng, D.: Mobile Robot Path Planning Based on Artificial Immune Algorithm. In: 2009 IEEE Workshop on Advanced Robotics and its Social Impacts (2009)
17. Dasgupta, D.: Artificial Immune Systems and Their Applications. Springer (1998)
18. De Castro, L.N., Von Zuben: Artificial Immune Systems: Part I – Basic Theory and Applications (1999)
19. de Castro, L.N., Von Zuben, F.J.: Artificial Immune Systems: Part II – A Survey of Applications. Technical Report – RT DCA 02/00, FEEC/UNICAMP, Brazil (2000)
20. Zheng, J., Chen, Y., Zhang, W.: A Survey of artificial immune applications. Springer Science+Business Media B.V (2010)
21. Timmis, J.: Artificial immune systems-today and tomorrow. Natural Computing 6, 1–18 (2007)
22. Ulutas, B.H., Kulturel-Konak, S.: A review of clonal selection algorithm and its applications. Springer (2011)
23. Diaz-Delgadillo, F.J., Montiel-Ross, O., Sepúlveda, R., Melin, P.: Automatic Handling of Expert Knwoledge Using Artificial Immune System Applied to Combinatorioal Optimization Problems. Engineering Letters 20(1), 81–87 (2012)
24. Montiel, O., Medina, N., Sepúlveda, R., Melin, P.: Methodology to Optimize Manufacturing Time for a CNC Using a High Performance Implementation of ACO. International Journal of Advanced Robotic Systems: Robot Control (accepted paper, 2012)

Bio-inspired Optimization Methods on Graphic Processing Unit for Minimization of Complex Mathematical Functions

Fevrier Valdez, Patricia Melin, and Oscar Castillo

Tijuana Institute of Technology, Tijuana, Mexico
fevrier@tectijuana.mx

Abstract. Although GPUs have been traditionally used only for computer graphics, a recent technique called GPGPU (General-purpose computing on graphics processing units) allows the GPUs to perform numerical computations usually handled by CPU. The advantage of using GPUs for general purpose computation is the performance speed up that can be achieved due to the parallel architecture of these devices. This paper describes the use of Bio-Inspired Optimization Methods as Particle Swarm Optimization and Genetic Algorithms on GPUs to demonstrate the performance that can be achieved using this technology with regard to use CPU primarily.

Keywords: PSO, GA, GPU, CPU, Optimization.

1 Introduction

We describe in this paper the optimization of a set of mathematical functions using bio-inspired algorithms. In this paper, we are using Genetic Algorithms (GAs) and Particle Swarm Optimization (PSO), Simulated annealing (SA), and Pattern Search (PS) to optimize the functions. The main idea is compare this meta heuristic methods using the Central Processing Unit (CPU) and the Graphic Processing Unit (GPU). Nowadays several approaches have been made to optimize mathematical functions, for example in [1] [2] [3][13][15][16]. However our approach differs of these approaches because we are doing a comparison between the advantage of run the methods in a CPU and a GPU with the aim of achieve results quickly.

In section 1 we describe the introduction to a proposed method. In section 2, the bio-inspired optimization methods are described. In section 3, a brief history of GPUs is presented, in section 4 the experimental results are shown, in section 5 the conclusions are presented and finally in section 7, the references are presented.

2 Bio-inspired Optimization Methods

To compare the performance between run the methods in a CPU or GPU is necessary evaluate the methods used in this paper. Therefore, in this section we describe

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 313–322.
springerlink.com

a brief description about of the Bio-inspired optimization methods used in this research. The methods used are described as follow:

2.1 Genetic Algorithms

Holland, from the University of Michigan initiated his work on genetic algorithms at the beginning of the 1960s. His first achievement was the publication of *Adaptation in Natural and Artificial System* [7] in 1975.

He had two goals in mind: to improve the understanding of natural adaptation process, and to design artificial systems having properties similar to natural systems [8].

The basic idea is as follows: the genetic pool of a given population potentially contains the solution, or a better solution, to a given adaptive problem. This solution is not "active" because the genetic combination on which it relies is split between several subjects. Only the association of different genomes can lead to the solution.

Holland's method is especially effective because it not only considers the role of mutation, but it also uses genetic recombination, (crossover) [9]. The essence of the GA in both theoretical and practical domains has been well demonstrated [1][4]. The concept of applying a GA to solve engineering problems is feasible and sound. However, despite the distinct advantages of a GA for solving complicated, constrained and multiobjective functions where other techniques may have failed, the full power of the GA in application is yet to be exploited [12] [14].

2.2 Particle Swarm Optimization

Particle swarm optimization (PSO) is a population based stochastic optimization technique developed by Eberhart and Kennedy in 1995, inspired by the social behavior of bird flocking or fish schooling [3].

PSO shares many similarities with evolutionary computation techniques such as Genetic Algorithms (GA) [6]. The system is initialized with a population of random solutions and searches for optima by updating generations. However, unlike the GA, the PSO has no evolution operators such as crossover and mutation. In PSO, the potential solutions, called particles, fly through the problem space by following the current optimum particles [10].

Each particle keeps track of its coordinates in the problem space, which are associated with the best solution (fitness) it has achieved so far (The fitness value is also stored). This value is called *pbest*. Another "best" value that is tracked by the particle swarm optimizer is the best value, obtained so far by any particle in the neighbors of the particle. This location is called *lbest*. When a particle takes all the population as its topological neighbors, the best value is a global best and is called *gbest*.

The particle swarm optimization concept consists of, at each time step, changing the velocity of (accelerating) each particle toward its p*best* and *lbest* locations (local version of PSO). Acceleration is weighted by a random term, with separate random numbers being generated for acceleration toward *pbest* and *lbest* locations.

In the past several years, PSO has been successfully applied in many research and application areas. It is demonstrated that PSO gets better results in a faster, cheaper way compared with other methods [11].

2.3 Simulated Annealing

SA is a generic probabilistic metaheuristic for the global optimization problem of applied mathematics, namely locating a good approximation to the global optimum of a given function in a large search space. It is often used when the search space is discrete (e.g., all tours that visit a given set of cities). For certain problems, simulated annealing may be more effective than exhaustive enumeration provided that the goal is merely to find an acceptably good solution in a fixed amount of time, rather than the best possible solution.

The name and inspiration come from annealing in metallurgy, a technique involving heating and controlled cooling of a material to increase the size of its crystals and reduce their defects. The heat causes the atoms to become unstuck from their initial positions (a local minimum of the internal energy) and wander randomly through states of higher energy; the slow cooling gives them more chances of finding configurations with lower internal energy than the initial one. By analogy with this physical process, each step of the SA algorithm replaces the current solution by a random "nearby" solution, chosen with a probability that depends both on the difference between the corresponding function values and also on a global parameter T (called the *temperature*), that is gradually decreased during the process. The dependency is such that the current solution changes almost randomly when T is large, but increasingly "downhill" as T goes to zero [18].

2.4 Pattern Search

Pattern search is a family of numerical optimization methods that do not require the gradient of the problem to be optimized and PS can hence be used on functions that are not continuous or differentiable. Such optimization methods are also known as direct-search, derivative-free, or black-box methods.

The name, pattern search, was coined by Hooke and Jeeves [20]. An early and simple PS variant is attributed to Fermi and Metropolis when they worked at the Los Alamos National Laboratory as described by Davidon [21] who summarized the algorithm as follows:

They varied one theoretical parameter at a time by steps of the same magnitude, and when no such increase or decrease in any one parameter further improved the fit to the experimental data, they halved the step size and repeated the process until the steps were deemed sufficiently small.

3 A Brief History of GPUs

We have already looked at how central processors evolved in both clock speeds and core count. In the meantime, the state of graphics processing underwent a

dramatic revolution. In late 1980s and early 1900s, the growth in popularity of graphically driven operating system such Microsoft Windows helped create a market for a new type of processor. In early 1990s, users began purchasing 2D display accelerators for they personal computers. These display accelerators offered hardware-assisted bitmap operations to assist in the display and usability of graphical operating systems [22]. From a parallel-computing standpoint, NVIDIA's release of the GeForce 3 series in 2001 represents arguably the most important breakthrough in GPU technology. The GeForce 3 series was the computing industry's first chip to implement Microsoft's then-new DirectX 8.0 standard. This standard required that complaint hardware contain both programmable vertex and programmable pixel shading stages. For the first time, developers had some control over the exact computations that would be performed on their GPUs [22].

3.1 CUDA

In November 2006, NVIDIA unveiled the industry's first DirectX 10 GPU, the GeForce 8800 GTX. The GeForce 8800 GTX was also the first GPU to be built with NVIDIA's CUDA Architecture. This architecture included several new components designed strictly for GPU computing and aimed to alleviate many of the limitations that prevented previous graphics processors from being legitimately useful for general-purpose computation [22].

4 Experimental Results

This section presents the experimental results obtained with the optimization methods analyzed in this research. The main contribution of this paper is demonstrate the advantages of use GPUs to calculated complex processes.

To validate the proposed method we used a set of 5 benchmark mathematical functions; all functions were evaluated with different numbers of dimensions, in this case, the experimental results were obtained with 32 dimensions.

Table 1 shows the definitions of the mathematical functions used in this paper. The global minimum for the test functions is 0.

Table 1. Mathematical functions

Function	Definition
De Jong's	$f_1=\sum_{n=i}^{N} x_n^2$
Rotated Hyper-Ellipsod	$f(x)=\sum_{i=1}^{n}\left(\sum_{j=1}^{i} x_j\right)^2$
Rosenbrock's Valley	$f(x)=\sum_{i=1}^{n-1} 100\left(x_{i+1}-x_i^2\right)^2+(1-x_i)^2$
Rastrigin's	$f(x)=10.n+\sum_{i=1}^{n}\left(x_i^2-10.\cos\left(2.\pi x_i\right)\right)$
Griewank's	$f(x)=\sum_{i=1}^{n}\frac{x_i^2}{4000}-\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$

Table 2 and 3 shows the experimental results for the benchmark mathematical functions used in this research using the CPU and GPU to processing of the GA. The Table shows the experimental results of the evaluations for each function with 32 dimensions; where it can be seen the best and worst values obtained, and the average of 50 times after executing the method. Also, the time of processing in seconds is presented.

Table 2. Experimental results with 32 dimensions with GA on CPU

Function	Average	Best	Worst	Time (sec)
De Jong's	0.00094	1.14E-06	0.0056	1.883603
Rotated Hyper-Ellipsod	0.05371	0.00228	0.53997	2.015548
Rosenbrock's Valley	3.14677173	3.246497	3.86201	3.001564
Rastrigin's	82.35724	46.0085042	129.548	1.452212
Griewank's	0.41019699	0.14192331	0.917367	2.548792

Table 3. Experimental results with 32 dimensions with GA on GPU

Function	Average	Best	Worst	Time (sec)
De Jong's	0.000084	1.14E-08	0.00040	0.360003
Rotated Hyper-Ellipsod	0.005371	0.00228	0.53997	0.004590
Rosenbrock's Valley	2.325468	1.97548	3.86201	0.005594
Rastrigin's	70.35724	41.54879	130.598	0.502254
Griewank's	0.31019699	0.04192331	0.917367	0.920154

Table 4 and 5 shows the experimental results for the benchmark mathematical functions used in this research using the CPU and GPU to processing of the PSO. The Table shows the experimental results of the evaluations for each function with 32 dimensions; where it can be seen the best and worst values obtained, and the average of 50 times after executing the method. Also, the time of processing in seconds is presented.

Table 4. Experimental results with 32 dimensions with PSO on CPU

Function	Average	Best	Worst	Time (sec)
De Jong's	5.42E-11	3.40E-12	9.86E-11	2.5442154
Rotated Hyper-Ellipsod	5.42E-11	1.93E-12	9.83E-11	1.2456487
Rosenbrock's Valley	3.2178138	3.1063	3.39178762	1.3659478
Rastrigin's	34.169712	16.14508	56.714207	3.569871
Griewank's	0.0114768	9.17E-06	0.09483	5.2654587

Table 5. Experimental results with 32 dimensions with PSO on GPU

Function	Average	Best	Worst	Time (sec)
De Jong's	2.20E-11	2.40E-12	9.86E-11	0.05040454
Rotated Hyper-Ellipsod	4.20E-11	2.30E-12	9.83E-11	0.02045687
Rosenbrock's Valley	3.1071308	2.16020	3.39178762	0.03659470
Rastrigin's	34.199999	15.14508	53.802564	0.05678710
Griewank's	0.0201564	9.17E-06	0.094831	0.02654580

Table 6 and 7 shows the experimental results for the benchmark mathematical functions used in this research using the CPU and GPU to processing of the SA. The Table shows the experimental results of the evaluations for each function with 32 dimensions; where it can be seen the best and worst values obtained, and the average of 50 times after executing the method. Also, the time of processing in seconds is presented.

Table 8 and 9 show the experimental results for the benchmark mathematical functions used in this research using the CPU and GPU to processing of the PS. The Table shows the experimental results of the evaluations for each function with 32 dimensions; where it can be seen the best and worst values obtained, and the average of 50 times after executing the method. Also, the time of processing in seconds is presented.

Table 6. Experimental results with 32 dimensions with SA on CPU

Function	Average	Best	Worst	Time (sec)
De Jong's	0.1210	0.0400	1.8926	3.0124
Rotated Hyper-Ellipsod	0.9800	0.0990	7.0104	3.0215
Rosenbrock's Valley	1.2300	0.4402	10.790	2.9999
Rastrigin's	25.8890	20.101	33.415	3.2145
Griewank's	0.9801	0.2045	5.5678	4.0555

Table 7. Experimental results with 32 dimensions with SA on GPU

Function	Average	Best	Worst	Time (sec)
De Jong's	0.10100	0.06012	1.2699	1.000124
Rotated Hyper-Ellipsod	0.81200	0.0891	7.1003	1.001015
Rosenbrock's Valley	1.31200	0.40002	10.1290	1.018787
Rastrigin's	25.3256	21.100	32.2315	1.010145
Griewank's	0.99010	0.3050	6.50678	1.000325

Table 8. Experimental results with 32 dimensions with PS on CPU

Function	Average	Best	Worst	Time (sec)
De Jong's	0. 3528	0.2232	2.0779	4.2521
Rotated Hyper-Ellipsod	16.2505	3.1667	25.782	6.2154
Rosenbrock's Valley	4.0568	3.0342	5.7765	5.2565
Rastrigin's	31.4203	25.7660	33.9866	3.25654
Griewank's	0.6897	0.0981	3.5061	2.1548

Figure 1 shows the comparison results between the processing time on GPU and CPU, in the figure is shown the difference in time of each best time obtained in the experiments realized in the paper. The blue line represents the time of processing in CPU and the brown line represent the time of processing in the GPU. Is clear how the best time achieved is when the algorithms were executed on GPU.

Table 9. Experimental results with 32 dimensions with PS on GPU

Function	Average	Best	Worst	Time (sec)
De Jong's	0. 5208	0.1232	2.1579	1.1021
Rotated Hyper-Ellipsod	16.5005	3.6197	25.182	2.1154
Rosenbrock's Valley	4.0588	3.00215	4.2565	2.5105
Rastrigin's	31.5203	25.4530	33.9866	1.6054
Griewank's	0.14970	0.00980	3.5061	1.4858

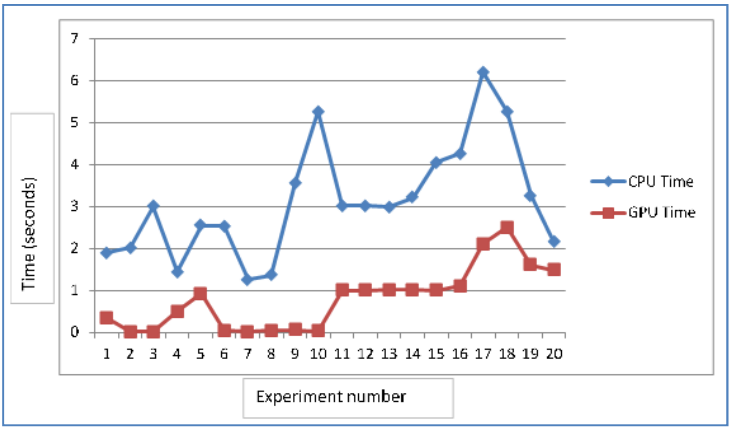


Fig. 1. Comparison results between GPU and CPU

5 Conclusions

The analysis of the experimental results of the bio inspired methods considered in this paper, the FPSO+FGA, lead us to the conclusion that for the optimization of these benchmark mathematical functions the execution on GPU is a good alternative, because it is easier and very fast to optimize and achieve good results than to try it with PSO, GA, SA and GPS on CPU [5], especially when the number of dimensions is increased. This is, because the processing on GPUs if faster than the processing on CPUs. Also, the experimental results obtained with the use of GPUs in this research were compared with other similar approach [17][19], achieving good results quickly.

References

1. Man, K.F., Tang, K.S., Kwong, S.: Genetic Algorithms: Concepts and Designs. Springer (1999)
2. Eberhart, R.C., Kennedy, J.: A new optimizer using particle swarm theory. In: Proceedings of the Sixth International Symposium on Micromachine and Human Science, Nagoya, Japan, pp. 39–43 (1995)
3. Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: Proceedings of IEEE International Conference on Neural Networks, Piscataway, NJ, pp. 1942–1948 (1995)
4. Holland, J.H.: Adaptation in natural and artificial system. The University of Michigan Press, Ann Arbor (1975)
5. Valdez, F., Melin, P.: Parallel Evolutionary Computing using a cluster for Mathematical Function Optimization. In: NAFIPS, San Diego CA, USA, pp. 598–602 (June 2007)
6. Castillo, O., Melin, P.: Hybrid intelligent systems for time series prediction using neural networks, fuzzy logic, and fractal theory. IEEE Transactions on Neural Networks 13(6), 1395–1408 (2002)
7. Fogel, D.B.: An introduction to simulated evolutionary optimization. IEEE Transactions on Neural Networks 5(1), 3–14 (1994)
8. Goldberg, D.: Genetic Algorithms. Addison Wesley (1988)
9. Emmeche, C.: Garden in the Machine. The Emerging Science of Artificial Life, p. 114. Princeton University Press (1994)
10. Valdez, F., Melin, P.: Parallel Evolutionary Computing using a cluster for Mathematical Function Optimization. In: NAFIPS, San Diego CA, USA, pp. 598–602 (June 2007)
11. Angeline, P.J.: Using Selection to Improve Particle Swarm Optimization. In: Proceedings 1998 IEEE World Congress on Computational Intelligence, Anchorage, Alaska, pp. 84–89. IEEE (1998)
12. Back, T., Fogel, D.B., Michalewicz, Z. (eds.): Handbook of Evolutionary Computation. Oxford University Press (1997)
13. Montiel, O., Castillo, O., Melin, P., Rodriguez, A., Sepulveda, R.: Human evolutionary model: A new approach to optimization. Inf. Sci. 177(10), 2075–2098 (2007)
14. Castillo, O., Valdez, F., Melin, P.: Hierarchical Genetic Algorithms for topology optimization in fuzzy control systems. International Journal of General Systems 36(5), 575–591 (2007)
15. Kim, D., Hirota, K.: Vector control for loss minimization of induction motor using GA–PSO. Applied Soft Computing 8, 1692–1702 (2008)
16. Liu, H., Abraham, A.: Scheduling jobs on computational grids using a fuzzy particle swarm optimization algorithm. Future Generation Computer Systems, Article in press
17. Mohammed, O., Ali, S., Koh, P., Chong, K.: Design a PID Controller of BLDC Motor by Using Hybrid Genetic-Immune. Modern Applied Science 5(1) (February 2011)
18. Kirkpatrick, S., Gelatt, C.J., Vecchi, M.: Optimization by Simulated Annealing. Science 220(4598), 671–680 (1983)
19. Valdez, F., Melin, P., Castillo, O.: An improved evolutionary method with fuzzy logic for combining Particle Swarm Optimization and Genetic Algorithms. Appl. Soft Comput. 11(2), 2625–2632 (2011)

20. Hooke, R., Jeeves, T.A.: Direct search' solution of numerical and statistical problems. Journal of the Association for Computing Machinery (ACM) 8(2), 212–229 (1961)
21. Davidon, W.C.: Variable metric method for minimization. SIAM Journal on Optimization 1(1), 1–17 (1991)
22. Sanders, J., Kandrot, E.: CUDA BY EXAMPLE An Introduction to General-Purpose GPU Programming. Addisson Wesley (2011)

Variants of Ant Colony Optimization: A Metaheuristic for Solving the Traveling Salesman Problem

Iván Chaparro and Fevrier Valdez

Tijuana Institute of Technology, Tijuana, México

Abstract. Ant Colony Optimization (ACO) has been used to solve several optimization problems. However, in this paper, the variants of ACO have been applied to solve the Traveling Salesman Problem (TSP), which is used to evaluate the variants ACO as Benchmark problems. Also, we developed a graphical interface to allow the user input parameters and having as objective to reduce processing time through a parallel implementation. We are using ACO because for TSP is easily applied and understandable. In this paper we used the following variants of ACO: Max-Min Ant System (MMAS) and Ant Colony System (ACS).

Keywords: ACO, TSP, Optimization, Combinatorial Problems.

1 Introduction

There are different algorithms based on the simulation of natural processes and genetics such as genetic algorithms and Ant Colony Optimization (ACO), based on heuristic problem solving. Currently required to solve more complex problems which require too much processing time for result. Therefore, we can work with highly complex problems getting results with less processing time with a parallel implementation. In this paper we describe several variants of Ant Colony Optimization (ACO) to solve the Traveling Salesman Problem (TSP) allowing the user input parameters using a graphical interface and performing parallel processing.

2 ACO Variants

The ACO metaheuristic is inspired by observing the behavior of real ant colonies, which presented as an interesting feature how to find the shortest paths between the nest and food, on its way the ants deposit a substance called pheromone, this trail allows the ants back to their nest from the food; it uses the evaporation of pheromone to avoid a unlimited increase of pheromone trails and allow to forget the bad decisions, thus avoiding the persistence of the pheromone trails and therefore, the stagnation in local optima [4].

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 323–331.
springerlink.com

2.1 Traveling Salesman Problem (TSP)

This problem is defined as to visit "n" cities, starting and ending with the same city, visiting each city once and making the tour with the lowest cost, this cost can be expressed in terms of time or distance, i.e., travel a minimum of kilometers or perform a tour in the shortest time possible. More formally, the TSP can be represented by a complete weighted graph G= (N, A) with N being the set of nodes representing the cities, and A being the set of arcs. Each arc (i, j) ∈ A is assigned a value (length) d_{ij}, which is the distance between cities i and j, with i,j ∈ N. In the general case of the asymmetric TSP, the distance between a pair of nodes i,j is dependent on the direction of traversing the arc, that is, there is at least one arc (i, j) for $d_{ij} \neq d_{ji}$. In the symmetric TSP, $d_{ij} = d_{ji}$, holds for all the arcs in A. The goal in the TSP is find a minimum length Hamiltonian circuit of the graph, where a Hamiltonian circuit is a closed path visiting each of n=$|N|$ nodes of G exactly once. Thus, an optimal solution to the TSP is a permutation π of the node indices {1,2,…,n} such that the length $f(\pi)$ is minimal, where $f(\pi)$ is given by[4]:

$$f(\pi) = \sum_{i=1}^{n-1} d_{\pi(i)\pi(i+1)} + d_{\pi(n)\pi(1)} \quad (1)$$

2.2 Max-Min Ant System

This algorithm introduces four main modifications with respect to the Ant System [3]:

- It strongly exploits the best tours found.
- It limits the possible range of pheromone trail values to the interval $[\tau_{min}\ \tau_{max}]$.
- The pheromone trails are initialized to the upper pheromone trail limit, which, together a small pheromone evaporation rate, increases the exploration of tours at the start of the search.
- Pheromone trails are reinitialized each time the system approaches stagnation or when no improved tour has been generated for a certain numbers of consecutive iterations.

2.3 Ant Colony System (ACS)

This algorithm was proposed by Dorigo and Gambardella in the year 1997, and differs from Ant System in three main points [4]:

- It exploits the search experience accumulated by the ants more strongly than Ant System does through the use of a more aggressive action choice rule.

- Pheromone evaporation and pheromone deposit take a place only on the arcs belonging to the best-so-far tour.
- Each time an ant uses an arc (i,j) to move from city i to city j, it removes some pheromone from the arc to increase the exploration of alternative paths.

3 Graphical Interface in Matlab

A graphical Interface was developed with the objective of introducing parameters for the variants of ACO. The first user interface was made only as presentation to enter the options menu as shown in the figure 1.

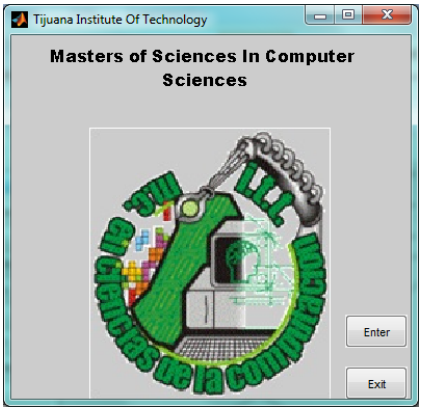


Fig. 1. Main Interface

Or to exit press the button exit, showing the following confirmation message:

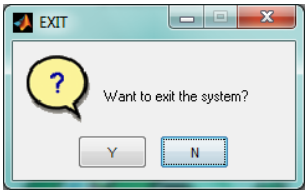


Fig. 2. Interface confirmation message

Figure 3, select the type of variant and method of execution or press the back button to go to the previous interface.

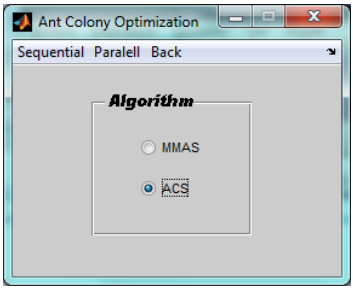


Fig. 3. Menu of options

Figure 4, show the execution time and minimum distance that is the cost of taking the tour of cities in addition to plot the location of cities

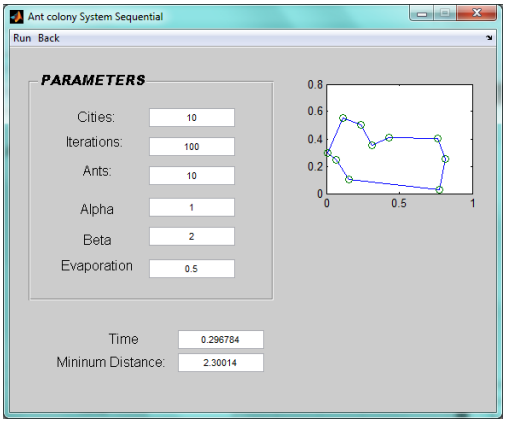


Fig. 4. Interface to introduce parameters

4 Results

This section shows the results obtained from experiments with the different variants of ACO with sequential and parallel processing.

30 experiments were performed where vary alpha, beta, rho factor and the number of iterations using 10 cities, the results are shown in table 1, we obtained a minimum distance of 2.3001 with 100 and 7 iterations of the algorithm; however, for less generations the processing time is lower reducing the execution time to 0.113762 seconds in experiment number 26. Average was obtained by execution of 1.48 seconds. An experiment was performed only with 1000 cities for more complexity to the algorithm and obtained an execution time of 16 seconds.

Table 1. Sequential Variant for ACS

EXPERIMENT	CITIES	ANTS	ITERATIONS	ALPHA	BETA	RHO	MINIMUN DISTANCE	TIME
1	10	100	100	1	2	0.3	2.4064	**2.297667**
2	10	100	100	1	2	0.3	2.4064	**2.204522**
3	10	100	100	1	2	0.3	2.4064	**2.151696**
4	10	100	100	1	2	0.3	2.4064	**2.191324**
5	10	100	100	1	2	0.3	2.4064	**2.1652**
6	10	100	100	1	2	0.3	2.4064	**2.185889**
7	10	100	100	1	2	0.3	2.4064	**2.171482**
8	10	100	100	1	2	0.3	2.4064	**2.172033**
9	10	100	100	1	2	0.3	2.4064	**2.175278**
10	10	100	100	1	2	0.3	2.4064	**2.379699**
11	10	100	100	2	3	0.5	2.3001	**2.293828**
12	10	100	100	2	3	0.5	2.3001	**2.343802**
13	10	100	100	2	3	0.5	2.3001	**2.364691**
14	10	100	100	2	3	0.5	2.3001	**2.3829**
15	10	100	100	2	3	0.5	2.3001	**2.330182**
16	10	100	100	2	3	0.5	2.3001	**2.277928**
17	10	100	100	2	3	0.5	2.3001	**2.261657**
18	10	100	100	2	3	0.5	2.3001	**2.288932**
19	10	100	100	2	3	0.5	2.3001	**2.321291**
20	10	100	10	2	3	0.5	2.3001	**0.301492**
21	10	100	10	2	3	0.5	2.3001	**0.266089**
22	10	100	10	2	3	0.5	2.3001	**0.269683**
23	10	10	10	2	3	0.5	2.3001	**0.072738**
24	10	10	9	2	3	0.5	2.3001	**0.125904**
25	10	10	8	2	3	0.5	2.3001	**0.078677**
26	10	10	7	2	3	0.5	2.3001	**0.113762**
27	10	10	6	2	3	0.5	2.5144	**0.075725**
28	10	10	5	2	3	0.5	2.5319	**0.081266**
29	10	10	4	2	3	0.5	2.4064	**0.07106**
30	10	10	3	2	3	0.5	2.53.19	**0.078017**
31	1000	100	10	2	3	0.5	40.4873	997.667909
							1.483147133	**Seconds**
							16.62779848	**Seconds**

The same numbers of experiments were performed using the same parameters for parallel execution, finding the minimum distance of 2.3001 also with 7 iterations but with less processing time of 0.079525 second in experiment number 26. In the parallel implementation are executed 4 processes simultaneously, therefore, the average by execution of one process is 0.01988125 seconds and obtained an average of all experiments of 1.05 seconds and obtained 8.7125918 seconds for the experiment was done with 1000.

MMAS variant to obtain the minimum distance of 46.5519 in all experiments and only 2 iterations is obtained the same distance showing the result in table 3.Average was obtained by execution of all experiments of 5.81 seconds and performed an experiment with a 22 cities with a time of 20.329047 seconds.

In parallel the same distance was obtained only reducing the processing time. The results are shown in Table 4.

Table 2. Parallel Variant for ACS

EXPERIMENT	CITIES	ANTS	ITERATIONS	ALPHA	BETA	RHO	MINIMUN DISTANCE	lab1	lab2	lab3	lab4
1	10	100	100	1	2	0.3	2.3001	4.985979	4.670103	**5.803356**	4.450128
2	10	100	100	1	2	0.3	2.3001	4.717038	**5.766149**	4.855537	4.638474
3	10	100	100	1	2	0.3	2.3001	**5.663317**	4.626035	4.773888	4.315661
4	10	100	100	1	2	0.3	2.3001	4.76076	4.650837	**5.480135**	4.432192
5	10	100	100	1	2	0.3	2.3001	4.562846	4.682623	4.794606	**5.590443**
6	10	100	100	1	2	0.3	2.3001	4.793052	4.706704	**5.581461**	4.903088
7	10	100	100	1	2	0.3	2.3001	4.675758	4.627855	**5.557725**	4.282445
8	10	100	100	1	2	0.3	2.3001	5.676394	4.586213	**5.067412**	4.835595
9	10	100	100	1	2	0.3	2.3001	4.690962	**5.544794**	4.667791	4.765641
10	10	100	100	1	2	0.3	2.3001	4.640137	4.616093	**5.856676**	4.217882
11	10	100	100	2	2	0.5	2.3001	4.740937	4.697812	4.502633	**5.72732**
12	10	100	100	2	2	0.5	2.3001	4.509803	4.65397	4.752962	**5.862888**
13	10	100	100	2	2	0.5	2.3001	5.163928	4.935564	4.534519	**5.81736**
14	10	100	100	2	2	0.5	2.3001	5.184917	**5.842456**	4.668559	4.347392
15	10	100	100	2	2	0.5	2.3001	4.7837	4.693819	4.962278	**6.054081**
16	10	100	100	2	2	0.5	2.3001	4.615381	**5.656933**	4.580497	4.195778
17	10	100	100	2	2	0.5	2.3001	**5.58897**	4.735533	5.434506	4.359453
18	10	100	100	2	2	0.5	2.3001	5.051927	4.583253	**5.678237**	4.344494
19	10	100	100	2	2	0.5	2.3001	4.30736	4.675053	4.67073	**5.894579**
20	10	100	100	2	2	0.5	2.3001	4.667901	4.375274	4.707069	**5.797504**
21	10	100	100	2	2	0.5	2.3001	**5.730341**	4.426355	4.604745	4.909657
22	10	100	100	2	2	0.5	2.3001	4.583651	4.664926	**5.881932**	4.22536
23	10	10	10	2	2	0.5	2.3001	0.070359	**0.077827**	0.070337	0.05964
24	10	10	9	2	2	0.5	2.3001	**0.108555**	0.072157	0.072857	0.05581
25	10	10	8	2	2	0.5	2.3001	**0.096793**	0.065903	0.045364	0.072012
26	10	10	7	2	2	0.5	2.3001	0.062876	0.058779	**0.079525**	0.045117
27	10	10	6	2	2	0.5	2.5144	0.052096	0.04201	**0.106847**	0.042266
28	10	10	5	2	2	0.5	2.5319	0.040768	**0.061523**	0.03778	0.052522
29	10	10	4	2	2	0.5	2.4064	**0.132485**	0.060001	0.047999	0.046622
30	10	10	3	2	2	0.5	2.5319	**0.059991**	0.044205	0.050919	0.032897
31	1000	100	10	2	2	0.5	40.4873	1038.578504	1045.511016		
								1.051396792 Seconds			
								8.7125918 Seconds			

Table 3. Sequential Variant for MMAS

EXPERIMENT	CITIES	ANTS	ITERATIONS	ALPHA	BETA	RHO	MINIMUN DISTANCE	TIME
1	10	100	100	1	2	0.3	46.5519	**8.513724**
2	10	100	100	1	2	0.3	46.5519	**8.540868**
3	10	100	100	1	2	0.3	46.5519	**8.561641**
4	10	100	100	1	2	0.3	46.5519	**8.786021**
5	10	100	100	1	2	0.3	46.5519	**8.463686**
6	10	100	100	1	2	0.3	46.5519	**8.513592**
7	10	100	100	1	2	0.3	46.5519	**8.530144**
8	10	100	100	1	2	0.3	46.5519	**8.531728**
9	10	100	100	1	2	0.3	46.5519	**8.537607**
10	10	100	100	1	2	0.3	46.5519	**8.532742**
11	10	100	100	2	3	0.5	46.5519	**8.757501**
12	10	100	100	2	3	0.5	46.5519	**8.574803**
13	10	100	100	2	3	0.5	46.5519	**8.584994**
14	10	100	100	2	3	0.5	46.5519	**8.647263**
15	10	100	100	2	3	0.5	46.5519	**8.658588**
16	10	100	100	2	3	0.5	46.5519	**8.748089**
17	10	100	100	2	3	0.5	46.5519	**8.594535**
18	10	100	100	2	3	0.5	46.5519	**8.743204**
19	10	100	100	2	3	0.5	46.5519	**8.732453**
20	10	100	10	2	3	0.5	46.5519	**0.996702**
21	10	100	10	2	3	0.5	46.5519	**0.985875**
22	10	100	10	2	3	0.5	46.5519	**0.980589**
23	10	10	10	2	3	0.5	46.5519	**0.991487**
24	10	10	9	2	3	0.5	46.5519	**0.985393**
25	10	10	8	2	3	0.5	46.5519	**0.988249**
26	10	10	7	2	3	0.5	46.5519	**0.990082**
27	10	10	6	2	3	0.5	46.5519	**0.989426**
28	10	10	5	2	3	0.5	46.5519	**0.98293**
29	10	10	4	2	3	0.5	46.5519	**0.979736**
30	10	10	3	2	3	0.5	46.5519	**0.992027**
31	22	100	100	2	3	0.5	75.3097	20.329047
							5.813855967	**Seconds**
							20.329047	**Seconds**

Table 4. Parallel Variant for MMAS

EXPERIMENT	CITIES	ANTS	ITERATIONS	ALPHA	BETA	RHO	MINIMUN DISTANCE	lab1	lab2	lab3	lab4
1	10	100	100	1	2	0.3	46.5519	15.616404	15.656199	15.725126	**15.825714**
2	10	100	100	1	2	0.3	46.5519	15.678724	15.587746	15.675984	**15.923952**
3	10	100	100	1	2	0.3	46.5519	15.774443	15.751579	15.747124	**15.884351**
4	10	100	100	1	2	0.3	46.5519	**15.771519**	15.598393	15.7615	15.716631
5	10	100	100	1	2	0.3	46.5519	15.910909	15.829529	15.946475	**16.128292**
6	10	100	100	1	2	0.3	46.5519	15.742674	15.663364	15.72186	**15.988445**
7	10	100	100	1	2	0.3	46.5519	15.708747	15.692472	15.700473	**15.800296**
8	10	100	100	1	2	0.3	46.5519	15.767879	15.576825	15.869953	**15.948438**
9	10	100	100	1	2	0.3	46.5519	15.919635	15.658901	**15.972495**	15.800514
10	10	100	100	1	2	0.3	46.5519	15.724937	15.706604	15.862844	**15.87672**
11	10	100	100	2	3	0.5	46.5519	15.861694	15.737939	15.916608	**16.02257**
12	10	100	100	2	3	0.5	46.5519	**15.872514**	15.749501	15.821333	15.8214
13	10	100	100	2	3	0.5	46.5519	**15.975199**	15.763913	15.827095	15.844473
14	10	100	100	2	3	0.5	46.5519	15.827191	15.798001	15.831821	**16.165865**
15	10	100	100	2	3	0.5	46.5519	15.827191	15.798001	15.831821	**16.165865**
16	10	100	100	2	3	0.5	46.5519	15.891914	15.836529	16.037666	**16.1051**
17	10	100	100	2	3	0.5	46.5519	15.874714	15.852839	**16.935693**	15.865242
18	10	100	100	2	3	0.5	46.5519	15.87535	15.805074	**15.996528**	15.843767
19	10	100	100	2	3	0.5	46.5519	15.866498	15.82524	15.784151	**16.050466**
20	10	100	100	2	3	0.5	46.5519	15.850193	15.708793	15.816849	**16.069687**
21	10	100	100	2	3	0.5	46.5519	15.823103	15.77423	**16.148439**	15.962383
22	10	100	100	2	3	0.5	46.5519	15.872622	15.737472	**16.052308**	15.989298
23	10	10	10	2	3	0.5	46.5519	1.765158	**1.765994**	1.761965	1.729142
24	10	10	9	2	3	0.5	46.5519	1.737158	**1.790317**	1.788179	1.740829
25	10	10	8	2	3	0.5	46.5519	1.750755	1.71053	1.783875	**1.794154**
26	10	10	7	2	3	0.5	46.5519	1.778715	1.790239	1.770882	**1.825998**
27	10	10	6	2	3	0.5	46.5519	1.757422	1.717733	1.760698	**1.844163**
28	10	10	5	2	3	0.5	46.5519	**1.786446**	1.754178	1.772288	1.771232
29	10	10	4	2	3	0.5	46.5519	1.741479	1.724236	**1.784747**	1.783529
30	10	10	3	2	3	0.5	46.5519	**1.80604**	1.687742	1.78379	1.770516
31	22	100	100	2	3	0.5	75.3097	38.742169	38.24625	37.974074	38.057135
								3.058986 Seconds			
								9.6855423 Seconds			

5 Conclusions

Experiments were performed using the same parameters and processes executing sequentially and parallel form with different algorithms, ACS and MMAS. Similar results were obtained but with less processing time a parallel implementation. In the experiments where we used 1000 cities with variant ACS, time is doubled in the sequential implementation with a time of 16 minutes and in parallel only use 8 minutes. Subsequently, the variant MMAS was performed an experiment with 22 cities and the best processing was of 9.68 seconds obtained in parallel. Using a parallel implementation we can run complex optimization problems, for example, using a larger number de cities and getting results with less processing time.

References

[1] Almirón, M., Barán, B., Chaparro, E.: Ant Distributed System for Solving the Traveling Salesman Problem. In: XXV lnformatic Latinoamerican Conf.-CLEI, Paraguay, pp. 779–789 (1999)

[2] Barán, B., Sosa, R.: A New approach for AntNet routing. In: IEEE Ninth International Conference onComputer Commnunications and Networks, Las Vegas, Nevada (2000)
[3] de la Cruz, J., Mendoza, A., del Castillo, A., Paternina, C.: Comparative Analysis of heuristic Approaches Ant Q, Simulated Annealing and Tabu Search in Solving the Traveling Salesman. Departamento de Ciencias de la Computación e Inteligencia Artificial, E.T.S. Ingenieria Informática, Granada, España (2003)
[4] Dorigo, M., Stutzle, T.: Ant Colony Optimization. Massachusetts Institute of Technology, MIT Press, Bradford, Cambridge (2004)
[5] Favaretto, D., Moretti, E., Pellegrini, P.: Ant colony system for variants of traveling salesman problem with time windows, Technical Report. Applied Mathematics Department of Ca' Foscari University of Venice, No. 120/2004 (2004)
[6] Cordón, O., Moya, F., Zarco, C.: A new evolutionary algorithm combining simulated annealing and genetic programming for relevance feedback in fuzzy information retrieval systems. Soft Computing 6(5), 308–319 (2002)
[7] Pavez, A., Acevedo, H.: An Algorithm ACS Motion Prompt and Operator 2-Opt, Departamento de Informatica. Universidad Técnica Federico Santa Maria (2002)
[8] Website of Ant Colony Optimization Algorithms official, `http://www.aco-metaheuristic.org` (accessed May 5, 2012)
[9] Website of interface design, `http://www.matpic.com`, MC. Diego O. Barragán Guerrero, Universidad Estatal de Campinas, Brasil, `http://www.unicamp.br` (accessed May 2012)
[10] Website of Matlab, `http://www.mathworks.com` (accessed May 2012)

Part IV

Intelligent Optimization Methods and Applications

Memetic Algorithm for Solving the Problem of Social Portfolio Using Outranking Model

Claudia G. Gómez S.[1], Eduardo R. Fernández Gonzalez[2], Laura Cruz Reyes[1], S. Samantha Bastiani M[1], Gilberto Rivera Z.[1], and Victoria Ruíz M.[1]

[1] Instituto Tecnológico de Ciudad Madero, 1ro. de Mayo y Sor Juana I. de la Cruz s/n CP. 89440, Tamaulipas, México
{cggs71,b_shulamith}@hotmail.com,
{riveragil,victoria.rzrz}@gmail.com,
lcruzreyes@prodigy.net.mx

[2] Universidad Autónoma de Sinaloa (UAS), Justicia Social SN, Ciudad Universitaria, 81229 Los Mochis, Sinaloa. (52) 668 812 33 01
eddyf@uas.uasnet.mx

Abstract. The government institutions at all levels, foundations with private funds or private companies that support social projects receiving public funds or budget to develop its own social projects often have to select the projects to support and allocate budget to each project. The choice is difficult when the available budget is insufficient to fund all projects or proposals whose budget requests have been received, together with the above it is expected that approved projects have a significant social impact. This problem is known as the portfolio selection problem of social projects. An important factor involved in the decision to make the best portfolio, is that the objectives set out projects that are generally intangible, such as the social, scientific and human resources training. Taking into account the above factors in this paper examines the use of multi objective methods leading to a ranking of quality of all selected projects and allocates resources according to priority ranking projects until the budget is exhausted. To verify the feasibility of ranking method for the solution of problem social portfolio constructed a population memetic evolutionary algorithm, which uses local search strategies and cross adapted to the characteristic of the problem. The experimental results show that the proposed algorithm has a competitive performance compared to similar algorithms reported in the literature and on the outranking model is a feasible option to recommend a portfolio optimum, when little information and the number of projects is between 20 and 70.

1 Introduction

Organizations are constantly making decisions about how to invest and managing their resources to satisfy social needs, as: money, time, equipment and people, among others. Usually resources are lower than demand, resulting in not being

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 335–348.
springerlink.com

able to provide benefits to all competing projects because of these organizations are forced to select the best portfolio that will consist of a subset of projects that maximizes the benefits social. One of the main tasks of the manager is to select projects that best meet the objectives of the company [12]. Incorrect decisions regarding the selection of projects have two main consequences: i) resources generally limited, are wasted on projects that, although they may be good, not the most appropriate for the company and ii) the organization loses the benefits that could have gotten if it had invested in more suitable projects.

The selection of projects for a portfolio of social projects needs special treatment for the following reasons [7]:

1. The quality of projects is generally described by multiple criteria that are often in conflict.
2. Typically, requirements are not accurately known. Many concepts have no mathematical support for having entirely subjective nature.
3. Heterogeneity, or differences between the objectives of the projects, makes it difficult to compare.

These features of the *Social Portfolio Problem* (*SPP*) represent a challenge for multi-objective optimization algorithms [8]. Moreover, although optimal solutions are found, the problem has not been completely resolved. Even the decision maker has the task of implementing just one of the alternatives presented. The decision maker will evaluate the alternatives according to her/his criteria and preferences.

In this paper we are interested in solving the Social Portfolio Problem using a set of ranked projects as input data instance and thereafter shall be adopted in preference relations using methods of categorization and outranking model. All these of these features will be included in a memetic algorithm that aims to generate the best recommendation for the decision maker.

2 Background

In order to place the research in context, this section is divided in six parts. The first part define portfolio problem and the second part explain social project. The third part defines multi-objetive optimization and the next part describes the problem formulation, finally the last part explains multi-objective evolutionary algorithm and memetic algorithm.

2.1 Portfolio Problem

A *project* is a temporary, unique and unrepeatable process which pursues a specific set of objectives [2]. In this work, it is not considered that the projects can be broken down into smaller units such as tasks or activities. In other words, a project cannot be divided to run only a part, however, different versions of the same project can be proposed, each version may vary in amount of activity, time required and requested resources.

A *portfolio* consists of a set of projects that can be performed in the same period of time [2]. For this reason, projects in the same portfolio share available resources in the organization. Thus, it is not sufficient to compare the projects individually, but must compare groups of projects to identify what portfolio makes the greatest contribution to the organization objectives.

The proper selection of projects to integrate the portfolio, which will receive the organization's resources, is one of the most important decision problems for both public and private institutions [3, 11]. The main economic and mathematical models to the portfolio problem assume that there is a defined set of n projects, each project well characterized with costs and revenues, of which the distribution over time is known. The *Decision Maker* (DM) is responsible for selecting the portfolio that the company will implement [7].

2.2 Social Projects

Social projects are characterized by objectives whose fulfillment benefits society. These objectives are generally intangible, such as social and scientific impact, as well as human resource training, among others, without regard to potential economic benefit as the main element of measure. In addition, the amount of desired objectives in these projects can be of several tens, depending on the level of detail and the conditions under which it is restricted.

It is also important to note that such projects are usually assigned to one area and region. The project area is mainly the social sector, e.g. education, public health, safety, scientific development, among others. The region is primarily concerned with the physical area that will benefit, for example by state, county, district, or similar. Thus, to form social portfolios should be considered [11]:

1. No area/region monopolizes most of the budget, leaving remaining areas/regions with poor resources.
2. All areas/regions receive at least a minimal budget, ensuring its permanence and growth.

2.3 Multi-objective Optimization

From the several emergent research areas in which EAs have become increasingly popular, multi-objective optimization has had one of the fastest growing in recent years. A multi-objective optimization problem (MOP) differs from a single-objective optimization problem because it contains several objectives that require optimization [4].

Real-world optimization problems are extremely complex with many attributes to evaluate and multiple objectives to optimize [4, 15]. The attributes correspond to quantitative values that describe the problem and are expressed in terms of decision variables. The objectives are the directions for improvement of the attributes and can be to maximize or minimize.

In many cases, due to the conflicting nature of attributes is not possible to obtain a single solution and therefore the ideal solution for a MOP cannot be achieved because there is no one solution the problem. Typically, solving a MOP has a set of solutions that reached an aspiration level expected by the DM [4].

Therefore they are solutions that, although different, their performance is mathematically equivalent and cannot be overcome on both objectives simultaneously without leaving the feasible solution space [9]. This set of solutions is called the Pareto front, and find it is one of the main purposes of solving a MOP [6].

But finding the Pareto front does not completely solve a MOP. Now the DM should choose a solution from the front, according to his/her own criteria. This is not a difficult task if you are managing two or three objectives. However, when the number of objectives increases, three major difficulties arise:

1. The capacity of algorithms for finding the Pareto front is rapidly degraded [23]. It becomes extremely difficult for the DM, and even impossible, to establish valid criteria for comparing solutions when there are conflicting objectives [9].
2. The size of the Pareto front can grow exponentially with respect to the number of objectives. This complicates the task of the DM to choose a solution [9].

A *compromise solution* is understood as a Pareto solution in which the objectives achieved acceptable values for the DM, and therefore could be selected. The *best compromise* is the compromise solution that meets best the preferences of the DM. Thus, the solution to a MOP is not only finding the Pareto front, but also to identify the best compromise.

Identify the Pareto front (or at least an approximation) has been commonly the task of multi-objective algorithms, leaving the identification of the best compromise to the user. However, a typical DM is capable of processing only at most five to ten pieces of information at once [14], thus being unable to identify the best compromise when he/she needs to compare sets of solutions of a MOP over five or nine objectives. To address this problem requires the creation of algorithms for MOP that show a set of solutions as small as possible, but without discarding those that the DM could choose as a final solution.

Since all Pareto solutions are mathematically equivalent, the DM should provide information about his/her preferences to MOP algorithms. Such information can be provided before or after to generate the Pareto solutions or the process can be interactive, progressively consulting DM preferences [8].

2.4 Problem Formulation

The ***portfolio selection problem*** is defined by Fernández [8, 10, 21] based on the following premises:

- Consider a set A composed by N_A competing and non-interacting projects.
- Each project is described by a set of attributes Q that specify their quality as public-policy, and each project entails a somewhat imprecise request for funds. N_A is considered to be a large number, and Q contains both tangible and intangible attributes.

- A *DM*, representing the higher-level preferences in the organization, is assumed.
- It is assumed that some multicriteria-analysis technique was previously applied to a set of competing projects. Thus, unacceptable projects have been eliminated in advance and the remaining projects are ranked in a quality-descendant ordering.
- This ranking is the input information for the problem. Let A' denote the set of acceptable projects and $N = card(A')$.
- The projects may be classified (e.g., regarding activity, function or geographical impact) in accordance with some demands from the *DM*.
- There is a fixed budget that intends to be fairly distributed among the projects.

The solution concept consists in establishing a subset C of A' whose components will be financed according to specific assignations. In what follows, C will denote the project portfolio.

In first place the *DM* should weigh the number of supported projects against quality. As stated, the quality of the projects is comprised in the ranking. The conventional way for allocating resources according to ranks arises from the necessity of respecting the information on the projects' quality. However, a rank contains imprecise information. A ranking is basically qualitative and depends on the multi-criteria evaluation method that is applied for its construction.

2.5 Multi-Objective Evolutionary Algorithms

MOEA have become a popular technique for solving multi-objective problems [4]. Thus, using MOEA, the DM does not need to do a series of optimizations for each objective, as is usually done in the methods of operations research [22, 18].

The objective of a *Multi-Objective Evolutionary Algorithms* (MOEA) is to converge to the true Pareto front of a problem which normally consists of a diverse set of points. MOPs can present an uncountable set of solutions, which when evaluated produce vectors whose components represent trade-offs in decision space [4].

However, a limitation on the MOEA is the fact that only involves the process of finding a solution set without considering the most important aspect, the decision process. Most current approaches to MOEA are focused on finding an approximation to the optimal set of Pareto front, however, identify the best compromise has usually been omitted.

2.6 Memetic Algorithm (MAs)

The method is based on a population of agents and proved to be of practical success in a variety of problem domains and in particular for the approximate solution of NP-hard optimization problems [17].

An important characteristic is the use of a meme, suggests that in cultural evolution processes, information is not simply transmitted unaltered between individuals. This enhancement is accomplished in MAs by incorporating heuristics, approximation algorithms, local search techniques, specialized recombination operators, truncated exact methods, etc. In essence, most MAs can be interpreted as a search strategy in which a population of optimizing agents cooperate and compete. The success of MAs can probably be explained as being a direct consequence of the *synergy* of the different search approaches they incorporate [17]. The Figure 1 shows the basic template of MAs.

```
0 Memetic Algorithm ()
1 Population←RandomGeneratepopulation( );
2 ImprovePopulation←Local Search (Population);
3 while (not stop condition)
4       if stagnation then
5 Population←Select best individual (Population);
6 Population←Random Generate Population( );//diversification
7 ImprovePopulation←(Population);
8       end_if
9       for i←1..#crossoversdo
10        select π_a, π_b from Population( );
11        offsprings←crossover( );
12      end_for
13      Population←SortPopulation(Population);
14      Best_individual←Select_best(SortedPopulation)
15 end_while;
16 end_Memetic Algorithm
```

Fig. 1. Template basic of the MAs

3 Proposed Algorithm

This section describes in detail the *Memetic Algorithm* (*MAs*) used for SPP and the method to evaluate the solutions in according to the model proposed. The *MAs* are evolutionary algorithms that are intimately coupled with local search algorithms, resulting in a population-based algorithm at effectively searches in the space of local optimal.

The Figure 2 shows the proposed *MAs* that we used in our implementation. In the **line 1**, a portfolio initial is obtained, the first individual is generated by following the order provided by the ranking; it is assumed a distribution of the budget among the projects. Then, the rest of population is randomly built; we experimented with a population of *200* individuals.

```
Memetic Algorithm ( )
1   Genes←Generate_ population ( );
2   do
3       EvaluatePopulation (Genes);
4       //Selection of parents per tournament
5           for i=0 to i<population
6                   Candidate1←randomPosition(Genes)
7                   Candidate2←randomPosition(Genes)
8                   while (candidato1 != candidato2);
9                           if (netFlow_calculation(Candidate1)
                            <netFlow_calculation(Candidate2))
10                              parents[i]←Candidate1;
11                          else
12                              parents[i+1]←Candidate2;
13                          end_if
14                  end_while
15          end_for
16      //Crossover
17          for i=0 to i<population
18              Cross_point←randomPosition(n_projects);
19              for j=0 to i<Cross_point
20                      temp[i]←parents[i];
21              end_for
22              for j=Cross_point to i<n_projects
23                      temp[i]← parents[i+1];
24              end_for
25              for int j=0 to i<n_projects
26                      offspring[i]←temp[i];
27              end_for
28          end_for
29      LocalSearch( );
30      EvaluatePopulation( );
31  end_while // until reach the set number of generations
```

Fig. 2. Memetic algorithm propose

From **line 2 to 31**,in each iteration (generation) the memetic procedure is shown, this algorithm is composed by a set of operators, evaluating population, parent's selection, crossover and local search, these operators are described below. In the **line 3** the population is evaluated, this means that the current values maximums local and global of the solutions are calculated. Then, from **line 4** ***to 15***, the parent's selection is performed per tournament, each individual from the population are evaluate to determine the best candidate in according with the *net flow calculation*. Soon, from the **line 16 to 28** the crossover operator is performed by a random point and exchanging the two individuals selected before, so are generated new individuals. In the **line 29**, the local search is applied to a solution generated by the genetic operators; once again an evaluation of population is performed.

To determine the *calculation of net flow* [11], consider a comparison between solutions composing a set *E*, let *x, y* be members of E. By analogy with outranking methods [17], then may be defined the affirmation "x outranks y" as: $\sigma(x,y) = \Sigma w_i$, where the sum is carried out upon the indices of the criteria where "x is at least

as preferred as y" holds, then an outranking net flow measure is associated to every $a \in E$, as follows: $fn\ (a)\ \Sigma\ \sigma\ (a,x)$ - $\Sigma\ \sigma(x,a)$, in which the sum involves every $x \in E$, $x \neq a$. The *outranking net flow* is considered as a quality criterion for every solution.

The Figure 3 describes the procedure to compare the solutions in the model proposed [21]; this validation is given by a net flow calculation and each solution is characterized by three attributes (strong disagreements, weak disagreements and cardinality of the portfolio).

Net Flow Calculation ( )
1 Categorization of projects ( );
2 Calculating preference relations ( );
3 Calculation superiority ( );
4 Calculation weak disagreements ( );
5 Calculation strong disagreements ( );
6 return net flow

Fig. 3. Outranking model procedures

To illustrate the procedure of calculations shown in the Figure 4, first suppose a normalized weight *w* is assigned by the DM to each attribute, acting as a measure of its importance. In step one performs the categorization of projects, from a given instance of ranked projects; this categorization is given by 5 divisions, which are vanguard, high-medium, medium, low-medium and rearguard as shows the example in the Figure 4.

Id:	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Portfolio:	1	0	1	1	0	1	0	1	1	1	0	1	1	1	0	1	0	1	0	1
Ranking:	1	2	2	3	4	5	5	6	7	7	8	9	10	11	11	12	13	14	15	15
Costs:	400	300	200	150	300	250	300	450	150	100	150	50	200	200	350	250	150	350	400	200

vanguard *high-medium* *medium* low-medium *rearguard*

Fig. 4. Example of the categorization of the initial portfolio

Then a calculation of preference relation is executed, and the possible values that can make this calculation are: absolute, strict, weak and indifferent. These are shown in the Table 1, which are calculated to institute a relation of superiority between projects, considering the veto and concordance conditions as shown in the Table 2 the thresholds *V1, V2* and *U1* are attributes which express cost.

The relation of superiority is given formally considering (*a*,*b*) projects such that $b \in$ Portfolio and $a \notin$ Portfolio, then a relation of superiority indicates that *a*S'*b* if

and only if $(a >> b) \vee (a > b) \vee (a > \sim b)$ and there are no veto conditions. Finally to generate the net flow calculation is necessary to calculate the weak and strong disagreements in addition with the cardinality of portfolio.

The disagreement is calculated by the next rules: Let *C* a set of projects which integrate the portfolio, formally a disagreement is calculated by: $D = \{(a,b) \in A\ x\ A \mid aS'b,\ b \in C\ y\ a \notin C\}$; To calculate weak disagreements must be considered the relation of weak preference as follows: $D_d = \{(a,b) \in AxA \mid a > \sim b,\ costo(b) + V1 > costo(a) > costo(b) + U1, b \in C\ y\ a \notin C\}$ The strong disagreements are composed by the elements which are not consider in weak disagreements, that is considering the relations absolute, strict and indifferent.

Table 1. Preference relations

Notation	Preference
a>>b	*Absolute*
a>b	*Strict*
a>~b	Weak
a~b	Indifference

Table 2. Concordance and veto conditions

Concordance Conditions	Veto Conditions
a>>b	No veto
a>b	Cost(*a*) >> Cost(*b*) + *V1*
a>~b	Cost(*a*) > Cost(*b*) + *V2*
A~b	Cost(*a*) > Cost(*b*) + *U1*

4 Experimental Result

In this section, we describe experiments carried out on the memetic algorithm. The objective of the first experiments is study the performance of memetic algorithm in comparison with exact algorithm, and the second experiment memetic algorithm was compared against a genetic algorithm state of the art proposed in the [10].

4.1 Experimental Environment

The following configuration corresponds to the experimental conditions that are common to the tests described in this work:

1. **Software.** Operating system, Windows 7; programming language Java; compiler,
2. **Hardware.** Computer equipment dual-processor Xeon (TM) CPU 3.06 GHz in parallel and 4 GB RAM.

3. **Instances.** The 5 instances used for this study are randomly generated with twenty projects each; in addition we use other instance reported in [10].
4. **Performance measurement.** Performance is measured through the cardinality of the portfolio, which indicates the maximum number of projects included and the cost of the best portfolio found.

4.2 Algorithm Performance

The purpose of this section is to verify the quality of the solutions obtained by memetic Algorithm, which was implemented to solve the SPP. In our implementation, an SPP instance is formed by three attributes: Id, the budget of projects, and the ranking of the projects.

We generated the experimental instances as follows: we used 5 instances of size 20 that were generated randomly, finding optimal solutions for each of the instances and two instances of size 25 and 40 taken from work [10]. The experimentation with MAs was carried out in a total of 20 times and the optimal portfolio is selected as the best solution found.

For each instance were executed 100 generations as a limit. Table 3 shows the results obtained with the instances Opt_o9p200 until Opt_o9p205, which are the random instances. These results show that the algorithm has good performance since it achieves the number of optimal project portfolio that is 11, which was calculated with an exact algorithm, another important factor is the amount of money required to be applied in the optimal portfolio, observing that the difference of selected projects amounts vary with a small difference.

The table 4 shows the results of the instances Opt_o9p200 until Opt_o9p205 resolved with the exact algorithm, finding the optimal number of projects that can be financed with $ 80,000.00 is 11, show that both algorithms report the same number of projects that satisfy the optimal portfolio.

Continuing the analysis of results is observed that the cost of the best portfolios found is nearly equal. Taking into account the observations mentioned above it is concluded that the MAs that includes: the model of outranking and ranked instances, it is a good option to be applied on the recommendation of optimal portfolios.

Table 3. Experimental Result of the MAs (random instance)

Inst.	Cost Best Portfolio	MEMETIC RESULT Generations = 100; Population=200; Projects= 25; Amount = 1200; V1=30; V2=20; U1=10 Projects																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
20.0	79860	0	1	0	1	1	1	0	0	0	1	1	0	0	0	1	1	0	1	1	1
20.1	79945	0	0	1	0	1	0	1	0	1	0	1	1	0	1	1	0	0	1	1	1
20.2	77920	1	0	1	1	0	1	0	0	0	1	1	1	1	1	0	1	0	1	0	0
20.3	78175	0	1	1	0	0	0	1	1	1	0	1	1	1	0	1	0	1	0	1	0
20.4	79975	1	1	0	1	0	1	0	0	1	1	1	0	1	0	0	1	0	1	0	1

Table 4. Experimental Result of the Exact Algorithm (random instance)

Inst.	Cost Best Portfolio	EXACT ALGORITHM RESULT Projects																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
20.0	79985	0	1	0	1	1	0	1	0	1	0	0	1	1	0	1	0	0	1	1	1
20.1	76795	1	0	1	0	1	0	0	1	1	1	0	1	0	1	0	0	0	1	1	0
20.2	78255	0	0	1	1	0	1	0	1	1	0	0	0	1	1	1	1	0	1	0	1
20.3	79115	0	1	1	1	0	0	1	1	1	0	1	0	1	0	1	1	0	0	1	1
20.4	79520	1	1	1	1	0	0	1	1	0	0	1	1	1	0	1	0	0	0	0	1

Table 5. Comparison of the memetic algorithm, against genetic algorithm to form the optimal portfolio with 25 projects

Memetic Algorithm (a)Vs Genetic Algorithm Result (b) (a) Generations = 100; Population=200; Projects= 25; Amount = 1200; V1=30; V2=20; U1=10 (b) Generations = 200; Population=200; Projects= 25; Amount = 1200; V1=30; V2=20; U1=10																										
ALGORITHM	No. Project Best Portfolio	Cost Best Portfolio	**Projects**																							
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
MAs	17	1180	1	0	1	1	0	0	0	1	0	1	1	1	1	1	1	1	1	1	1	0	1	0	1	1	0
GA	18	1190	1	1	1	1	1	1	1	0	1	0	1	1	1	1	1	1	1	1	1	0	0	0	1	0	0

In the Table 5 show the results of the instances that were used in the work of [10]. Observing the comparison of best results for the two algorithms with the instance of 25 projects. Both algorithms start with a portfolio that includes only 11 projects, and spends a budget of $ 1180. After the execution, the genetic algorithm finds 18 projects in its portfolio optimum, unlike the memetic algorithm that finds 17 projects in its portfolio optimal. In analyzing the two optimal portfolios shows that the genetic algorithm eliminates the two most expensive projects, unlike the memetic algorithm that eliminates five projects not so expensive and optimal portfolio include eleven projects.

In the Table 6 and 7 show the results of the instances that were used in the work of [13]. Observing the comparison of best results for the two algorithms with the instance of 40 projects. Both algorithms start with a portfolio that includes only 24 projects, and spends a budget of $ 4982. After the execution, the genetic algorithm finds 27 projects in its portfolio optimum, unlike the memetic algorithm that finds 26 projects in its portfolio optimal.

Table 6. Comparison of the memetic algorithm, against genetic algorithm to form the optimal portfolio with 40 projects

Memetic Algorithm (a)Vs Genetic Algorithm Result (b) (a) Generations=100; Population=200; Projects=40; Amount=5000; V1=30; V2=20; U1=10 (b) Generations=200; Population=200; Projects=40; Amount=5000;V1=30; V2=20; U1=10																				
ALGORITHM	No. Project Best Port folio	Cost Best Port folio	Projects																	
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Mas	26	4883	0	1	1	0	0	0	1	0	1	0	1	1	1	1	1	1	1	0	1	1
GA	27	4965	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1

Table 7. Comparison of the memetic algorithm, against genetic algorithm to form the optimal portfolio with 40 projects

Memetic Algorithm (a)Vs Genetic Algorithm Result (b) (a) Generations=100; Population=200; Projects=40; Amount=5000; V1=30; V2=20; U1=10 (b) Generations=200; Population=200; Projects=40; Amount=5000;V1=30; V2=20; U1=10																						
ALGORITHM	No. Project Best Portfolio	Cost Best Portfolio	Projects																			
			21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
Mas	26	4883	1	1	0	1	1	1	0	0	1	0	1	0	1	1	1	1	1	0	1	0
GA	27	4965	1	1	0	1	1	1	0	0	0	0	0	0	0	1	0	0	0	1	0	0

5 Conclusions

This article provides a solution to the Social Portfolio Problem by Evolutionary Algorithms, creating a Memetic algorithm that includes the model of outranking and ranked instance through which DM´s preferences are modeled during the search process.

The results presented show that Memetic algorithm has a competitive performance. The solution that follows the ranking information is generally dominated by other solutions which increase the number of projects in the portfolio. The quality of solutions indicates that the algorithm reaches the zone where the best portfolios are located it helps the *DM* to explore the solutions at hand, analyze his preferences and to clarify his decision policies.

In the first experiment the memetic algorithm reaches the optimal number of projects that can be supported. In the second experiment the two algorithms

evaluated 25 projects, the memetic algorithm achieving an average error of 6.66% by comparing the amount of optimal portfolio projects found by the genetic algorithm and the second experiment evaluated 40 project, achieving an average error of 4.7% by comparing the amount of optimal portfolio projects found by the genetic algorithm.

References

[1] Punkka, A., Salo, A.: Rank-Based Sensitivity Analysis of Multiattribute Value Models. Abstract to INFORMS Annual Meeting 2008, Washington, DC, USA (2008)

[2] Carazo, A.F., Gómez, T., Molina, J., Hernández-Díaz, A.G., Guerreo, F.M., Caballero, R.: Solving a comprehensive model for multiobjective project portfolio selection. Computers & Operations Research 37(4), 630–639 (2010)

[3] Castro, M.: Development and implementation of a framework for I&D in public organizations. Master´s thesis. Universidad Autónoma de Nuevo León (2007)

[4] Coello Coello, C.A., Lamont, G.B., Van Veldhuizen, D.A.: Evolutionary Algorithms for Solving Multi-Objective Problems, 2nd edn. Genetic and Evolutionary Computation. Springer (2007)

[5] Deb, K.: Multi-Objective Optimization using Evolutionary Algorithms. John Wiley & Sons, Chichester-New York-Weinheim-Brisbane-Singapore-Toronto (2001)

[6] Durillo, J.J., Nebro, A.J., Coello Coello, C.A., García-Nieto, J., Luna, F., Alba, E.: A study of multi-objective metaheuristics when solving parameter scalable problems. IEEE Transactions on Evolutionary Computation 14(4), 618–635 (2010)

[7] Fernández, E., Navarro, J.: A genetic search for exploiting a fuzzy preference model of portfolio problems with public projects. Annals OR 117, 191–213 (2002)

[8] Fernández, E., López, E., Bernal, S., Coello Coello, C.A., Navarro, J.: Evolutionary multiobjective optimization using an outranking-based dominance generalization. Computers & Operations Research 37(2), 390–395 (2010a)

[9] Fernández, E., López, E., López, F., Coello Coello, C.A.: Increasing selective pressure towards the best compromise in evolutionary multiobjective optimization: The extended NOSGA method. Information Sciences 181(1), 44–56 (2010b)

[10] Fernández, E., Félix, L.F., Mazcorro, G.: Multi-objective optimization of an outranking model for public resources allocation on competing projects. Int. J. Operational Research 5(2), 190–210 (2009)

[11] García, R.: Hyper-Heuristic for solving social portfolio problem. Master´s Thesis, Instituto Tecnológico de Cd. Madero (2010)

[12] Ghasemzadeh, F., Archer, N., Iyogun, P.: A zero-one model for project portfolio selection and scheduling. Journal of the Operational Research Society 50(7), 745–755 (1999)

[13] Greco, S., Mousseau, V., Słowinski, R.: Ordinal regression revisited: multiple criteria ranking using a set of additive value functions. European Journal of Operational Research (2007) doi:10.1016/ j.ejor

[14] Marakas, G.M.: Decision Support Systems in the 21th Century. Prentice Hall, New Jersey (1999)

[15] Michalewicz, Z.: Genetic Algorithms + Data Structures = Evolution Programs. Springuer Series Artificial Intelligence, p. 390. Springer, New York (1996)

[16] Olmedo Pérez, R.A.: Avances en la modelación basada en Relaciones borrosas de Sobre-clasificación para sistemas de Apoyo a la decisión en grupo. Tesis para obtener el grado de Doctor en Ciencias de la Computación (2009)
[17] Osyczka, A.: Multicriteria optimization for engineering design. En Design Optimization, pp. 155–183. Academic Press (1985)
[18] Peñuela, C., Granada, M.: Optimización multiobjetivo usando un algoritmo genético y un operador elitista basado en un ordenamiento no dominado (NSGA-II). Scientia Et Technica 8(35), 175–180 (2007)
[19] Reiter, P.: Metaheuristic Algorithms for Solving Multi-objective/Stochastic Scheduling and Routing Problems. Tesis Doctoral, University of Wien (2010)
[20] Rivera Zárate, G.: Tesis de Doctorado "Solución a Gran Escala del Problema de Cartera de Proyectos Sociales". Instituto Tecnológico de Ciudad Madero (2011)
[21] Roy, B.: The Outranking Approach and the Foundations of ELECTRE methods. In: Reading in Multiple Criteria Decision Aid, pp. 155–183. Spinger (1990)
[22] Roy, B., Slowinski, R.: Handling effects of reinforced preference and counter-veto in credibility of outranking. European Journal of Operational Research 188(1), 185–190 (2008)
[23] Wang, Y., Yang, Y.: Particle swarm optimization with preference order ranking for multi-objective optimization. Information Sciences 179(12), 1944–1959 (2009)

Evolving Bin Packing Heuristic Using Micro-Differential Evolution with Indirect Representation

Marco Aurelio Sotelo-Figueroa[1], Héctor José Puga Soberanes[1], Juan Martín Carpio[1], Héctor J. Fraire Huacuja[2], Laura Cruz Reyes[2], and Jorge Alberto Soria Alcaraz[1]

[1] Instituto Tecnologico de León, México
{masotelof,soajorgea}@gmail.com, pugahector@yahoo.com, mcarpio61@hotmail.com
[2] Instituto Tecnologico de Ciudad Madero, México
{hfraire,lcruzreyes}@prodigy.net.mx

Abstract. The development of low-level heuristics for solving instances of a problem is relatedto the knowledge of an expert. He needs to analyze several components from the problemâ€™s instance and to think out an specialized heuristic for solving the instance. However if any inherent component to the instance gets changes, then the designed heuristic may not work as it used to do it. In this paper it is presented a novel approach to generated low-level heuristics; the proposed approach implements micro-Differential Evolution for evolving an indirect representation of the Bin Packing Problem. It was used the Hard28 instance, which is a well-known and referenced Bin Packing Problem instance. The heuristics obtained by the proposed approach were compared against the well-know First-Fit heuristic, the results of packing that were gotten for each heuristic were analized by the statistic non-parametric test known as Wilcoxon Signed-Rank test.

1 Introduction

Recent years have seen the emergence of a more general type of search algorithm, hyper-heuristics [4] [2], where the search space is a space of heuristics, rather than the space of solutions themselves. This leads to more general solution methods, and we wish to examine the scalability of these solutions in this paper. Hyper-heuristics are heuristics which choose “between a set of low-level heuristics, using some learning mechanism”[14]. i.e. a hyper-heuristic manages a fixed set of heuristics.

One of the motivations of hyper-heuristics [4] [2] is to “raise the level of generality at which optimisation systems can operate”, providing a search methodology which will deliver solutions which are “good-enough soon-enough

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 349–359.
springerlink.com

cheap-enough"[2]. While this "off the peg" approach is unlikely to produce solutions which are as good at those produced by a "tailormade" problem specific method, the goal is to underpin decision support systems which can be applied to a broader range of problems than is possible today.

Hyper-heuristics can also be used to produce new heuristics, which are not very dependent upon the problem at hand. In this case was use the μ DE has hyper-heuristics to try to obtain heuristics for the Bin Packing Problem.

There are many heuristics for the bin packing problem which are obtained by an experts through them experiences on this problems. The principal problem is that this heuristics are developed for a given for a kind of bin packing instances.

In this paper, we show that the evolved heuristics can be apply to the bin packing instances.

2 Bin Packing Problem

The classical one dimensional bin packing problem (BPP) [7] consists of a set of pieces, which must be packed into as few bins as possible. Each piece$j$ has a weight w_j, and each bin has capacity$c$. The objective is to minimise the number of bins used, where each piece is assigned to one bin only, and the weight of the pieces in each bin does not exceed$c$. This NP-complete decision problem naturally gives rise to the associated NP-hard optimization problem.

A mathematical definition of Bin Packing Problem [7] [10] is:

Minimize:

$$z = \sum_{i=1}^{n} y_i \tag{1}$$

Subject to:

$$\sum_{j=1}^{n} w_j x_{ij} \leq c y_i \quad i \in N = \{1, \ldots, n\} \tag{2}$$

$$\sum_{i=1}^{n} x_{ij} = 1 \quad j \in N \tag{3}$$

$$y_i \in \{0,1\} \quad i \in N \tag{4}$$

$$x_{ij} \in \{0,1\} \quad i \in N, j \in N \tag{5}$$

where:

- w_j: weight of item j.
- y_i: binary variable that shows if the bin i have items.

- x_{ij}: indicates wheter the item$j$ is in the bin i.
- n: number of avalible bins (also the number of items$n$).
- c: capacity of each bin.

There are many set tests instances like the Hard28 [13]. It is considered like the most difficult instances which could not be solved neither by the pure cutting plane algorithm from [1] nor by many reduction methods.

In [6] was propose an objective function which puts a premium on bins that are filled completely or nearly so. Importantly, the fitness function is designed to avoid the problem of plateaus in the search space, that occur when the fitness function does not discriminate between heuristics that use the same number of bins.

$$Fitness = 1 - \left[\frac{\sum_{i=1}^{n} \left(\frac{\sum_{j=1}^{m} w_j x_{ij}}{c} \right)^2}{n} \right] \tag{6}$$

where:

- n: number of bins.
- m : number of pieces.
- w_j: weight of piece j-th.
- $x_{ij} = \begin{cases} 1 \; if \; piece \; j \; es \; in \; bin \; i \\ 0 \; otherwise \end{cases}$
- c: bin capacity.

To solve the Bin Packing Problem are many heuristics like the First-fit [5]. It is a constructive heuristic which packs a set of pieces one at a time, in the order that they are presented. The heuristic iterates through the open bins, and the current piece is placed in the first bin into which it fits. This heuristic is for the online bin packing problem, because it packs pieces one at a time, and a piece cannot be moved once it has been assigned to a bin. The First-Fit algorithm can be seen in the algorithm 1.

Algorithm 1. First-Fit Algorithm

Require: L input list of pice to be packed.
1: Inicialise s with a empty bin.
2: **for all** pieces p in L **do**
3: **for all** bins b in s **do**
4: **if** $Capacity(b) \leq FreeSpace(b) + Size(p)$ **then**
5: Put the pice b in the bin b.
6: **end if**
7: **end for**
8: **if** p was not place in any bin **then**
9: Add a new bin b to s and add the pice p to the bin b.
10: **end if**
11: **end for**

3 Micro-differential Evolution

Differential Evolution (DE) [15] was developed by R. Storn and K. Price in 1996. It is a vector-based evolutionary algorithm, and it can be considered as a further development to genetic algorithm. It is a stochastic search algorithm with self-organizing tendency and it does not use the information of derivatives [16].

Like in genetic algorithms [8], the design parameters in a d-dimensional search space are represented as vectors, and various genetic operators are operated over their bit of strings. However, unlikely genetic algorithms, differential evolution carries out operations over each component (each dimension of solution).

For a $d-$dimensionaloptimization problem with $d$parameters, a population of n solution vectors are initially generated, we have $x_i$where $i = 1,2, \dots, n$. For each solution $x_i$at any generation$t$, we use the convetional notation as:

$$x_i^t = (x_1^t, x_2^t, \dots, x_d^t) \tag{7}$$

which consist of $d-$components in the $d-$dimensional space. This vector can be considered as the chromosomes or genomes.

This metaheuristics consists of three main steps: mutations, crossover and selection.

Mutation is carried out by the mutation scheme. For each vectorx_i at any time or generation$t$, first randomly choose three distinct vector x_p, x_q and x_r at$t$, and then generate a so-called donor vector by the mutation scheme.

$$v_i^{t+1} = x_p^t + F\left(x_p^t - x_r^t\right) \tag{8}$$

where$F \in [0,2]$ is a parameter, often refered to as the differential weight. This requires that the minimum number of population size is$n \geq 4$. We can see that the perturbation$\delta = F[x_q - x_r]$ to the vector x_p is used to generate a donor vectorv_i, and such perturbation is directed and self-organized.

The crossover is controlled by a probability$\mathrm{C_r} \in [0,1]$and actual crossover can be carried out in two ways: binomial and exponential. The binomial scheme performs crossover on each of thed componets or variables/parameters. By generating a uniformly distributed random number$\mathrm{r_i} \in [0,1]$, the jth componets of $\mathrm{v_i}$ is manipulated as:

$$u_{j,i}^{t+1} = \begin{cases} v_{j,i} \; if \; r_i \leq C_r \\ x_{j,i} \; otherwise \end{cases} \tag{9}$$

This way, each component can be decided randomly whether to exchange with donor vector or not.

In the exponential scheme, a segment of the donor vector is selected and this segment starts with a random$k$ with a random lenght L which can include many components. Mathematically, this is to choose$\mathrm{k} \in [0, \mathrm{d}-1]$ and $\mathrm{L} \in [0, \mathrm{d}]$ randomly, and we have:

$$u_{j,i}^{t+1} = \begin{cases} v_{j,i}^{t} \ \ for\ j = k, \dots, k - L \in [1, d] \\ \qquad x_{j,i}^{t} otherwise \end{cases} \tag{10}$$

As the binomial is simpler to implement we will use the binomial crossover in our implemention.

Selection is essentially the same as that used in genetic algorithms. It is to select the most fittest, and for minimization problem, the minimum objective value. Therefore, we have:

$$x_i^{t+1} = \begin{cases} u_{j,i}^{t+1} \ \ if\ f(u_i^{t+1}) \le f(x_i^{t}) \\ \qquad x_i^{t}\ otherwise \end{cases} \tag{11}$$

All the above three components can be seen in the pseudo code as shown in algorithm 2. It is worth pointing out there that the use of$J$ is to ensure that $v_i^{t+1} \neq x_i^t$, which may increase the evolutionary or exploratory efficiency. The overall search efficiency is controlled by two parameters: the differential weight *F* and the crossover probabilityC_r.

Algorithm 2. Differential Evolution Algorithm

Require: F differential weight, C_r crossover probability, n population size

1: Initializate the initial population.
2: **while** stopping criterion not met **do**
3: **for** $i = 1$ to n **do**
4: For each x_i randomly choose 3 distinct vector x_p, x_r and x_r.
5: Generate a new vector v by DE scheme 8.
6: Generate a random index $J_r \in \{1, 2, \dots, d\}$ by permutation.
7: Generate a randomly distributed number $r_i \in [0, 0]$
8: **for** $j = 1$ to n **do**
9: For each parameter $v_{j,i}$ (jth component of v_i), update
10: $u_{j,i}^{t+1} = \begin{cases} v_{j,i}^{t+1} & \text{if } r_i \le C_r \text{ or } j = J_r \\ x_{j,i}^{t} & \text{if } r_i > C_r \text{ or } j \neq J_r \end{cases}$
11: **end for**
12: Select and update the solution by 11.
13: **end for**
14: Update the counters such as $t = t + 1$
15: **end while**

In [9] can see 5 strategies to mutate the v vector:

- $DE/rand/1 : v_{\mathrm{i}} = \mathrm{x}_{r1} + \mathrm{F}(x_{r2} - x_{r3})$.
- $DE/best/1 : v_{\mathrm{i}} = \mathrm{x}_{best} + \mathrm{F}(x_{r1} - x_{r2})$.
- $DE/current\ to\ best/1 : v_i = x_i + F(x_{best} - x_i) + \mathrm{F}(x_{r1} - x_{r2})$.
- $DE/best/2 : v_i = x_{best} + F(x_{r1} - x_{r2}) + \mathrm{F}(x_{r3} - x_{r4})$.
- $DE/rand/2 : v_i = x_{r1} + F(x_{r2} - x_{r3}) + \mathrm{F}(x_{r4} - x_{r5})$.

where: r_1, r_2, r_3, r_4, r_5 are random and mutually different integer,witch should also be different from the trial vector's index$i$ and x_{best} is the individual vector with best fitness.

Micro-Differential Evolution (μDE) [11] has the same structure and operations with standard DE. The only difference is the population size, which is typically very small. Thus, although it is recommended to use populations of size up to $N = 10n$, where n is the problem dimension, μDE uses the smallest possible number of individuals. Taking into consideration the restriction, $N \geq 5$, that permits the application of all mutation operators, a population size, $N = 6$, can be considered a reasonable choice for μDE.

Moreover, μDE is expected to converge rapidly due to the small number of individuals. Usually, the ratio, n/N, is indicative of the difficulty met by an algorithm on a given problem. Small values of this ratio (less than 1) correspond to population with size larger than its dimension. On the other hand, values higher than 1 correspond to problem dimension higher than population size. Empirical evidence suggest that in most cases the higher the ratio is, the harder the problem becomes for the algorithm. Therefore, μDE can be considered as a promising approach in rather low-dimensional problems. The changes made in the μDE can be seen in the pseudo code as shown in algorithm 3.

Algorithm 3. Micro Differential Evolution Algorithm

Require: F differential weight, C_r crossover probability, n population size, $Restart$ number of fuction call to restart the population, e number of population of elitism.

Initializate the initial population.
Set $Cont = 0$
while stopping criterion not met **do**
  if $Cont\%Restart = 0$ **then**
    Sort population and choose the best e.
    Generate randomly the new $n - e$ elements.
  end if
  for $i = 1$ to n **do**
    For each x_i randomly choose 3 distinct vector x_p, x_r and x_r.
    Generate a new vector v by DE scheme 8.
    Generate a random index $J_r \in \{1, 2, \ldots, d\}$ by permutation.
    Generate a randomly distributed number $r_i \in [0, 0]$
    for $j = 1$ to n **do**
      For each parameter $v_{j,i}$ (jth component of v_i), update
      $u_{j,i}^{t+1} = \begin{cases} v_{j,i}^{t+1} & \text{if } r_i \leq C_r \text{ or } j = J_r \\ x_{j,i}^{t} & \text{if } r_i > C_r \text{ or } j \neq J_r \end{cases}$
    end for
    Select and update the solution by 11.
    Set $Cont = Cont + 1$.
  end for
  Update the counters such as $t = t + 1$
end while

3.1 Indirect Representation

The indirect representation used by the μDE is based on the Grammatical Evolution [12]. The Grammatical Evolution is based on the Backus Naur Form (BNF), which is a notation for expressing the grammar of a languaje in the form of production rules. BNF grammars consist of terminals, which are items that can appear in the laguage, and non-terminals, which can be expanded into one or more terminals and non-terminals. A grammar can be represented by the 4th-tuple, $\{N, T, P, S\}$, where N is the set of non-terminals, $T$is the set of terminals, $P$is the set of production rules that maps the elements of N to$T$, and S is a start symbol which is a member of$N$.

The Grammatical Evolution was used as inderect representation in the μDE. Each vector is a grammar constructed using each elements from the vector to build the grammar followed BNF.

4 Experiments

The proposed approach tries to find an heuristic, which is developed through the evolution of the grammar (an indirect representation of the problem) by μDE. The grammar used is:

<begin> ::= (<expr>) <= (<expr>)
<expr> ::= (<expr><op><expr>) | <var> | abs(<expr2>)
<expr2> ::= (<expr2><op><expr2>) | <var>
<var> ::= F | C | S <op> ::= + | * | - | /

where:

- S: size of the current piece.
- C: bin capacity.
- F: sum of the pieces already in the bin.

After an heuristic is generated by evolving the grammar, it is applied to all test instance from Hard28. Finally is calculated the fitness for each instance by using the objective function (equation 6).

The table 1 shows the parameters used for μDE. They were choosen by empirical evidence. The cores number of the machine used for the experiments served to establish the size population, it was 8. The number of function calls was only the 40% of the total number reported in [3], this number showed good results.

It was expected that both heuristics (the First-Fit and the evolved heuristic) would have the same performance over the hard28 instances.

Table 1. Initialisation parameters of each μDErun

Parametre	Value
Strategie	DE/rand/1
Functions Call	20000
Population Size	8
F	0.9
C_r	0.8
Restart	100
Elitism	3

5 Results

The table 2 shows the heuristics generated by the μDE. These heuristics are the results of the evolution of the grammar for each hard28's instance. The heuristics are applied over the BPP as follow: As first step, it must be analized if the chain that represents the generated heuristic, contains only terminal components. The next step is to apply the heuristic, if the current item fits in the first bin, then the item is put into that bin, otherwise the heuristic is applied until it is found a bin where the current item fits.

Table 2. Heuristics obtained for each instance for the Hard28

Instance	Envolved Heuristic
BPP 13	((S - (F - (F + F)))) ≤ (abs(C))
BPP 14	((F + S)) ≤ (C)
BPP 40	((abs(F) + S)) ≤ (abs(C))
BPP 47	(F) ≤ (abs((C - S)))
BPP 60	(abs((F + S))) ≤ (abs((S - (S - C))))
BPP 119	((abs(S) + F)) ≤ (C)
BPP 144	(abs((S + (S - S)))) ≤ (abs((F - C)))
BPP 175	(S) ≤ ((C - F))
BPP 178	(F) ≤ (abs((S - C)))
BPP 181	(abs(F)) ≤ (abs((((C / F) * F) - S)))
BPP 195	(S) ≤ (abs((C - (C / (C / F)))))
BPP 359	((F + S)) ≤ (C)
BPP 360	(abs(S)) ≤ (abs((C - F)))
BPP 419	(F) ≤ (abs((S - C)))
BPP 485	(abs(S)) ≤ (abs((C - F)))
BPP 531	((S + abs(F))) ≤ (C)
BPP 561	(F) ≤ (abs((C - S)))
BPP 640	(abs((F + S))) ≤ (C)
BPP 645	(S) ≤ ((C - abs(F)))

Table 2. (*continued*)

BPP 709	((F + abs(S))) ≤ (C)
BPP 716	(F) ≤ ((C - abs(S)))
BPP 742	(abs((F + S))) ≤ (C)
BPP 766	(S) ≤ ((abs(C) - F))
BPP 781	(F) ≤ ((abs(C) - abs(S)))
BPP 785	(F) ≤ (abs((C - S)))
BPP 814	(S) ≤ (((abs((C + F)) - F) - F))
BPP 832	(F) ≤ ((C - S))
BPP 900	(abs(S)) ≤ ((abs(C) - abs(F)))

The table 3 shows the results after packing and evaluating the instances with the objective function. It shows the results of each instance using the First Fit algorithm and the evolved Heuristic. it was applied the Wilcoxon Signed-Rank Test, to discern which heuristic have the best performance based on the table 3, and it could not discern if there is a difference.

Table 3. Results obtained from the set Hard28

Instance	First Fit	Evolved Heuristic
BPP 13	0.023376	0.023376
BPP 14	0.025691	0.025691
BPP 40	0.025589	0.025589
BPP 47	0.024322	0.024322
BPP 60	0.028981	0.028981
BPP 119	0.023894	0.023894
BPP 144	0.022027	0.022027
BPP 175	0.021079	0.021079
BPP 178	0.022417	0.022417
BPP 181	0.022276	0.022279
BPP 195	0.021526	0.021526
BPP 359	0.024243	0.024243
BPP 360	0.026152	0.026152
BPP 419	0.019084	0.019084
BPP 485	0.023592	0.023592
BPP 531	0.022172	0.022172
BPP 561	0.022407	0.022407
BPP 640	0.023383	0.023383
BPP 645	0.023268	0.023268
BPP 709	0.022516	0.022516
BPP 716	0.023494	0.023494
BPP 742	0.023274	0.023274
BPP 766	0.024135	0.024135
BPP 781	0.019128	0.019128
BPP 785	0.021841	0.021841
BPP 814	0.022449	0.022449
BPP 832	0.031609	0.031609
BPP 900	0.021422	0.021422

6 Conclusions and Future Work

It has been shown that human designed heuristics like firstfit can be easily obtained by an indirect representation along to μDE. The heuristics were evolved without any apriori knowledge. It was only used a grammar to represent the problem.

For all cases were obtained heuristics that can be applied to a BPP instance. The objective function used is capable to discern in a better way since it measures the available space of the bins instead the number of bins.

A protential area for future work is to generate only one heuristic for all the instances instead of one for each instance.

Acknowledgement. Authors thanks the support received from Consejo Nacional de Ciencia y Tecnologia (CONACyT).

References

1. Belov, G., Scheithauer, G.: A Cutting Plane Algorithm for the One-Dimensional Cutting Stock Problem with Multiple Stock Lengths. European Journal of Operational Research 141, 274–294 (2002)
2. Burke, E.K., Hart, E., Kendall, G., Newall, J., Ross, P., Schulenbur, S.: Hyperheuristics: An emerging direction in modern search technology. In: Glover, F., Kochenberger, G. (eds.) Handbook of Meta-Heuristics, pp. 457–474. Kluwer (2003)
3. Burke, E.K., Hyde, M.R., Kendall, G.: Evolving Bin Packing Heuristics with Genetic Programming. In: Runarsson, T.P., Beyer, H.-G., Burke, E.K., Merelo-Guervós, J.J., Whitley, L.D., Yao, X. (eds.) PPSN 2006. LNCS, vol. 4193, pp. 860–869. Springer, Heidelberg (2006)
4. Burke, E.K., Kendall, G.: Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques. Springer (2006)
5. Coffman Jr., E.G., Johnson, D.S., Mcgeoch, L.A., Weber, R.R.: Bin Packing with Discrete Item Sizes Part II: Average-Case Behavior of FFD and BFD 13, 384–402 (1997) (in preparation)
6. Falkenauer, E., Delchambre, A.: A genetic algorithm for bin packing and line balancing. In: Proceedings of IEEE International Conference on Robotics and Automation, vol. 2, pp. 1186–1192 (1992)
7. Garey, M.R., Johnson, D.S.: " Strong " NP-Completeness Results: Motivation, Examples, and Implications. J. ACM 25, 499–508 (1978)
8. Holland, J.H.: Adaptation in Natural and Artificial Systems. University of Michigan Press (1975)
9. Luke, S.: Essentials of Metaheuristics. Lulu (2009)
10. Martello, S., Toth, P.: Knapsack Problems, Algorithms and and Computer Implementations. John Wiley & Sons Ltd., New York (1990)
11. Parsopoulos, K.E.: Cooperative micro-differential evolution for high-dimensional problems. In: Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation in GECCO 2009, pp. 531–538. ACM, New York (2009)

12. Ryan, C., Collins, J.J., Neill, M.O.: Grammatical Evolution: Evolving Programs for an Arbitrary Language. In: Banzhaf, W., Poli, R., Schoenauer, M., Fogarty, T.C. (eds.) EuroGP 1998. LNCS, vol. 1391, pp. 83–95. Springer, Heidelberg (1998)
13. Schoenfield, J.E.: Fast, exact solution of open bin packing problems without linear programming. PhD thesis. US Army Space and Missile Defense Command, Huntsville, Alabama, USA (2002)
14. Soubeiga, E.: Development and application of hyperheuristics to personnel scheduling. PhD thesis, University of Nottingham (2003)
15. Storn, R., Price, K.: Differential Evolution - A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. J. of Global Optimization 11, 341–359 (1997)
16. Yang, X.S.: Nature Inspired Metaheuristic Algorithms, 2nd edn. Luniver Press (2008)

Improving the Performance of Heuristic Algorithms Based on Exploratory Data Analysis

Marcela Quiroz C.[1], Laura Cruz-Reyes[1], José Torres-Jiménez[2], Claudia G. Gómez S.[1], Héctor J. Fraire H.[1], and Patricia Melin[3]

[1] Instituto Tecnológico de Ciudad Madero, México
qc.marcela@gmail.com, {lcruzr,hfraire}@prodigy.net.mx, cggs71@hotmail.com
[2] CINVESTAV-TAMAULIPAS, México
jtj@cinvestav.mx
[3] Tijuana Institute of Technology, México
pmelin@tectijuana.mx

Abstract. This paper promotes the application of empirical techniques of analysis within computer science in order to construct models that explain the performance of heuristic algorithms for NP-hard problems. We show the application of an experimental approach that combines exploratory data analysis and causal inference with the goal of explaining the algorithmic optimization process. The knowledge gained about problem structure, the heuristic algorithm behavior and the relations among the characteristics that define them, can be used to: a) classify instances of the problem by degree of difficulty, b) explain the performance of the algorithm for different instances c) predict the performance of the algorithm for a new instance, and d) develop new strategies of solution. As a case study we present an analysis of a state of the art genetic algorithm for the Bin Packing Problem (BPP), explaining its behavior and correcting its effectiveness of 84.89% to 95.44%.

1 Introduction

Most optimization problems in the real world belong to a special class of problems called NP-hard, which means that there are no known efficient algorithms to find the optimal solution in the worst case. To solve these problems, the efforts of many researchers have resulted in a variety of heuristic algorithms that have shown satisfactory performance. However, to date there is no algorithm that is best for all possible situations. One of the main challenges of behavioral analysis of heuristic algorithms is to identify what strategies make an algorithm to show an improved performance and under what conditions they get it.

In this paper we formulate an experimental approach for a comprehensive study of the optimization process in order to identify inherent relationships among the factors that affect the algorithmic performance. The proposed approach combines methods of exploratory data analysis and causal inference in three stages. In the

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 361–375.
springerlink.com

characterization phase factors that influence performance are identified and quantified by indexes. In the characteristics refining stage incorrect and redundant indexes are discarded. In the study of relations stage an analysis of the characteristics of the optimization process is made in order to obtain performance relations that eventually will become algorithm behavior explanations.

To evaluate the contribution of the proposed approach we performed a study of the optimization process for the Bin Packing Problem (BPP). BPP is considered NP-hard [1, 2] and consists in packing a set of n items of different sizes $W=\{w_1,\ldots,w_n\}$ in the minor number of fixed size bins without violating the capacity c of any bin. BPP has an extensive number of industrial and logistic applications and frequently happens as a sub-problem in several practical problems [3, 13]. The case study confirms the importance of applying exploratory data analysis as a guide for understanding the performance of algorithms.

2 Performance Analysis of Heuristics Algorithms

The so-called NP-hard problems are of great interest in computer science. One feature of these problems is that the exact algorithms used to solve require an exponential amount of time in the worst case. In other words, these problems are very difficult to solve [1]. In these conditions it is necessary to use heuristic algorithms that provide approximate solutions in a reasonable time, but do not guarantee the optimal solution. Heuristic algorithms are procedures that used strategies based on common sense in order to obtain high quality solutions (not necessarily optimal) to complex problems efficiently.

The criteria for measuring the performance of heuristic algorithms depend on the methods chosen for characterization, which can be theoretical or experimental. In the first, for each algorithm, mathematical analysis are used to determine the amount of resources required as a function of the size of the better, worse or average case. The latter is based on experimentation for the characterization and, unlike the theoretical methods, allows describing the behavior of specific cases.

The theoretical study of heuristics algorithms performance is unusual, because the randomness in some of these algorithms and the complexity of the optimization problems that impede a proper mathematical analysis. Moreover, the applicability of the theoretical results is very limited, because these are obtained based on idealized conditions that do not occur in practical situations [4, 5].

Despite the importance of the experimental analysis of performance, this has not been sufficiently exploited in the study of heuristics algorithms. One of the reasons which make difficult this study is that, generally, the algorithms are considered as black boxes whose inner workings are unknown.

Recent works propose tools that can be taken into account to assist the analysis of the factors that influence the algorithmic performance and thus the explanation of performance [4, 6, 7, 8, 9]. In general, tools for data analysis (factors) proposed in these works can be grouped into two categories: exploratory data analysis techniques and confirmatory data analysis techniques.

In the exploratory data analysis (EDA) the aim is to obtain knowledge of the data set and its underlying structure. Includes statistical methods, tabular

comparisons, graphical analysis, causal inference and multivariate analysis that make possible the construction of a model that describes the set of relations of the factors under study [10, 11, 12].

Confirmatory data analysis (better known as statistical hypothesis testing), begins with assumptions (models) about functional relationships between variables (factors) of the data. Includes estimations of parameters of the models and statistical hypothesis testing to complement and validate proposed models [8, 11, 12].

The identification of appropriate data analysis techniques for the types of problems and opportunities that occur when analyzing the algorithmic performance is a research area still in development. The selection of tools to use depends on the characteristics of the data under study (problem and algorithm). For example, graphical methods of exploratory data analysis are most appropriate to analyze trends in large multivariate data sets. Also, methods of confirmatory data analysis can be problematic, depending on how much knowledge exists about the function (model), therefore, methods of data analysis, with exploratory bases, that do not start assuming functional relationships between the factors studied, are most appropriate when the objective is to discover relationships and evaluate various models [4].

3 Characterization of the Optimization Process

The characterization of the problem and the solution process is an essential part in the performance analysis of algorithms, and allows identifying the factors that influence the algorithmic behavior. A quality performance analysis requires the definition of appropriate indexes to quantify the features that impact final performance.

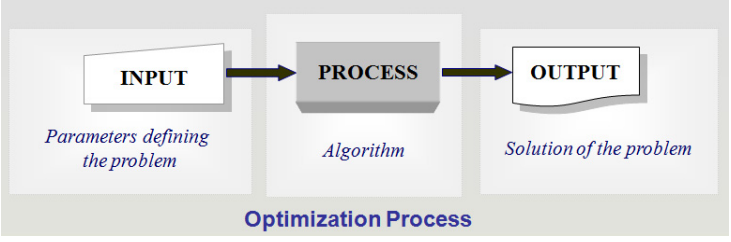


Fig. 1. The optimization process of a problem

To gain insight into the performance of a heuristic algorithm on an optimization problem, it is necessary to make a comprehensive study of the entire solution process. The optimization process can be understood as the act of solving an optimization problem (input) using an algorithm (process), obtaining a final solution (output). This process is illustrated in Figure 1.

The input consists of an instance of the optimization problem to solve, composed of a set of specific parameters that define it. The process includes the set of strategies used to solve the problem, like the functions and parameters used by the

algorithm. The output provides the solution of the problem and some important measures of performance, like number of iterations and execution time.

4 An Experimental Approach to Study the Optimization Process

Figure 2 shows the proposed approach for the experimental analysis of heuristics algorithms in optimization problems. The main objective of the *characterization* stage is to identify, in each phase of the optimization process, relevant and measurement feasible performance factors. These factors are characterized through characterization functions (indexes) that provide useful information to describe the algorithmic performance. In the second stage, *characteristics refining*, the indexes defined in the characterization stage are analyzed using exploratory techniques in order to rule out incorrect, redundant or irrelevant indexes. If necessary, new indexes are defined using multivariate analysis techniques.

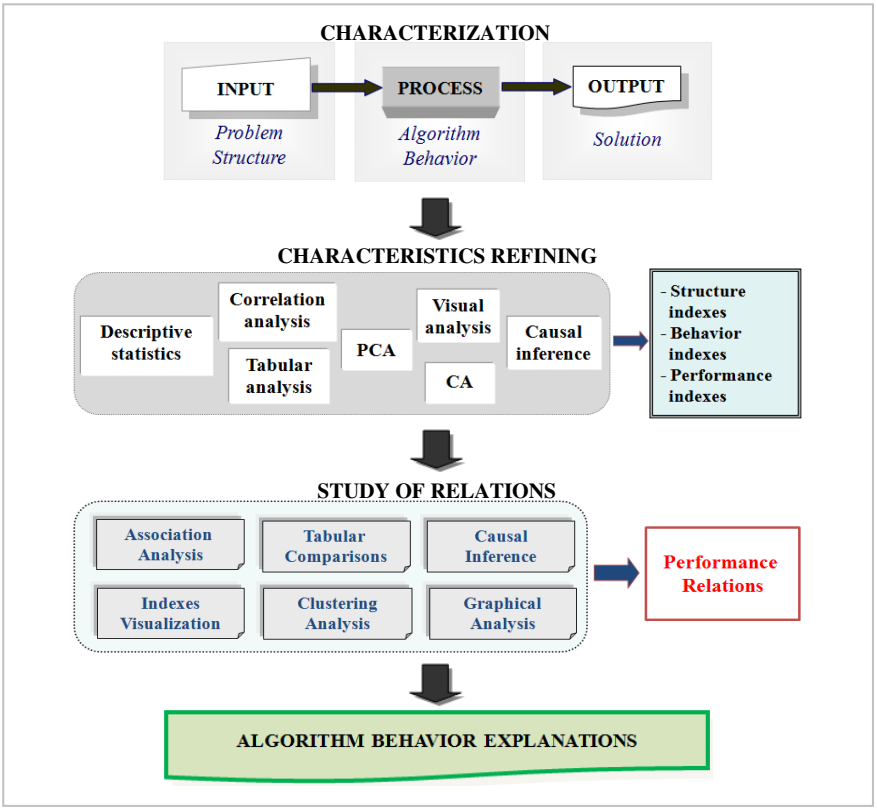


Fig. 2. Experimental approach to study the performance of heuristic algorithms

In the third stage, *study of relations*, an exploratory analysis of the characteristics of the optimization process is done in order to obtain performance relations that explain the behavior of the heuristic algorithm under study. For this purpose we use multivariate statistical methods, tabular and graphical analysis, data visualization techniques and causal analysis. The knowledge gained as a result of the study of relations allows understanding the behavior of the heuristic algorithm, explaining how its final performance is affected by several factors that cause it, visualizing possible improvements in its structure.

5 Case Study: Bin Packing Optimization Process

In this section we present the application of the proposed experimental approach to the characterization of the bin packing problem (BPP) by analyzing the performance of a state of the art genetic algorithm named HGGA-BP [13]. We introduce a new set of indexes for BPP and the algorithm behavior characterization; in addition, we show the redesign of HGGA-BP algorithm structure and present new experimental results.

5.1 Phase 1: Characterization

This stage is carried out to characterize the optimization problem, the behavior of the algorithm of interest and the final performance. Having identified the factors that influence the optimization process, it is analyzed which aspects can be measured in each category, and indexes that characterize these factors are defined.

Bin Packing Characterization

Characterize the structure of an instance of BPP is a key to predict the behavior that will have a heuristic algorithm at the time of the solution. It is known that factors like the number of items, the central tendency of the weights and their distribution, impact the degree of difficulty that an instance can have on a solution algorithm. The challenge is to formulate indicators that quantify these factors.

We carried out the compilation of 1668 benchmark instances recognized by the scientific community. These test instances were taken from Internet sites [14, 15, 16, 17]. Descriptive information for each instance of the problem is characterized by indexes which use information from the parameters of the problem in that instance. Different authors have proposed set of indexes for the characterization of BPP [18, 19].

With the goal of modeling the structure of a instance, Pérez and Cruz [18] formulated a specific-purpose indexes set, formed by five difficulty indexes: instance size (p), occupied capacity (t), dispersion (d), factors (f), and bin usage (b). In other work, Álvarez [19] proposed 21 indexes based on descriptive statistics, these indexes characterize the weight distribution of the BPP items and can be grouped into four categories of statistical measures: centralization, dispersion, position and form of the frequency distribution of the weights. We studied all these indexes by

analyzing if they allowed discriminating between different BPP instances. To assist in this study and contribute with the characterization of BPP, each instance was plotted showing the distribution of the weights of items in relation to the bin capacity.

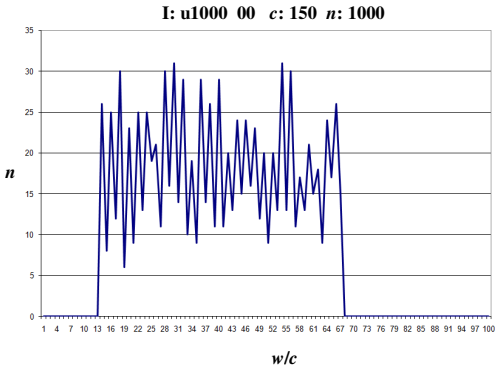


Fig. 3. Weight distribution graph for a BPP instance

Figure 3 shows the distribution plot of a BPP instance. At the top of the plot one can see the name of the instance (I: u1000_00), the bin capacity (*c*: 150) and the number of items (*n*: 1000). The horizontal axis represents the weight of the items as a percentage of the bin capacity ($0 < w_i/c \leq 1$). The vertical axis counts the number of items in each weights percentage. As it can be observed, the weights of the items are uniformly distributed between 13% and 67% of the bin capacity. The chart also shows that the number of items with each weight varies between 6 and 31.

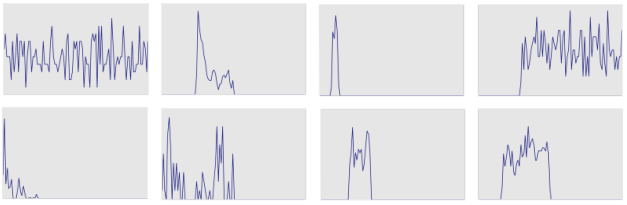


Fig. 4. Different weight distribution graphs for a BPP instances

Figure 4 shows representative plots of the set of instances. It can be observed the variety of forms of the frequency distribution of the weights of the items, as well as the ranges of weights, suggesting the importance of a correct characterization of these factors. The analysis of the distribution plots allows identifying differences between different classes of instances and helps to discover characterization indexes.

The study of the weight distribution graphs revealed that the indexes proposed in previous works for BPP characterization are able describe much of the structure of an instance of BPP. However it seems that there are certain aspects that were not taken into account by the authors, factors that are important to point out differences between instances. In this work we propose five new indexes of characterization. These new indexes can: locate the distribution range of the set of weights, identify important trends in the weights of items, measure the frequency of repetition of the weights, and contribute to the characterization of the form of the distribution of the weights. These indexes are defined below.

Lower_Weight and *Upper_Weight* indexes, respectively, represent the weight of the smallest and the biggest item, in relation to the bin capacity, and allow locating the beginning and the end of the weights distribution.

Multiplicity characterizes the average number of repetitions of each weight and helps to identify trends or peaks in the weights frequency distribution. Equation 1 defines this index, m_i is the number of items with weight $s_i \in S$, where S include the $\hat{n}$ different weights, without repetitions ($1 \leq \hat{n} \leq n$).

$$Multiplicity = \frac{\sum_{i=1}^{\hat{n}} m_i}{c} \tag{1}$$

Max_Repe index represents the maximum frequency of repetition of a weight in the set of items. Comparing this measure values with those of *Multiplicity* index, may suggest the existence of points of the weights frequency distribution having a greater accumulation items.

Uniformity measures the degree of uniformity of the weights distribution, from the division of the range of the weights into four segments of equal magnitude and the differences between the expected number and the actual number of items in each subrange. Given: n (number of items), $W = \{w_i : w_i < w_{i+1}\}$, $1 \leq \forall\, i \leq n$ (array of weights in increasing order), $R = \max(W) - \min(W)$ (range of the weights), $r_i = \min(W) + iR/4$, $0 \leq i \leq 4$ (division of subranges). The set of items contained in each subrange j is: $B_j = \{w \in W : r_{j-1} < w \leq r_j\}$, $1 \leq j \leq 4$. Equation 2 measures the degree of uniformity of distribution of the set of weights. A value of *Uniformity* close to one represents a uniform distribution.

$$Uniformity = 1 - \frac{\sum_{j=1}^{4} \left| \left(\frac{n}{4} - |B_j| \right) \right|}{n} \tag{2}$$

Algorithm Characterization

The behavior of the algorithm was measured by three indexes: a) *New_individuals* which is the average number of new individuals added to the population in each generation; b) *Deviation_Fitness* characterizes the average deviation in the fitness of the solutions of the population; c) *Deviation_Best* represents the deviation in the fitness of the best solutions of each generation [13].

Final Performance Characterization

The final performance of the algorithm was measured by means of three indexes: a) *Theoretical_Ratio*, which is the ratio between, the number of bins of the obtained solution, and the number of bins of the optimal solution; b) *Best_Fitness*, the fitness of the best solution obtained, for an instance of BPP, in *r* runs of the algorithm; c) *Generation*, represents the generation in which the algorithm finds the best solution for an instance.

5.2 Phase 2: Characteristics Refining

After an initial review of BPP indexes, it appears that several features of the structure of the problem (BPP) are characterized by a large number of indexes, so it is possible that some of them are redundant or unnecessary. The exploratory analysis of the measures helps to verify this fact. The correlation matrix, revealed the existence of association between certain variables that characterize the centralization, dispersion and form of the weights distribution. If there are lineal associations between pair of variables it is possible that statistical analysis based in the correlation matrix does not make sense or produce incorrect results. To avoid this, some indexes were discarded to eliminate redundancy and possible problems in future analyzes. Overall, it was decided to keep indexes requiring fewer calculations.

Having identified the new set of indexes, it is necessary to analyze the contribution and consistency of each measure. The aim of this study is to identify and eliminate those indexes that do not provide meaningful information for the characterization of the structure of the BPP instances. Inconsistent or incorrect indexes only add wrong information to the description of an instance and do not allow discriminating between instances of different nature.

An example of inconsistent indexes can be seen in Figure 5 that shows two scatter plots for characterization indexes of central tendency (*Mode* and *t*),

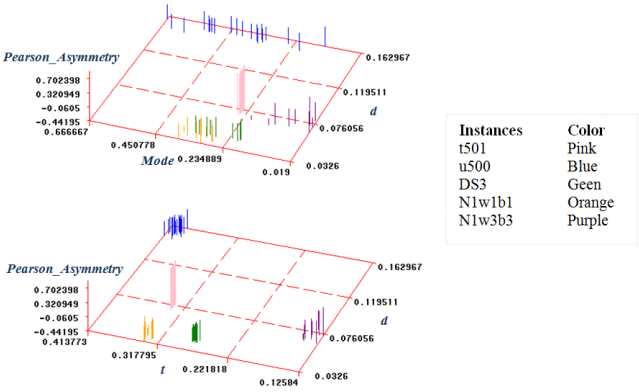


Fig. 5. Scatter plots for *Mode*, *t*, *d* and *Pearson_Asymmetry* for BPP instances

dispersion (*d*) and form (*Pearson_Asymmetry*). These indexes were applied to the characterization of five sets of instances with different structures. In the first plot, we use the index *Mode* proposed by Álvarez [19] as a measure of central tendency of the set of weights of items and it can be seen that this measure cannot discriminate between different instances. In the second plot, substituting *Mode* for *t* proposed by Pérez and Cruz [18], it can be clearly seen the importance of the information provided by the latter, by interacting with *d* and *Pearson_Asymmetry*; the different instances sets are better identified in groups.

The exploratory data analysis suggested that indexes *Mode*, *Quartiles*, *Deciles*, *Percentiles*, *Pearson_Coefficient_Mode*, *Bowley_Skewness* and *Standard_Error* adapted for BPP by Álvarez [19] did not offer significant information for the characterization and some of their values are inconsistent with the structure of the BPP instances.

The exploratory analysis of the indexes for BPP showed the contribution and importance of each, in the characterization of the problem. Starting from an initial set of 31 indexes, 10 were discarded due to redundancy and other 7 were eliminated by their inconsistency or no contribution. The final set of characterization indexes is shown in Table 1.

Table 1. Final set of BPP characterization indexes

Type	**Indexes**
Size	*p* [18]
Centralization	*t* [18]
Dispersion	*Range* [19] *Variation_Coefficient* [19] *d* [18]
Form	*Pearson_Coefficient* [19] *Kurtosis* [19] *Uniformity* [This work]
Location	*Lower_weight* [This work] *Upper_weight* [This work]
Relations items/Bins	*f* [18] *b* [18]
Multiplicity	*Multiplicity* [This work] *Max_Repe* [This work]

The first component, called *Variability*, measures the variation of the weights of items and groups the measures of central tendency, dispersion and location. The second, called *Form*, describes the weight distribution, grouping measures of form, location and size of the problem. The last component, called *Multiplicity*, measures the frequency of occurrence of the weights of items.

The principal components allowed plotting the characteristics of the six groups of instances, in Figure 6 is possible to distinguish clearly between six different classes of instances analyzed.

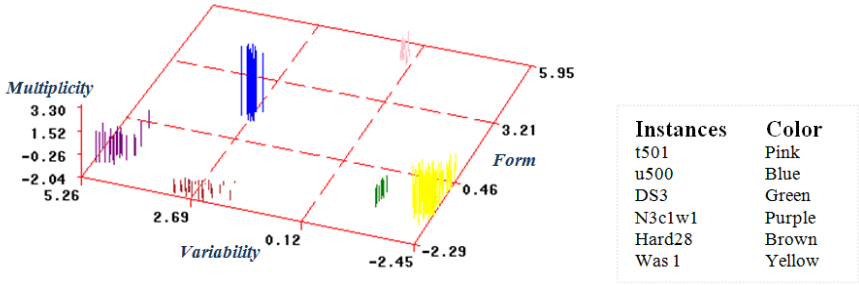


Fig. 6. Scatter plot for the first three principal components of the set of indexes

The analysis of the scatter plot of the weights distribution (defined in Section 5.1) proved the accuracy and discriminatory power of the proposed indexes, because the similarities and differences between the characteristics of the instances belonging to the six sets agree with those observed in the plots.

The analysis of algorithm and final performance measures showed that there were no incorrect, redundant or inconsistent indexes, as each captures important aspects of HGGA-BP algorithm and its performance.

5.3 Phase 3: Study of Relations

In this stage, the final set of indexes of characterization is studied in order to discover relationships of performance detailing the structure of the optimization process. The main objective of this study is to gain knowledge that could explain the performance of the algorithm.

To explain the optimization process is necessary to analyze the relation between the characterization indexes that define: the structure of BPP instances, the behavior shown by the algorithm and the final performance. These relationships were obtained by means of the association analysis, scatter plots and causal inference. Initially all the analyses were performed on the entire set of instances, for an overview of the performance relationships that define the behavior of the algorithm and the difficulty of the BPP instances.

In order to discover the underlying relation structure between BPP characteristics and the final performance of the algorithm we use causal inference. The PC learning algorithm of the TETRAD tool [20] was applied for automatically learning a causal model from the measures of the principal components of BPP (*Variability*, *Form* and *Multiplicity*) and the three performance indexes (*Theoretical_Ratio*, *Best_Fitness* and *Generation*). The causal graph obtained through the analysis is presented in Figure 7, which clearly shows that BPP instances characteristics have a direct influence on the final performance of the algorithm.

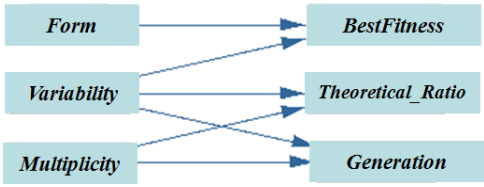


Fig. 7. Causal relations between problem characteristics and final performance

Figure 8 shows the causal graph for the algorithm characteristics (*Deviation_Fitness*, *New_individuals* and *Deviation_Best*) and the final performance. It seems that *New_individuals* measure (highlighted in yellow) have no relation to the final performance, it seems that the procedures used to create new individuals are not helping to escape from local optima.

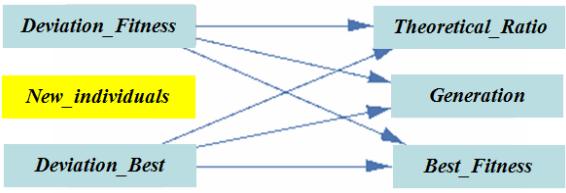


Fig. 8. Causal relations between algorithm behavior and final performance

Figure 9 includes two scatter plots that associate problem and algorithm behavior measures and the theoretical radius of the final solutions of the algorithm. Each plot shows the way a feature of problem (*t* is average weight of the items) affects the behavior (*Deviation_Best* and *Deviation_Fitness*) and the final performance of the algorithm (*Teoretical_Ratio*). From the examination of the plots it can be seen that instances with large items are the easiest. Also, in some cases, diversification of the population helps the algorithm to find better solutions. The plot a) shows a very important algorithmic behavior: for instances with small and medium weight the algorithm does not achieve good results when the best solutions are highly associated with each other, indicating that the algorithm is stuck at local optima. In plot b) we use different colors to highlight different sets of instances, it can be observed that within the same class of instances there are easy and difficult cases.

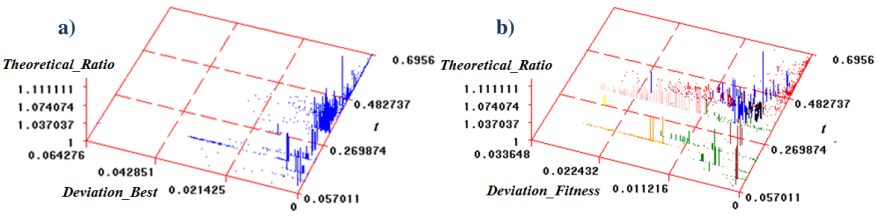


Fig. 9. Relations between problem, algorithm and final performance measures

Figure 10 shows scatter plots illustrating the behavior shown by the algorithm in two instances classes. The difference in the complexity inside every instance class appears to be caused by the shape of the weight distribution as well as the multiplicity and size of the problem. The instances for which the optimal was found were marked in blue and the more "difficult" were marked with black.

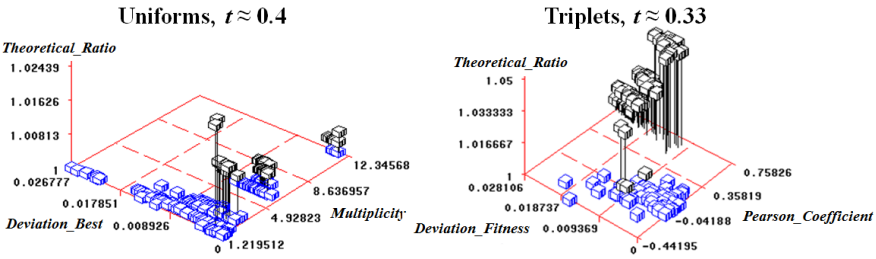


Fig. 10. Relations between problem, algorithm and performance measures

After analyzing the performance relations we can infer the following algorithm behavior explanations:

- The main features that reflect the degree of difficulty of an instance of BPP are the central tendency and variability of the weights of items.
- For instances that have similar values of central tendency and variability measures, the degree of difficulty of the instances is influenced by the form and multiplicity of the set of weights.
- The easier instances are those with larger items and greater variability in the weights of items.
- The bigger the problem, relative to the size of the bin (this implies that the items are large and/or there are many items), the instance is easier; this may possibly be because the solution space is smaller and flatter.
- The strategies included in the algorithm had good results for instances with large items but do not appear to be adequate to address instances with medium and small items.
- For "difficult" instances, strategies for generation of individuals do not allow to add diversity in the population and genetic operators do not lead the algorithm to new regions.

5.4 Redesign of the Algorithm and Performance Improvement

The algorithm behavior explanations gained from the experimental analysis of the optimization process of the HGGA-BP algorithm showed that the strategies used for the creation of solutions and the search space exploration were not helping the algorithm to obtain good solutions for instances with medium and small items, this knowledge was used to redesign the algorithm structure by modifying the procedures that had the greatest impact on the algorithm performance.

After a detailed analysis of the main strategies that define the structure and behavior of the algorithm, the procedures used for the creation and exploration of solutions were modified in order to include a greater diversification and exploitation of the search space. We incorporated efficient random heuristics to increase the solutions diversity and to prevent the premature convergence of the algorithm [13].

Diversification was added to the population by: a) including different packing strategies in the creation of individuals; b) increasing the size of the population; c) adding new individuals to replace individuals with repeated fitness.

Intensification was added to the search by: a) including a new strategy to improve the filling of the bins; b) including a new crossover operator for good solutions; c) including a new crossover operator for the improvement of solutions fitness.

The application of the knowledge obtained in the analysis of the characteristics of the optimization process of BPP using the algorithm HGGA -BP, yielded a significant improvement in the performance of the algorithm. The final results are shown in Table 2, which includes the results obtained by the old and the new version of the algorithm. For every class of instances, it first shows the number of test cases (Column inst.), followed by the results obtained by each version of HGGA-BP: the number of optimal solutions found (Column opt.), the average execution time measured in seconds (Column time (s)) and the average generation (Column gen.).

The experimental results make obvious the usefulness and applicability of the algorithm behavior explanations obtained by means of the experimental approach proposed in this work. The effectiveness of the heuristic algorithm HGGA-BP was improved from 84.8% to 95.4%. Also, it was possible to outperform results of the best state of the art algorithms. HI-BP algorithm [21] was outperformed in three instances of the Gau 1 set (TEST0058, TEST0082 and TEST0005). Perturbation-SAWMBS algorithm [22] was outperformed in three instances of Hard28 set (hBPP640, hBPP531 and hBPP814).

Table 2. Improved results for the HGGA-BP algorithm

Class	inst.	HGGA-BP Original			HGGA-BP Improved		
		opt.	time (s)	gen.	opt.	time(s)	gen.
Uniform	80	52	8.53	47	79	1.67	13
Triplets	80	0	1.90	69	80	5.14	53
Data Set 1	720	692	2.43	43	718	2.67	19
Data Set 2	480	450	0.57	10	480	0.90	6
Data Set 3	10	8	1.75	8	9	8.10	77
Was 1	100	99	0.05	6	100	0.04	3
Was 2	100	98	0.34	28	100	0.99	32
Gau 1	17	9	0.59	54	15	1.92	40
Hard28	28	5	1.53	90	8	6.75	87
NIRUP	53	3	1.09	81	3	4.45	89
Total	**1668**	**1416**	**1.88**	**44**	**1592**	**3.26**	**42**
Effectiveness		**0.848**			**0.954**		

6 Conclusions and Future Work

This work shows that the exploratory data analysis is useful in the study of NP-hard problems and heuristics algorithms, because it allows identifying clearly what are the characteristics of the problem that impact the final performance of the algorithms and to what extent they do.

We propose an experimental approach that combines exploratory data analysis techniques for the performance analysis of heuristic algorithms with the objective to explain the algorithmic optimization process. As a case study we perform a comprehensive study of the optimization process for the Bin Packing Problem (BPP) solved by a heuristic algorithm.

We studied and characterized the structure of bin packing instances and we proposed 5 indexes for the bin packing characterization. The optimization process of BPP was explained by relations, which allowed us to understand the behavior of a genetic algorithm in the solution of BPP instances with different structures. The case study confirmed the importance of applying exploratory data analysis techniques as a guide for understanding the performance of algorithms. The knowledge gained from models of explanation led to improve performance of the HGGA-BP algorithm, correcting the effectiveness of 84.89% to 95.44% for a set of 1668 instances, outperforming the effectiveness of the best state of the art algorithms in some instances.

As future work, in the case of the genetic algorithm, we are planning to make a deeper study within the features of the algorithm which seem to be the most promising to increase the final performance and design new models that includes the impact of the parameters that control the behavior of the algorithm in order to obtain more detailed explanations of the optimization process.

In general, it is expected that the work presented in this paper represents a guideline to study the performance of heuristic algorithms through the application of exploratory data analysis techniques in other algorithms and optimization problems. The experimental approach presented in this work allows obtaining a deep understanding about algorithmic behavior; this knowledge can be used to improve the performance.

References

[1] Garey, M.R., Jonson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman and Company. A classic introduction to the field (1979)

[2] Basse, S.: Computer Algorithms, Introduction to Design and Analysis. Editorial Addison-Wesley Publishing Company (1998)

[3] Cruz Reyes, L., Nieto-Yáñez, D.M., Rangel-Valdez, N., Herrera Ortiz, J.A., González B, J., Castilla Valdez, G., Delgado-Orta, J.F.: DiPro: An Algorithm for the Packing in Product Transportation Problems with Multiple Loading and Routing Variants. In: Gelbukh, A., Kuri Morales, Á.F. (eds.) MICAI 2007. LNCS (LNAI), vol. 4827, pp. 1078–1088. Springer, Heidelberg (2007)

[4] McGeoch, C.C.: Experimental Analysis of Algorithms. In: Pardalos, P.M., Romeijn (eds.) Handbook of Global Optimization, vol. 2, pp. 489–513 (2002)
[5] Hoos, H.H., Stützle, T.: Empirical Analysis of Randomized Algorithms. In: Handbook of Approximation Algorithms and Metaheuristics. Chapman & Hall/CRC, Taylor & Francis Group (2007)
[6] Hooker, J.N.: Needed: An empirical science of algorithms. Operations Research 42(2), 201–212 (1994)
[7] Barr, S., Golden, L., Kelly, P., Resendez, M., Stewart, R.: Designing and Reporting on Computational Experiments with Heuristic Methods. Journal of Heuristics 1, 9–32 (1995)
[8] Cohen, P.R.: Empirical Methods for Artificial Intelligence. The MIT Press, Cambridge (1995)
[9] Lemeire, J.: Learning Causal Models of Multivariate Systems and the Value of it for the Performance Modeling of Computer Programs. PhD thesis. Vrije Universiteit Brussel (2007)
[10] Tukey, J.W.: Exploratory Data Analysis. Addison-Wesley (1977)
[11] Hartwig, F., Dearing, B.E.: Exploratory Data Analysis. Sage University Paper Series on Quantitative Research Methods, vol. 16. Sage, Newbury Park (1979)
[12] Liu, X.: Intelligent Data Analysis: Issues and Challenges. The Knowledge Engineering Review 11, 365–371 (1996)
[13] Quiroz, M.: Caracterización de Factores de Desempeño de Algoritmos de Solución de BPP. Tesis de maestría, Instituto Tecnológico de Cd. Madero, Tamaulipas, México (2009)
[14] Beasley, J.E.: OR-library: Distributing test problems by electronic mail. Journal of the Operational Research Society 41(11), 1069–1072 (1990), http://people.brunel.ac.uk/~mastjjb/jeb/orlib/binpackinfo.html
[15] Klein, R., Scholl, A.: Bin Packing benchmark data sets, http://www.wiwi.uni-jena.de/Entscheidung/binpp/
[16] Euro Especial Interest Group on Cutting and Packing. One Dimensional Cutting and Packing Data Sets, http://paginas.fe.up.pt/~esicup/tiki-list_file_gallery.php?galleryId=1
[17] Cutting and Packing at Dresden University. Benchmark data sets, http://www.math.tu-dresden.de/~capad/cpd-ti.html#pmp
[18] Pérez, J., Pazos, R.R.A., Frausto, J., Rodríguez, G., Romero, D., Cruz, L.: A Statistical Approach for Algorithm Selection. In: Ribeiro, C.C., Martins, S.L. (eds.) WEA 2004. LNCS, vol. 3059, pp. 417–431. Springer, Heidelberg (2004)
[19] Álvarez, V.: Modelo para representar la Complejidad del problema y el desempeño de algoritmos. Tesis de maestría, Instituto Tecnológico de Cd. Madero, Tamaulipas, México (2006)
[20] The TETRAD Project: Causal Models and Statistical Data. TETRAD Homepage, http://www.phil.cmu.edu/projects/tetrad/
[21] Alvim, A.C.F., Ribeiro, C.C., Glover, F., Aloise, D.J.: A hybrid improvement heuristic for the onedimensional bin packing problem. Journal of Heuristics 10(2), 205–229 (2004)
[22] Fleszar, K., Charalambous, C.: Average-weight-controlled bin-oriented heuristics for the onedimensional bin-packing problem. European Journal of Operational Research 210(2), 176–184 (2011)

An Interactive Decision Support System Framework for Social Project Portfolio Selection

Laura Cruz-Reyes[1], Fausto A. Balderas J.[1], Cesar Medina T.[1], Fernando López I.[2], Claudia G. Gómez S.[1], and Ma. Lucila Morales R.[1]

[1] Instituto Tecnológico de Ciudad Madero, México
lcruzr@prodigy.net.mx, {tono.21,lmoralesrdz}@gmail.com,
{rpgamer86,cggs71}@hotmail.com
[2] Universidad Autónoma de Nuevo León, México
flopez65@gmail.com

Abstract. In this paper, we present the development of a Decision Support System (DSS) for the project portfolio selection problem, financed with public funds. The selection of the portfolio is a complex optimization problem with multiple subjective criteria, which are difficult to compare. The Decision Maker (DM) has to find a manageable and most preferred set among many non-dominated (efficient) solutions. There is a vast variety of techniques and software for multi-criteria decision making. However, the portfolio selection is an area that requires more and better software. An interactive "Framework" is presented; it is designed to help the DM to select the best portfolio in a flexible way. The Framework is based on the classic decision process of Simon, the SMART method and a friendly man-machine interface. The SMART method was adapted to allow the DM the discovery of his preferences and to express them on the terms of the objective weights and budget constraints. The graphic user interface assist the DM through the visualization of the impact on the change of preferences and also on the projects within a reference portfolio, this way he can make a decision or adjust the necessary changes.

1 Introduction

This article addresses the development of a Decision Support System (DSS) for project portfolio selection to be financed by public funds (SPP, Social Portfolio Problem). Portfolio selection is a complex optimization problem involving conflicting, subjective criteria difficult to compare.

According to Weistroffer [1], the general problem for the implementation of DSS for project selection is in its beginning and there are few studies on it. Weistroffer criticizes the current DSS and mentions that there is not a methodology that is best for all problems of project portfolios.

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 377–391.
springerlink.com

By the above, in the present investigation a framework was developed that allows interaction of the decision maker (DM) with the decision process involved in selecting public portfolio.

One of the main tasks of management in government organizations at all levels is to evaluate a set of social impact projects competing for financial support. With an amount to distribute less than the demand, the benefit to all competing projects cannot be granted. The decision on the allocation of resources is regularly held by a person who will make the decision whether or not resources goes to a particular project.

Under certain restrictions determined by the orientation of public policies, quality portfolios projects just be formed which maximize the impact (with ideological connotations of the decision maker) of the chosen solution (see Fig.1). This is an big social problem that the cost of poor solutions is simply immense, but its complexity has prevented so far real progress to solve it.

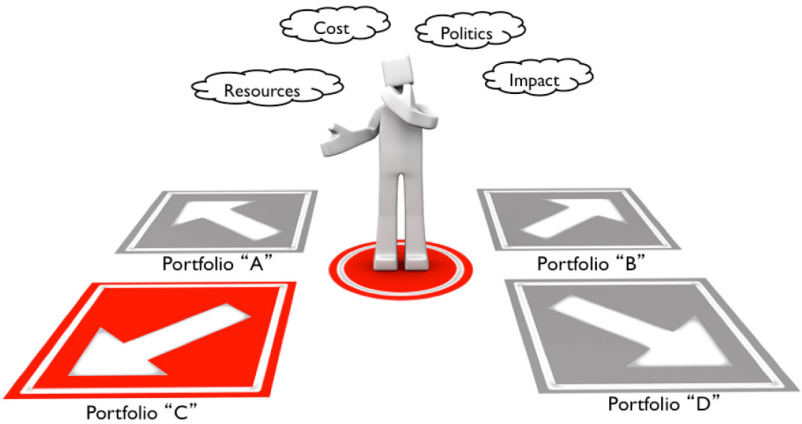


Fig. 1. Decision Maker

Counting on a framework to guide them through an appropriate order of steps, applications and data conversions, large organizations that distribute public resources for public projects may choose, through a framework, projects that have a greater benefit to society and therefore optimize their resources. It is expected that the realization of this project will contribute to decision-makers to make proper use of public resources.

This paper is organized in five parts, starting from the introduction. The second section reviews the general Portfolio Selection Problem that occur during the resources allocation. This problem involves differrent kinds of situation, wich requires different solution. For this reason, the third section describes three solution approaches, based on multi-criteria optimization, for the Social Portfolio Problem (SPP). Besides, it is well known that the decision maker (DM) has to find an acceptable (compromise) solution from among many efficient (non-dominated) solutions given by optimizers. Because the rational capacity of human being is

limited for this task, an interactive Decision Suport System (DSS) is imperaty to mainly identify the most preferred solution. Section 4 analyzes the literatura related with frameworks to develope DSS´s. Finally, Section 5 presents our framework proposed for SPP.

2 Project Portfolio Selection

A *project* can be defined as a complex effort, usually less than three years, consisting of interrelated tasks carried out by various organizations, with a clear objective, timetabled and a budget. A *portfolio* is a set of projects undertaken under the administration of an organization. These projects must compete for scarce resources, because usually the resources provided by the organization are not sufficient to carry out all proposed projects for the portfolio. The *project portfolio selection* is the activity of creating a portfolio from the proposed projects of the organization, which met in the best way with the objectives set by the organization, and not exceeding the amount of resources given or breaking other constraints given by managers.

The portfolio selection process use assessment and evaluation techniques divided into three stages:

- *Strategic considerations*. The techniques used at this early stage may help in determining a strategic approach to the overall distribution of the budget for the portfolio.
- *Individual assessment of projects*. Techniques at this stage can be used for evaluating projects independently of the others.
- *Portfolio Selection*. This last stage deals with the portfolio selection based on the parameters of candidate projects. For Example, the project synergy is important because the value of the portfolio is different from the sum of the values of the individual projects.

Following this three-stage process, Archer [13] proposes a set of suggestions that specify the requirements that addresses each phase:

Stategic Considerations Phase

Strategic decisions relating to the portfolio approach and budget considerations must be made in a broader context that takes into consideration internal and external business factors before the portfolio is selected.

A framework for selecting projects must be sufficiently flexible to allow interested parts to choose a way forward with specific techniques or methodologies which they are comfortable, to analyze relevant data and make decisions about the types of projects on hand.

To simplify the portfolio selection process, it should be organized in a number of stages, allowing decision makers moving logically toward a comprehensive consideration of the projects most likely to be chosen based on theoretical models.

Users should not be overloaded with unneeded data, but should be able to access relevant information when needed.

Project Evaluation Phase

Common Measurements should be chosen that could be calculated separately for each project under consideration. These allow a fair comparison of the projects during the selection process.

The current projects that have reached milestones should be re-evaluated while new projects are being considered for selection. This allows a combined portfolio to be generated without violating the resource constraints at regular intervals due to (a) compliance of the project or neglect, (b) new project proposals, (c) changes in strategic focus, and (e) changes in the environment.

The filtering should be used, based on carefully specified criteria to eliminate projects to be considered before the portfolio selection process is carried out.

Portfolio Selection Phase

Interactions of projects (synergy) through direct dependencies or resource competition should be considered in selecting the portfolio.

The selection of the portfolio must take into account the time-dependent nature of the resource consumption of projects.

Decision makers must provide interactive mechanisms to control and cancel any portfolio generated by any algorithm or model, and also should receive feedback from the consequences of such changes. The project portfolio selection should be adaptive to environments for decision support groups.

3 Social Portfolio Problems, SPP

The selection of projects of a social portfolio, unlike the selection with other types of projects (Research and Development, Information System and Financial investment), requires a special treatment for the following reasons [2]:

a) The quality of projects is usually described by multiple-criteria that are often in conflict.
b) Often, the requirements are not known accurately. Many concepts have no mathematical support due their entire subjective nature.
c) The heterogeneity among potential projects in a portfolio, making it difficult to compare.
d) Information provided by the DM, is not strong so it can be called *incomplete preference information.*
e) The impact on social prosperity, which is the most important concept of the problem of public project portfolio, is a variable of subjective nature and usually takes very long term to achieve the expected benefit, depends on the DM put a value on this variable for each project.

The above points are characteristic of public projects such as projects focused to education, health, public transport and general prosperity. To our knowledge, there are only two scientific approaches concerning this type of project portfolio [2]:

1) The most used is the cost-benefit analysis. Some statistical techniques can reduce the number of variables of a project to easily represent a monetary value.
2) Using multi-criteria analysis to explore the preferences of the DM as well as manage the inherent complexity of DM. Multi-criteria analysis is a good alternative to overcome the limitations of cost-benefit analysis, since it can handle ambiguous and intangible preferences and conditions of veto. Multi-criteria analysis provides techniques for selecting the best project or a small set of best projects that are equivalent, classifying the projects into several categories according to predefined preferences or priorities given by the DM.

3.1 Multi-criteria Solution Approaches

Using multi-criteria analysis, the decision on which projects should receive funding, may be based on the best individual projects or based on the best portfolio on the set of all feasible portfolios. For the problem of public portfolio is insufficient to compare projects with each other, as this does not guarantee that all the best projects is the best portfolio.

For example, under the scenario of portfolio selection, it is possible to reject a good project (in terms of social impact), because it requires an excessive financing that might conflict with the preferences of a decision maker who wants to encourage more projects.

The preferences of the decision maker, to form a portfolio, can be modeled from different perspectives, using different approaches to reach the goal. These approaches depend on who the decision maker is (A single person or a heterogeneous group), and how much effort the DM is willing to invest in finding the solution to the problem. This work considers the information about the impact of projects; their quality can be obtained from the DM using three different approaches to solving this problem, two of them are described in the following sections.

Value Function

Given a set of premises (insufficient funds, projects that meet minimum requirements of acceptability, ethical conduct, etc.), it is possible to create a value model for portfolios from the perspective of the main decision maker (SDM, Supra Decision Maker). The set of premises to consider might be based on the following assumptions [9]:

a) Each project and each portfolio has a subjective value for the SDM, even if the initial value can not be quantified.
b) The SDM have a consistent system of preferences or has aspirations to build it.

c) The SDM want to invest a considerable amount of mental effort in order to define the consistent set of preferences and produce the value model.

Priority Ranking

Flerida develops a model for the composition of socially oriented portfolios [14]. Information concerning the quality of the projects is in a priority (ranking) of projects, which can be obtained by a suitable application of an adequate multi-criteria method, but the ranking does not take a proper assessment of social impact.

The model provides a preference relation on the portfolio positions in the ranking of projects, project costs and the rejection of the DM toward costly projects. A better portfolio is mainly found through multi-objective optimization respect to the violations of preset preferences of the decision maker and the cardinalities of competing portfolios.

An example of this approach is the division of the ranking in five categories labeled as Vanguard, Medium-High, Medium, Medium-low and Rear. With this characterization four preference relations are built according to the ranking: absolute preference, strict preference, weak preference and indifference.

It is necessary to compare the quality of the portfolios to find the best. The best portfolio is defined not only by the quality of the projects but also by the number of projects it contains. Some discrepancies may be acceptable between the information given by the ranking and the decisions concerning the approval of some projects, provided that this increase in the number of projects in the portfolio. However, this inclusion should be controlled because the admission of unnecessary discrepancies when comparing portfolios is equivalent to underestimate the ranking information. The model of Flerida considers 3 objectives:

1) Number of strong discrepancies (D_s)
2) Number of weak discrepancies (D_w)
3) Portfolio cardinality (n_c)

The portfolio to be elected should be the best solution to the multiobjective problem:

$$Minimize\ (D_s,\ D_w)$$

$$Maximize\ (n_c) \tag{1}$$

$$C \in R_F$$

Where C denotes portfolios and RF is the feasible region according to budgetary constraints.

4 Frameworks for the Development of Decision Support Systems

In the work presented by Jichang Dong, et al. [3], a framework for portfolio selection is proposed, which is adaptable to the needs of financial organizations

and individual investors. It focuses on the implementation of a web-based framework, adopting technologies such as online analytical processing, as an additional tool for analysis, as well as the parallel use of a virtual machine to improve overall performance. This framework, however, leaves many problems unresolved, like the problems of transaction costs, multi periods and incomplete information. On the other hand, it does not use many of the existing methodologies and models because they are too complex and require lots of input data.

RPM is a methodology for decision support to analyze portfolio problems with multiple criteria [4]. The RPM framework extends the use of preferential programming methods in the portfolio problem. RPM is based on the calculation of the set of non-dominated portfolios accord to the incomplete information. It includes performance measures that help to analyze the attractiveness, robustness of portfolios and individual project proposals. According to the authors, this approach provides a systematic and transparent framework for decision support that may have a valuable contribution in the selection of project portfolio, especially when the number of proposals is high. In summary the RPM approach provides:

- A scoring model applicable to the evaluation of the projects.
- Proactive Analysis of uncertainty parameters (robustness).
- Identification of projects to be unquestionably included or excluded.
- Analysis and negotiation of projects on the border.
- Transparency of individual projects through performance measures.
- Tentative/split Conclusions at any stage of the selection process.
- A variety of functions to plot interactively the decision support.

In the research done by Castro [6], it is designed an extensible framework that serves to adjust algorithms for exploration and optimization of the space of the project portfolio of R & D in public organizations by implementing a factorial experiment. The framework can also be used for adjusting parameters in optimization methods for the project portfolio of R & D in public organizations.

The implementation of the framework is incomplete without logbook functions and data recovery; there is no possibility to compare different methods for portfolio optimization.

The framework was designed following a three-tier architecture design paradigm: the Model-View-Controller. The functions are grouped into packages seeking a balance between the internal cohesion of each packet and the coupling with the rest of the packets.

Among the most representative researchers are Dong [3] and Liesio [4]. Table 1 shows these and others. The analysis of related work reveals that the activities that have been less addressed can be seen in column 5 and column 7, they are related with the generation of a reference portfolio and the architecture design, respectivately. This paper proposes a DSS that cover all the analyzed stages of the decision-making through a comprehensive architectural design.

Table 1. Comparative table of frameworks for decision support systems

-	Project Capture		Interaction with the DM	Reference Portfolio Generation	Architechture	
	Data	Preferences			Conceptual	Physical
Dong, et. al [3]	✓	✓	✓	-	✓	-
Castro [6]	✓	-	-	-	-	✓
Liesio [4]	✓	✓	✓	-	✓	-
Lourenco [7]	✓	-	✓	-	-	-
Yeh, et. al. [8]	-	✓	✓	-	✓	-
Este trabajo	✓	✓	✓	✓	✓	✓

5 Framework Proposal

We designed an interactive DSS to assist the DM to select the best portfolio in a flexible manner. The core of the DSS is a Framework, which includes the SMART method and a friendly man-machine interface with the next characteristics:

1) The SMART method was adapted to allow the DM the discovery of their preferences and to express them in terms of weights of the objectives and budget constraints.
2) The Framework is based on the classical process of decision making by Simon [9], and the model-view-controller paradigm.
3) The graphical interface assists the DM by visualizing the effects of changes in their preferences and projects on a reference portfolio, so a decision can be made.

5.1 SMART Method for Preferences Elicitation

To obtain the preferences, the SMART (Simple Multi Attribute Rating Technique) [10] method was adapted; at each stage the DM is involved with the required information:

1) Identify the *alternatives* and relevant *attributes* for the problem and for each attribute assign a value to each alternative. These values form a descending *order* of preference.
2) For each *attribute* determine the associated *absolute weight* expressed as a percentage of preference and according to the established order.
3) For each *alternative* distribute the *absolute weight* associated with the evaluated attribute according to their preferences.

4) Calculate the *relative weights* of each alternative with its attribute, based on the percentages given.
5) Perform sensitivity analysis to check the consistency of preferences; this implies repeating the process until the DM requires it.

5.2 Model-View-Controller

The MVC (Model-View-Controller), see Figure 2, allows us to separate the control logic (what needs to be done but not how), business logic (how things are done) and the presentation logic (how to interact with the user). MVC is recommended for design interactive web applications [11]. Using this type of pattern is possible to achieve higher quality, an easier maintenance and extension. One of the most important things that allow the use of this pattern is to normalize and standardize the software development.

The architecture of the DSS framework was design by the Model-View-Controller pattern with the next general functions:

- In the view layer, the user interface is developed.
- In the controller layer, the problem of portfolio selection of social projects is solved.
- In the Model layer, there are the protocols for: data import, incorporation of obtaining methods and incorporation of optimizers. Also this layer includes two entities: database and libraries.

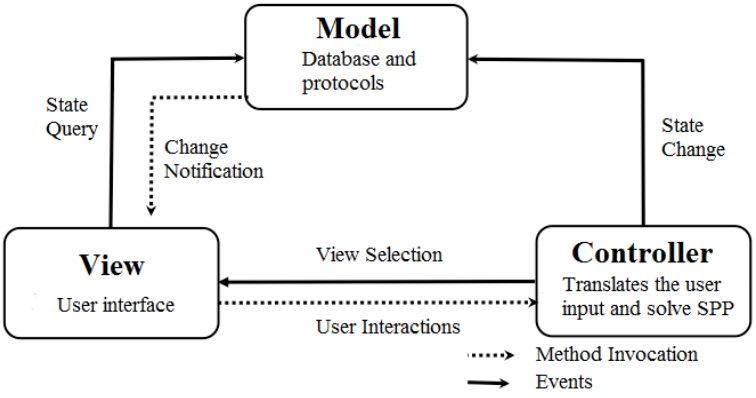


Fig. 2. MVC pattern for SPP

5.3 Decision Making Process Based on Simon

Another perspective of the architecture of the framework was design following the traditional process of decision making of Simon: Intelligence, Design and Selection, as shown in Figure 3. The interaction with the DM is detailed in Figure 4.

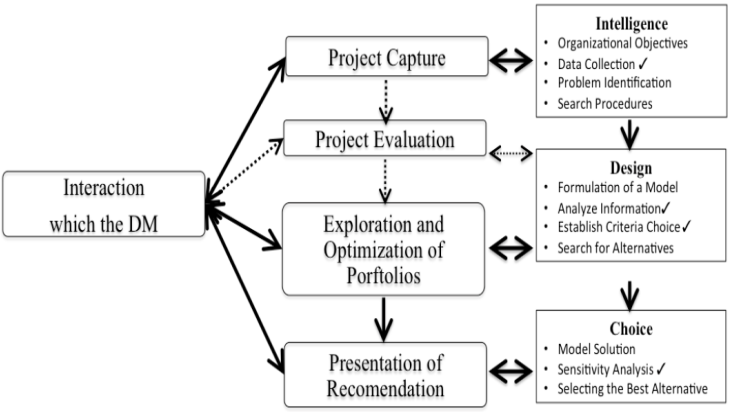


Fig. 3. Decision making process for SPP

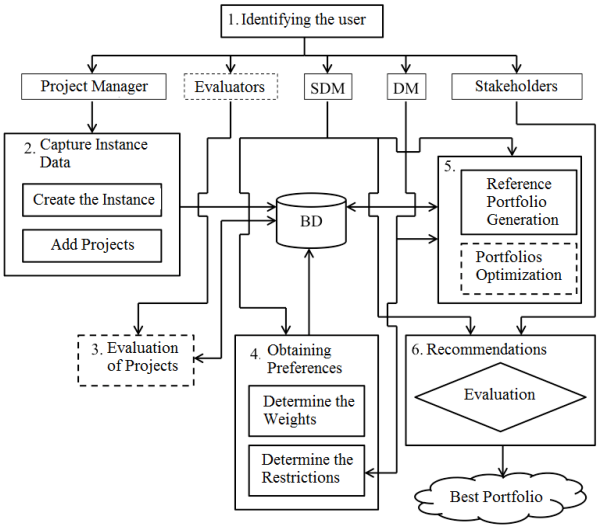


Fig. 4. Interaction with the DM

5.4 Architecture

The architecture shown in Figure 5 is the result of the integration of three conceptual independent designs: model-view-controller (MVC in Figure 2), decision making processs (DMP in Figure 3) and user interaction (Figure 4). The modules with solid lines were implemented with simple stategies to show the feasibility of the proposed method for preference elicitation. The dotted lines represent what was left out for future work.

The description of the final architecture follows the DMP phases (intelligence, design and election), for each phase the three layers of MVC are explained. Because in the social portfolio selection users can participate in all processes, restricting their tasks, we decided not to make a roles distinction in this architecture and include it as an access control scheme.

Intelligence

The intelligence phase within the view layer, involve the data capture of the instance, the project evaluation capture, the capture of preferences. All of these are within the Intelligence since they involve an in depth preliminary analysis.

In the controller layer, there are the goals, scales, projects and project evaluation breakdown of preferences methods, each of these in this layer undergoes a consistency and coherence checker.

In the model layer is the protocol for data import, so that data can be processed, these should preferably be in the same format, and the protocol to incorporate techniques of project evaluation and preference disaggregation.

Design

Here we find the capture of expected portfolios provided by the DM. It will be a reference profile of alternatives and used for the comparison of portfolios, all in the view layer, due this is what the end user see. The DM forms his reference portfolios and has the possibility to compare these portfolios against generated by optimizers, recommenders and benchmark portfolios.

In the controller layer, we have the methods of obtaining preferences and portfolio evaluation; here also all of of these undergoes a consistency and coherence checker.

In the Model layer, we need protocols for the incorporation of new methods to get the decision maker's preferences by elicitation and methods for the portfolio evaluation

Election

In the view layer, we have the visual representation of the recommendation, it represents visually the portfolios for the user and the projects that are in the portfolio, and it presents the information in a visual and friendly manner.

In the controller layer, we Optimization Manager, which is where the optimizers developed by the network, in the future it is planned that in this part the algorithms for different approaches to the public portfolio problem will be stored. The architecture provides the existence of three approaches for solution (function value, rank, Fuzzy Preference Relation), which help the DM depending on the amount of effort willing to contribute. Also, here is also the recommendation manager.

The Model layer have the protocol for the incorporation of portfolio optimizers and the protocol to incorporate recommenders. The latter is necesarry to overcome the limited rational capacity of the DM to deal with a big set of optimized portolios.

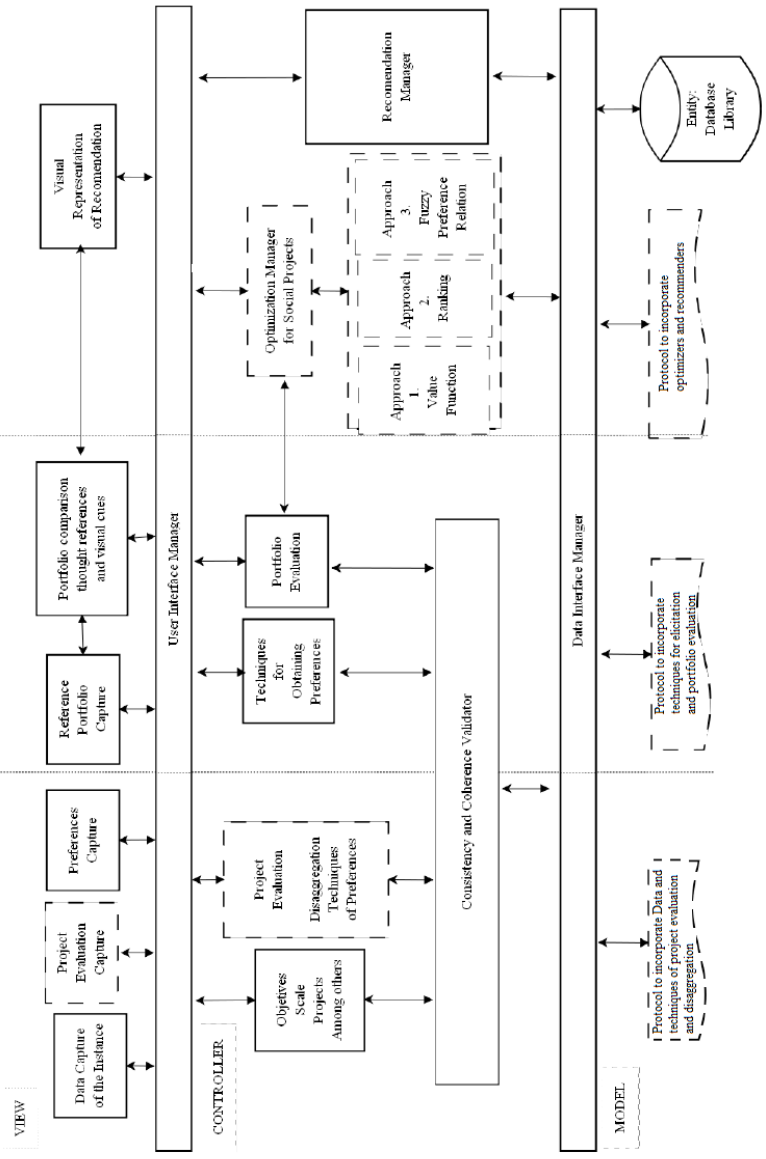


Fig. 5. Process architecture

5.5 User Interface

The Figures 6 and 7 correspond to two screens showing a part of the interaction of the DM with the DSS. The first shows the result of applying the SMART method to generate two reference portfolios. In the second, the user can choose the final portfolio looking at the impact of the portfolios, which are differentiated by colored bubbles.

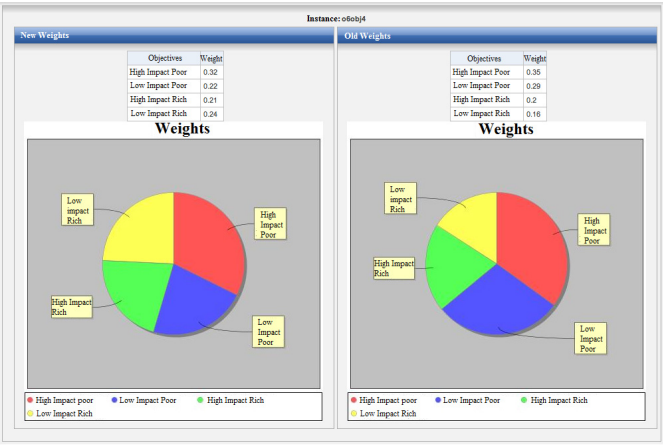


Fig. 6. Weights Editing

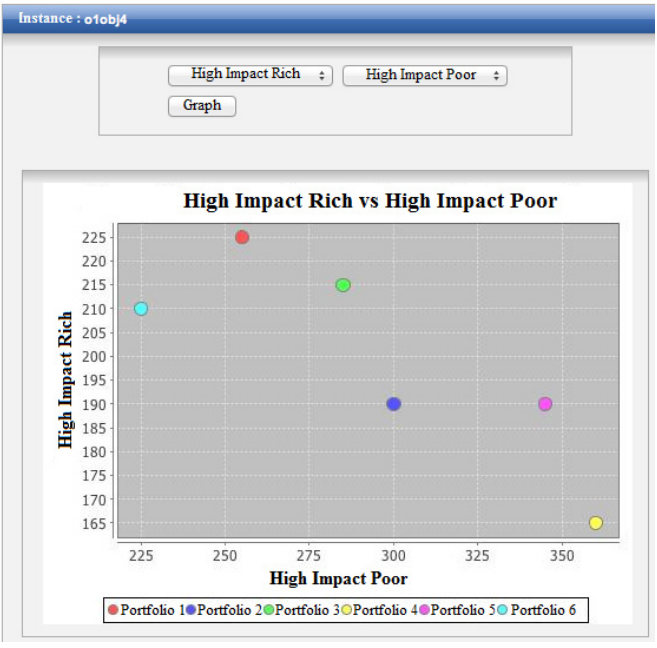


Fig. 7. Presentation of the recommendation

6 Conclusions and Future Work

A DSS was developed for the portfolio selection problem of social projects with the following contributions:

a) A Framework based on MVC that involves the entire decision making process (Intelligence, Design, Election) and its interaction with the DM.
b) A simple method for obtaining preferences of a single DM.
c) A simple and intuitive interface, which facilitates the obtaining of DM preferences, without so many complications, and visually displays the effects of changes in their preferences.

The proposed solution helps to solve some issues involved with the interaction of a single DM whose preferences are difficult to obtain. Because in the subjective world, the opinions vary widely and consensus between DM´s and the organization is hard to find**,** we need a new robust method that consider all this interactions. Besides, as a future work an evaluation in terms of usability is considered.

Other recommendations of future work are:

- Design and implement of the DSS with multiple views, either for each type of user, the available time of the DM, and the access control scheme.
- Develop protocols to have and extensible DSS, that can be easily upgraded with new methods.
- Complement the DSS with different solution approaches to the problem of social project portfolio, as well as the group decision support for these approaches.

References

[1] Weistroffer, H.R., Smith, C.H.: Decision support for portfolio problems. Southern Association of Information Systems (SAIS). Savannah, Georgia (2005)
[2] Fernández, E., Navarro, J.: A genetic search for exploiting a fuzzy preference model of portfolio problem with public projects. Kluwer Academic Publishers (2003)
[3] Dong, J., Du, H.S., Wang, S., Chen, K., Deng, X.: A framework of web based decision support systems for portfolio selection with olap and pvm. Decision Support Systems 37(3), 367–376 (2004)
[4] Liesiô, J.: Portfolio decision analysis for robust project selection and resource allocation. Helsinki University of Technology (2008)
[5] Sprague, R.H.: A framework for the development of decision support systems. MIS Quarterly, 1–26 (1980)
[6] Castro, M.A.: Desarrollo e Implementación de un Framework para la Formación de Carteras de Proyectos de I&D en Organizaciones Publicas. MS Thesis (2007)
[7] Lourenco, J.C., Costa, C.A.: PROBE a multicriteria decision support system for portfolio robustness evaluation. Technical report, Working Paper LSEOR 09.108, London School of Economics, London (2009)

[8] Yeh, C.H., Deng, H., Wibowo, S., Xu, F.: Fuzzy multicriteria decision support for information systems project selection. International Journal of Fuzzy Systems 12(2), 170–174 (2010)
[9] Navarro, J.: Herramientas inteligentes para la evaluación y selección de proyectos de investigación-desarrollo en el sector público. PhD Thesis, Universidad Autónoma de Sinaloa, Sinaloa, México (2005)
[10] Edwards, W.: How to use multiattribute utility measurement for social decision making. IEEE Transactions on Man and Cybernetics 7(5), 326–340 (1977)
[11] Sistemas de Información Cooperativos Universidad de Malaga. Java Server Faces Tutorial. Departamento de Lenguajes y Ciencias de la Computación, Universidad de Málaga (2011)
[12] Russell, A.: Managing High-Technology Programs and Projects, 2nd edn. Wiley, New York (1992)
[13] Archer, N.P., Ghasemzadeh, F.: An integrated framework for project portfolio selection. International Journal of Project Management 17, 207–216 (1999)
[14] Fernandez, E., Felix, L.F., Mazcorro, G.: Multi-objective optimization of an outranking model for public resources allocation on competing projects. International Journal of Operational Research 5(2), 190–210 (2009)

Constructive Algorithm for a Benchmark in Ship Stowage Planning

Laura Cruz-Reyes[1], Paula Hernández H.[1], Patricia Melin[2], Héctor J. Fraire H.[1], and Julio Mar O.[3]

[1] Instituto Tecnológico de Ciudad Madero, México
{lcruzr,hfraire}@prodigy.net.mx,
paulahdz314@hotmail.com
[2] Tijuana Institute of Technology, Mexico
pmelin@tectijuana.mx
[3] Universidad Autónoma de Tamaulipas, México
jmar@uat.edu.mx

Abstract. The efficiency of a maritime container terminal mainly depends on the process of handling containers, especially during the ships loading process. A good stowage planning facilitates these processes. This paper deals with the containership stowage problem, referred to as the Master Bay Plan Problem (MBPP). It is a NP-hard minimization problem whose goal is to find optimal plans for stowing containers into a containership with a low containership operation cost, subject to a set of structural and operational restrictions. For MBPP, data are not available for confidentiality reasons. The lack of a performance evaluation benchmark of solution algorithms for MBPP raises the need for a generation of instances. Due to this limitation, we present a generation scheme of instances for the MBPP, which is based random generation according on selected sets of parameters. The parameters are variable within certain ranges to characterize the vessel and containers; the ranges are real-life values taken from the literature. A constructive loading heuristic for stowing containers into a containership is proposed in this paper to have reference solutions. An instance set, its known-best solutions and the generator are available on-line.

1 Introduction

Stowage planning for container ships is a core activity for the shipping lines. This problem, known as MBPP, is a difficult problem due to the combinatorial nature of alternative allocations of containers stowed at locations on the ship, and the number of constraints associated with the type of vessels and containers. Good planning of stowage container has great impact on a port terminal, because it facilitates loading and unloading of containers, which have high cost.

Several researchers have proposed various algorithms for solving real-world instances, but their results are not repeatable, because the instances are not available for confidentiality reasons. In this sense, there is no a reference point for

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 393–408.
springerlink.com

comparing the quality of the proposed algorithms. One possibility is to use the result obtained by the algorithms on the same set of artificial instances. Another alternative is to use the limits produced by exact methods. In this paper we propose a simple strategy following the first approach.

With the proposed generator, instances are obtained randomly by considering ranges of the parameters that define an instance of MBPP. These ranges were determined from reports in the literature for real-world instances. In order to have reference solutions, we create a set of instances and propose a Constructive Loading Heuristic, which does the stowage of containers without considering load balancing. The algorithm, instances and its solution are available online to enable the reproduction of our experiments and the evaluation of new algorithms.

This paper is organized into seven parts. The second section reviews basic concepts. The third section describes MBPP. Section 4 presents a heuristic proposed to solve MBPP. Section 5 describes the proposed instances generator. Section 6 corresponds to the experimentation.

2 Background

In this section the information related to research is described. The concepts of a port terminal are very wide; the most commonly used in this work are described below.

2.1 Maritime Port

Ports are interfaces between the different transport modes and are typically combined transport centers. In short, they are multifunctional, commercial and industrial areas when the goods are not only in transit, but also are handled, manufactured and distributed. The ports are multifunctional systems, which, to function properly, must be integrated into the global logistics chain [1, 2].

2.2 Port Terminal

In a port, maritime terminals can be found, that are specific sections of the port where a company or more can make loading and unloading operations of goods and all other operations required to perform these processes. The maritime terminals have operation and flow of containers areas [3, 4].

2.3 Container

A container is a metal box in which goods can be stored. Each container has a weight, height, length, and port where it has to be unloaded (discharge port), and may need to be provided with electric power (reefer container). Containers can be 20', 40', or 45' long, and 8'6" or 9'6" high (high-cube containers).

Each cell in a stack can hold one 40-foot container or two 20-foot containers, but there may be some cells in the location that are limited to a single length. Containers that are already on board the vessel when the stowage plan is made are called *loaded containers*. A container in a stack is *overstowing* another container in the stack if it is stowed above it and discharged at a later port. An overstowing container is expensive, since it must be removed in order to discharge the overstowed container [5].

The basic container unit today is of size 20'×8'×8'6" (length, width, height), also referred to as a TEU (Twenty-foot equivalent unit). The containers prevailing in maritime, road, and rail transport have a length of 40' feet, represented by two TEU, but also referred to as FEU (Forty foot equivalent unit) [6].

2.4 Container Vessel

Container vessel is a ship that transports box formed containers on a fixed cyclic route. The cargo space of a vessel is divided into bays. Each bay is divided into an on deck and hold part by a hatch cover, which is a flat, leak-proof structure that prevents the vessel from taking in water and allows containers to be stowed on top of it. On and below deck parts of a bay are transversally divided into stacks that are one container wide and are composed of two TEU stacks and a FEU stack. We can see all these elements in Figure 1 [5].

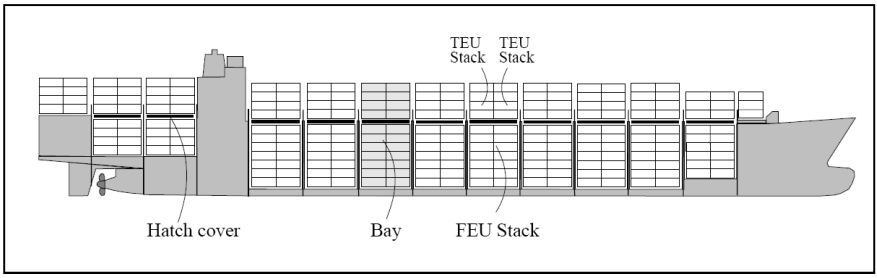


Fig. 1. The arrangement of cargo space in a container vessel

2.5 Locations in the Vessel

Vessel contains a set of locations, and each one is identified by three indices. In particular, each location is addressed by the following identifiers:

- *Bay*, that gives its position relative to the cross section of the ship (counted from bow to stern).
- *Row*, that gives its position relative to the vertical section of the corresponding bay (counted from the centre to outside).
- *Tier*, that gives its position related to the horizontal section of the corresponding bay (counted from the bottom to the top).

Thus a container is located in a given bay, on a given row and on a given tier [5, 7, 8]. The Figure 2 shows that each 20' bay is numbered with an odd number, i.e. bay 01, 03, 05, etc., while two contiguous odd bays conventionally yield one even bay, for the stowage of 40' containers, i.e. bay 06= bay 05 + bay 07. The effective even bays depend on the particular structure of the ship under consideration but, in any case, each even bay, is associated with two contiguous odd bays.

As for the second index (row), ship locations have an even number if they are located on the seaside, i.e. row 02, 04, 06, and an odd number if they are located on the yard side, i.e. row 01, 03, 05, etc.

Finally, for the third index (tier), the levels are numbered from the bottom of the hold to the top with even number, i.e. tier 08, 10, 12, etc., while in the upper deck possible numbers are 82, 84 and 86. Note that, in the final stowage plan such tier numbers allow the containers stowed in the hold to be distinguished from those in the upper deck.

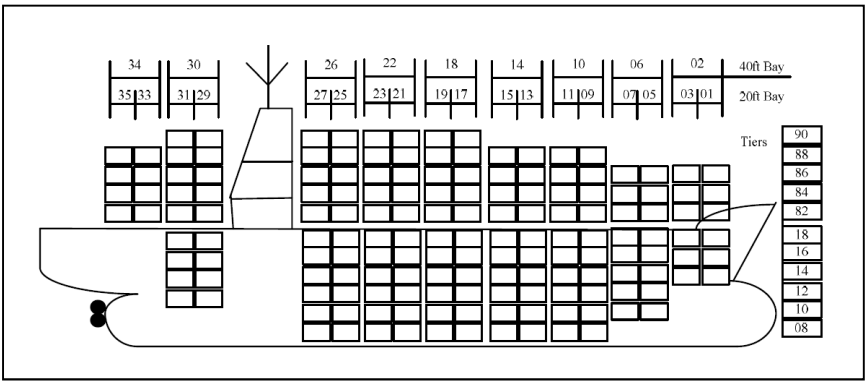


Fig. 2. Cross-sectional view of a containership [9]

3 Description of Master Bay Plan Problem

The stowage of a containership is carried out daily by each terminal management. This problem can be to define as following [7]:

Given a set C of n containers of different types to be loaded on the ship and a set S of m available locations on the containership, we have to determine the assignment of each container to a location of the ship, in such a way, to satisfy all the given structural and operational constraints related to ship and containers, and to minimize the total stowage time.

In MBPP each container $c \in C$ must be stowed in a location $l \in S$ of the ship. The l-th location is actually addressed by the indices i, j, k representing, respectively: the bay (i), the row (j), and the tier (k). We denote by I, J and K, respectively, the set of bays, rows and tiers of the ship, and by b, r and s their corresponding cardinality.

The objective function is expressed in terms is the sum of the time t_{lc} required for loading a container c, $\forall c \in C$, in location l, $\forall l \in S$, such that $L = \sum_{lc} t_{lc}$. However, when two or more quay cranes are used for the loading operations the objective function is given by the maximum over the minimum loading time (L_q) for handling all containers in the corresponding ship partition by each quay crane q, that is $L = \max_{QC}\{L_q\}$, where QC, is the set of available quay cranes.

The main constraints that must be considered for the stowage planning process for an individual port are related to the structure of the ship and focused on the size, type, weight, destination and distribution of the containers to be loaded. The description of each of them will be found below.

Size of containers. Usually, set C of containers is considered as the union of two subsets, T and F, consisting, respectively, of 20 and 40 feet containers, such that $T \cap F = \emptyset$ and $T \cup F = C$. Containers of 40 feet require two contiguous locations of 20 feet. Note that according to a practice adopted by the majority of maritime companies, bays with even number are used for stowing 40' containers and correspond to two contiguous odd bays that are used for the stowage of 20' containers. Consequently, if a 40' container is stowed in an even bay (for instance bay 02) the locations of the same row and tier corresponding to two contiguous odd bays (e.g. bay 01 and bay 03) are not anymore available for stowing containers of 20'.

Type of containers. Different types of containers can usually be stowed in a containership, such as standard, carriageable, reefer, out of gauge and hazardous. The location of reefer containers is defined by the ship coordinator (who has a global vision of the trip), so that we know their exact position. This is generally near power points in order to maintain the required temperature during transportation. Hazardous containers are also assigned by the harbour-master's office which authorizes their loading. In particular, hazardous containers cannot be stowed either in the upper deck or in adjacent locations. They are considered in the same way as 40' containers. Note that, for the definition of the stowage plan we consider only dry and dry high cube containers, having exterior dimensions conforming to ISO standards of 20 and 40 feet long, either 8 feet 6 inches or 9 feet 6 inches high and 8 feet depth.

Weight of containers. The standard weight of an empty container ranges from 2 to 3.5 tons, while the maximum weight of a full container to be stowed in a containership ranges from 20–32 and 30–48 tons for 20' and 40' containers, respectively. The weight constraints force the weight of a stack of containers to be less than a given tolerance value. In particular the weight of a stack of 3 containers of 20' and 40' cannot be greater than an a priori established value, say *MT* and *MF* respectively. Moreover, the weight of a container located in a tier cannot be greater than the weight of the container located in a lower tier having the same row and bay and, as it is always required for security reasons, both 20' and 40 containers cannot be located over empty locations.

Destination of containers. A good general stowing rule suggests to load first (i.e. in the lower tiers) those containers having as destination the final stop of the ship and load last those containers that have to be unloaded first.

Distribution of containers. Such constraints, also denoted stability constraints, are related to a proper weight distribution in the ship. In particular, we have to verify different kinds of equilibrium, namely:

- Horizontal equilibrium: the weight on the right side of the ship, including the odd rows of the hold and upper deck, must be equal (within a given tolerance, say Q_1) to the weight on the left side of the ship.
- Cross equilibrium: the weight on the stern must be equal (within a given tolerance, say Q_2) to the weight on the bow.
- Vertical equilibrium: the weight on each tier must be greater or equal than the weight on the tier immediately over it. Let us denote by *L* and *R*, respectively, the set of rows belonging to the left /right side of the ship and by *A* and *P*, respectively, the sets of anterior and posterior bays of the ship.

The MBPP is NP-hard combinatorial optimization problem. In Avriel et al. [10] the demonstration of the complexity is presented. The following defines the problem.

Given:

$S =$ Set of available locations on the ship, where the *s*-th location is identified by the indices i, j, k, such that $i \in I$, $j \in J$ y $k \in K$ representing, respectively its bay, row and tier address.

$C =$ Set of containers to be loaded into the ship.

$T =$ Stowage time required for loading the set of containers. Each t_{lc} is given by time for handling container c from the quay to the location l of the ship.

We seek a solution X, that minimize the total stowage time *L*:

$$Min\ L = \sum_{l \in S} \sum_{c \in C} t_{lc} x_{lc} \qquad (1)$$

$$X \in R_F$$

Where:

- x_{lc} are binary variables (0,1) of decision of the problem, $l = 1, \ldots, |S|$, $c = 1, \ldots, |C|$ and $x_{lc} \in X$.
- R_F is the feasible region determined by five types of restrictions on container (size, type, weight, destination and distribution).

4 Constructive Loading Heuristic

The constructive loading heuristic proposed is described in Algorithm 1. This heuristic has as purpose to construct an initial solution for MBPP. This proposal does not consider the constraint of distribution of containers.

The operation of the Algorithm 1 can be divided into three phases. In the initial phase (Line 1), the set of containers C is partitioned in $2\times D$ subsets, where D is the total number of destination ports. Each subset is identified by C_{hX}, where $h = 1, \ldots, D$ and $X \in \{T, F\}$.

In the last one phase of the algorithm (Lines 3 to 20) is trying to load containers into the vessel. First containers belonging to C_{hT} (Lines 4 to 19) are assigned and subsequently this procedure is repeated again to load containers belonging to C_{hF}. In this procedure first the containers of each subset is sorted in decreasing order of the weights (Line 4). Then the largest number of containers to locations available to each bay of each subset hX is trying to assign (Lines 5 to 15), so that the heaviest to be loaded first (Lines 7 to 13). Finally in this last phase, the remaining containers are trying to locate above the loaded containers, without violating the size, weight and destination constraints (Lines 16 to 19).

Algorithm 1. Constructive loading heuristic (CHL). It generates an initial solution for MBPP.

1. Let $C' = \{C_1, C_2, \ldots, C_h, \ldots, C_D\}$ be a partition of C, where C_h denotes the set of containers having as destination port h. Split C_h according to the type of containers, that is 20' and 40', thus obtaining sets C_{hT} and C_{hF}, respectively.
2. Apply the bay partitioning procedure based in Ambrosino et al. [11] to determine the subsets $I_h \subseteq I$ of bays assigned to destination h, h=1,…,D. Split each I_h according to the type of bays, that is 20'(odd bays) and 40' (even bays), thus obtaining sets I_{hT} and I_{hF}, respectively.
3. For each destination h, starting from the last one (D) back to 1, assign first containers belonging to C_{hT} (Twenty) as follows: {this cycle (Lines 3 to 20) is repeated for containers belonging to C_{hF} (Forty)}
4. Sort C_{hX} (where $X \in \{T, F\}$) in decreasing order of the weights.
5. Repeat – While $C_{hX} \neq \emptyset$ and $I_{hX} \neq \emptyset$
6. Select $\mathrm{i} \in \mathrm{I_{hX}}$, the first one on the set
7. For *cont*=1 to $|C_{hX}|$
8. For k=1 to K
9. For j=1 to J

10. If location (i,j,k) is available, assign container $c \in C_{hX}$ to location (i,j,k), then set $C_{hX} = C_{hX} \setminus \{c\}$, *cont*=*cont*-1, *k*=*K* and *j*=*J*.
11. End For
12. End For
13. End For
14. Set $I_{hX} = I_{hX} \setminus \{i\}$
15. End While
16. If $C_{hX} \neq \emptyset$ and $I_{hX} = \emptyset$, try to locate the remaining containers without violating the size, weight and destination constraints as follows:
17. If $C_{hT} \neq \emptyset$, the remaining containers are possibly located above 20' containers having destination $h' \geq h$ without violating the weight constraints.
18. If $C_{hF} \neq \emptyset$, the remaining containers are possibly located above 20' containers if they have the same destination, otherwise, either above 20' or 40' containers having destination $h' \geq h$, provided that the weight constraints are satisfied.
19. End If
20. End For

5 Instances Generator Description

Recent work analyzed so far have worked with industrial collaborators, therefore to validate their proposed approaches, they have used real test instances, which are private [1]. Due to this limitation, in this work we present an instances generator for MBPP; the main characteristics of this generator are described below.

The proposed generator is based on selected sets of parameters. The supported parameter values are listed in Table 1. Certainly, some dependencies among parameters may exist, e.g. the containers number to be loaded will likely correlate with the vessel capacity (TEUs of the ship). However, we have not restricted the parameterization of the generator with regard to this, in order to provide highest possible flexibility for the generation of MBPP instances.

The instances, that are generated thought the proposal scheme, have some particulars characteristics, as well the container vessel as the set of containers, at this moment. Figure 3 shows the structure of the generated output file. The first 12 rows are for the single parameters. The next n rows are for specify the containers to load; each row has five parameters for one container. The final rows are for the average loading time of any container into each vessel location; the set of times is specified as a matrix of $r \times s$. The next section details and exemplifies this format.

Table 1. Parameter values

Parameter	Description	Ranges
B	Number of even bays	{6, 11}
R	Number of rows	{4, 12}
S	Number of tiers	{5, 11}
sH	Number of tiers in the hold	{3, 6}
sD	Number of tiers in the deck	{2, 5}
Q_1	Maximum horizontal equilibrium tolerance in tons	{20, … , 30}
Q_2	Maximum cross equilibrium tolerance in tons	{40, … , 50}
MT	Maximum vertical equilibrium tolerance in tons for 20' containers	{45}
MF	Maximum vertical equilibrium tolerance in tons for 40' containers	{66}
TEU	Number of containers in TEUs	{138, … , 1800}
N	Number of containers in absolute number (#)	{100, … , 1413}
T	Number of 20' containers	{62, … , 1026}
F	Number of 40' containers	{38, … , 387}
L	Number of containers for low weight class	{46, … , 424}
M	Number of containers for medium weight class	{50, … , 848}
H	Number of containers for high weight class	{4, … , 141}
D	Number of the destination port	{2 , 3}
ND	Number of containers for each destination	{40, … , 721}
LL	Lowest loading time (times are in 1/100 of minute)	{120, 200 }
HL	High loading times (times are in 1/100 of minute)	{270, 230}

The instances were designed under the following conditions:

- All bays of a vessel are of same size.
- The ship is empty [12].
- We assume that the number of containers to load on board is not greater than the number of available locations; this means that we are not concerned with the problem of selecting some containers to be loaded among all [11].
- We assume that the capacity of the ship related to the maximum weight and size available for all containers to be loaded is verified [11].

n Number of containers
T_t TEUs of the ship
D Number of the destination ports
b Number of even bays
r Number of rows (J)
s Number of tiers (K)
sH Number of tiers in the hold
sD Number of tiers in the deck
Q_1 Maximum horizontal equilibrium tolerance in tons
Q_2 Maximum vertical equilibrium tolerance in tons
MT Maximum vertical equilibrium tolerance in tons for 20' containers
MF Maximum vertical equilibrium tolerance in tons for 40' containers
$Container_1$ $Size_1$ $Destination_1$ $Weight_1$ $Weight\ Class_1$
...
$Container_n$ $Size_n$ $Destination_n$ $Weight_n$ $Weight\ Class_n$
Loading times in 1/100 *of minute* $(Row_r, Tier_1)...(Row_r, Tier_s)$
Loading times in 1/100 *of minute* $(Row_{r-2}, Tier_1)...(Row_{r-2}, Tier_s)$
....
Loading times in 1/100 *of minute* $(Row_{r-3}, Tier_1)...(Row_{r-3}, Tier_s)$
Loading times in 1/100 *of minute* $(Row_{r-1}, Tier_1)...(Row_{r-1}, Tier_s)$

Fig. 3. Description of the instance format

5.1 Description of Test Instance Sets

Two sets of instances (set A and set B) were generated randomly, using the ranges on Table 1; each generated instance is a file with the format in Figure 3. Ranges are given for those parameters that are variable within a set and were taken of specialized literature [12, 13].

The first column of Table 2 shows the particular parameters, while the second and third columns show parameter values for each set. According to the literature, the first set A of problem instances concerns a small size containership, and the second one was considered for a medium size.

Set A

The first set is formed by small size test instances. Table 3 reports the characteristics of the considered 10 instances, showing the total number of containers, both in TEU and absolute number (n), the number of 20' (T) and 40' (F) containers, the number of containers for three classes of weight (L: low, M: medium, H: high) and the partition of containers for each destination. Three classes of weight are considered, namely low (from 5 to 15 tons), medium (from 16 to 25 tons) and high containers (from 26 to 32 tons).

Table 4 shows the loading times for the small containership; in that case loading times, depending on the row and tier and growing from left side rows to the right ones and from highest tiers on the deck to the lowest ones in the hold, have been assumed in the literature [13].

Set B

A second set of 14 instances was considered for a medium size containership. Table 4 shows the characteristics of the medium size instances and in Table 5 the corresponding loading times as done for Tables 3 and 4, respectively.

Table 2. Parameters used for the generation of instance sets

Parameter	Set A	Set B
b	{6}	{11}
r	{4}	{12}
s	{5}	{11}
sH	{3}	{6}
sD	{2}	{5}
Q_1	{20, … , 30}	{20, … , 30}
Q_2	{40, … , 50}	{40, … , 50}
MT	{45 }	{45 }
MF	{66 }	{66 }
TEU	{138, … , 188}	{945, … , 1800}
n	{100, … , 150}	{715, … , 1413}
T	{62, … , 120}	{485, … , 1026}
F	{38, … , 50}	{230, … , 387}
L	{46, … , 70}	{215, … , 424}
M	{50, … , 78}	{429, … , 848}
H	{4, 5}	{71, … , 141}
D	{2 , 3}	{2 , 3}
ND	{40, … , 85}	{264, … , 721}
LL	{200}	{120}
HL	{270}	{330}

Table 3. Containers for the set A (Small size test instances)

Instance	TEU	*n*	Type (*n*)		Weight (*n*)			Destination (*n*)		
			T	*F*	*L*	*M*	*H*	1	2	3
1	138	100	62	38	46	50	4	47	53	—
2	165	120	75	45	52	64	4	55	65	—
3	170	130	90	40	60	66	4	62	68	—
4	175	130	85	45	58	68	4	62	68	—
5	180	140	100	40	62	74	4	61	79	—
6	180	150	120	30	70	76	4	65	85	—
7	185	130	75	55	60	66	4	62	68	—
8	185	140	95	45	58	78	4	65	75	—
9	185	140	95	45	62	73	5	50	40	50
10	188	148	108	40	68	76	4	50	50	48

Table 4. Loading times for the set A (small size test instances), the times are expressed in 1/100 of minute

Tier	*Row*			
	3	*1*	*2*	*4*
2	240	250	260	270
4	230	240	250	260
6	220	230	240	250
82	210	220	230	240
84	200	210	220	230

6 Experimental Results

In this section the performance of the Algorithm 1, the constructive loading heuristic CLH, is presented. First we describe the experimental environment.

6.1 Experimental Environment

The following configuration corresponds to the experimental conditions that are common to the test described.

- *Software:* Operating system Microsoft Windows 7 Home Premium; Java programming language, Java Platform, JDK 1.6; and integrated development, Eclipse 3.4.
- *Hardware:* Computer equipment with processor Intel (R) Core (TM) i5 CPU M430 2.27 GHz and RAM memory of 4 GB.

- *Instances:* There are 10 small size test instances (see Tables 2 to 4) and 14 medium size test instances (see Tables 2, 5 and 6); their characteristics are described in Section 5.1.

Table 5. Containers for the set B (Medium size test instances)

Instance	TEU	*n*	Type (*n*)		Weight (*n*)			Destination (*n*)		
			T	*F*	*L*	*M*	*H*	1	2	3
1	945	715	485	230	215	429	71	350	365	—
2	1022	762	502	260	228	458	76	374	388	—
3	1120	820	520	300	246	492	82	402	418	—
4	1218	898	578	320	270	541	87	442	456	—
5	1320	980	640	340	295	589	96	334	382	264
6	1380	1090	800	290	327	654	109	534	556	—
7	1386	984	582	402	296	590	98	483	501	—
8	1415	1069	723	346	321	642	106	363	396	310
9	1420	1060	700	360	318	636	106	520	540	—
10	1528	1202	876	326	361	721	120	589	613	—
11	1522	1138	754	384	341	683	114	392	438	308
12	1627	1215	803	412	365	729	121	600	615	—
13	1724	1331	938	393	400	799	132	454	517	360
14	1800	1413	1026	387	424	848	141	692	721	—

Table 6. Loading times for the set B (medium size test instances), the times are expressed in 1/100 of minute

Tier	*Row*											
	11	*9*	*7*	*5*	*3*	*1*	*2*	*4*	*6*	*8*	*10*	*12*
2	220	230	240	250	260	270	280	290	300	310	320	330
4	210	220	230	240	250	260	270	280	290	300	310	320
6	200	210	220	230	240	250	260	270	280	290	300	310
8	190	200	210	220	230	240	250	260	270	280	290	300
10	180	190	200	210	220	230	240	250	260	270	280	290
12	170	180	190	200	210	220	230	240	250	260	270	280
82	160	170	180	190	200	210	220	230	240	250	260	270
84	150	160	170	180	190	200	210	220	230	240	250	260
86	140	150	160	170	180	190	200	210	220	230	240	250
88	130	140	150	160	170	180	190	200	210	220	230	240
90	120	130	140	150	160	170	180	190	200	210	220	230

6.2 Performance Measurement of the Constructive Loading Heuristic

In this section we show experimentally the performance of the Algorithm 1, named CLH. Tables 7 and 8 show the results for the small and medium size test instances, respectively. The first column of each table shows the name of the instances of each dataset, the second one indicates the containers number to be loaded (n) and the third column shows the loaded containers (nC) by the CLH algorithm. Finally the last column indicates the stowage time for each instance, this stowage time is also determined by CLH.

Table 7. Performance of the CLH Algorithm for the set A (small size)

CLH Algorithm			
Instance	n	nC	L (1/100 of minute)
Instance_1.txt	100	100	23750
Instance_2.txt	120	120	28430
Instance_3.txt	130	130	30800
Instance_4.txt	130	130	30890
Instance_5.txt	140	118	27910
Instance_6.txt	150	145	34170
Instance_7.txt	130	130	30830
Instance_8.txt	140	138	32620
Instance_9.txt	140	140	33440
Instance_10.txt	148	148	35280
Percent	100%	97.8%	-

Table 8. Performance of the CLH Algorithm for the set B (medium size)

CLH Algorithm			
Instance	n	nC	L (1/100 of minute)
Instance_11.txt	715	715	164120
Instance_12.txt	762	670	152230
Instance_13.txt	820	820	189210
Instance_14.txt	898	892	204010
Instance_15.txt	980	934	216390
Instance_16.txt	1090	1027	232430
Instance_17.txt	984	953	217780

CLH Algorithm (Continuation)			
Instance	*n*	*nC*	*L* (1/100 of minute)
Instance_18.txt	1069	963	222270
Instance_19.txt	1060	1006	228450
Instance_20.txt	1202	1161	265110
Instance_21.txt	1138	1112	255680
Instance_22.txt	1215	1115	253150
Instance_23.txt	1331	1293	296540
Instance_24.txt	1413	1413	323830
Percent	100%	95.8%	

7 Conclusions

The Master Bay Planning Problem (MBPP) is one of the relevant problems involves in the efficient operation of ports. Nowadays, does not exist a test bed to compare algorithms for MBPP, mainly because real-data are confidential.

In order to contribute to the design of a suitable test bed, we propose an initial and simple test bed integrated by 14 instances, an instance generator, and a solution algorithm called HCL (Heuristic Constructive Loading). The test bed is available online at:
https://sth-se.diino.com/lauracruzreyes/MBPP/TestBed1.

The random generation scheme for MBPP is based on the selection of a parameters set. The parameters were obtained from established ranges from the specialized literature; theirs values characterize containers and vessel from real-life.

In the small and medium size test instances, HCL loads in vessel at least 95% of the total of the containers. This heuristic generates a solution without considering the equilibrium constraints. This type of solution can be used as an initial solution of other algorithms. We are developing a metaheuristic that takes advantages of HCL.

References

[1] Amil, C.: Historia de los puertos. Instituto Universitario de Estudios Marítimos (December 9, 2004), `http://www.udc.es/iuem`

[2] Rúa, C.: Los puertos en el transporte marítimo. Instituto de organización y control de sistemas industriales. Universidad de Cataluña, Barcelona (2006)

[3] Steenken, D., Voss, S., Stahlbock, R.: Container terminal operation and operations research – A classification and literature review. OR Spectrum 26(1), 3–49 (2004)

[4] Vacca, I., Salani, M., Bierlaire, M.: Optimization of operations in container terminals: hierarchical vs integrated approaches. European Journal of Operational Research (2010)

[5] Delgado, A., Jensen, R.M., Janstrup, K., Rose, T.H., Andersen, K.H.: A Constraint Programming Model for Fast Optimal Stowage of Container Vessel Bays. European Journal of Operational Research (2012)
[6] Meisel, F.: Seaside Operations Planning in Container Terminals. Contributions to Management Science. Physica Verlag Heidelberg (2009)
[7] Ambrosino, D., Sciomachen, A., Tanfani, E.: Stowing a Containership: The Master Bay Plan problem. Transportation Research Part A: Policy and Practice 38, 81–99 (2004)
[8] Fan, L., Low, M.Y.H., Ying, H.S., Jing, H.W., Min, Z., Aye, W.C.: Stowage Planning of Large Containership with Tradeoff between Crane Work-load Balance and Ship Stability. In: Proceedings of the International Multi-Conference of Engineers and Computer Scientists, vol. 3 (2010)
[9] Zeng, M., Low, M.Y.H., Hsu, W.J., Huang, S.Y., Liu, F., Win, C.A.: Automated stowage planning for large containerships with improved safety and stability. In: Proceedings of the 2010 Winter Simulation Conference, WSC 2010 (2010)
[10] Avriel, M., Penn, M., Shpirer, N.: Container ship Stowage Problem: Complexity and Connection to the Coloring of Circle Graphs. Discrete Applied Mathematics 103(1), 271–279 (2000)
[11] Ambrosino, D., Sciomachen, A., Tanfani, E.: A decomposition heuristics for the container ship stowage problem. Journal of Heuristics 12, 211–233 (2006)
[12] Ambrosino, D., Anghinolfi, D., Paolucci, M., Sciomachen, A.: An experimental comparison of different heuristics for the master bay plan problem. Experimental Algorithms, 314–325 (2010)
[13] Ambrosino, D., Anghinolfi, D., Paolucci, M., Sciomachen, A.: A new three-step heuristic for the master bay plan problem. Maritime Economics and Logistics 11(1), 98–120 (2009)

Small Hydroponics Garden Improved Using Cultural Algorithms

Alberto Ochoa-Zezzatti[1], Ruben Jaramillo[2], Sandra Bustillos[1], Nemesio Castillo[1], José Martínez[3], Samantha Bastiani[3], and Victoria Ruíz[3]

[1] Juarez City University, México
[2] LAPEM, México
[3] Instituto Tecnológico de Ciudad Madero, México
alberto.ochoa@uacj.mx

Abstract. The paper discusses a research related with the innovative sense of using Decision Support System based on a Bioinspired Algorithm related with an Agribussiness and hobby too about Hydroponics, to determine the correct and adequate selection of seeds in small spaces to build scenarios of location in the future to analyze the way to improve Mexican families' economy, this research which permits select a specific number of seeds to cultivate each one of 25 different seeds –In a time horizon of a seasonal cultivate (approximately three months)-, these seeds are evaluated from a information repository with data from another suceesful hydroponics systems. Each harvest was analyzed to built their cost-benefit during different times and scenarios and determine the viability of cultivate in the time horizon using a formal methodology based on Bioinspired Algorithms. The group of 25 seeds cultivated by "Leguizamo Povedano Cooperative" is characterized and analyzed by obtain the most representative and sucessful future scenario to determine the quantity of seeds cultivated which try to improve the limited resources and the perspectives of determine the correct selection to stablishment a signifcately life day. A case of study is presented regarding to the proposal horizons using data obtained from the Repository of the Cooperative. The intention of the present research is to apply the computational properties; in this case of established a Model of Hydropoics cultivate in a Cooperative. In addition, we analyzed the selection and location of location to cultivate a specific seed using a similarity model to locate this. The sample of study allowed analyzing the individual features of each harvest with the emulation from set matching features (commercialization, climate, in others). By means of this is possible to predict the best location to cultivte.

Keywords: Cultural Algorithms, Pattern Recognition and Decision Support System.

1 Introduction

Researchers discovered many features of plant's physiology in the nineteenth century, for example that plants absorb essential minerals through inorganic ions

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 409–416.
springerlink.com

dissolved in water. In a normal environment under natural conditions, the soil acts as a mineral nutrient reservoir but the soil itself is not essential for plant growth. For this reason when the mineral nutrients in the soil dissolve in water, plant roots are able to absorb them. And for this action when the mineral nutrients are introduced into the water supply plant and the soil is not required for the plant to thrive, see Figure 1. Almost of any terrestrial plant can grow with hydroponics, but some may do better than others. The Biological Hydroponics is also a standard technique in biological research, education and a popular pastime.

Today this activity is reaching a boom in countries where conditions are adverse for agriculture, combining hydroponics with good management of the greenhouse yields reach much higher than those obtained in open crops. It is a simple, clean and inexpensive to produce fast-growing plants and generally rich in nutrients. With this technique of small-scale farming using the resources that people have on hand, as waste materials, unused spaces, and free time.

During 2011 we can say that hydroponics or soilless commercial standards and has achieved some foods, ornamentals and snuff young plants are grown in this way for various reasons having to do with the lack of suitable soils, for soils contaminated by microorganisms that cause disease to plants or by using groundwater that degraded the quality of these soils. Hydroponics requires advanced knowledge for those intending to carry out a cash crop. By not using ground and there is no cushion or buffer the effect which provides an agricultural soil. It also has several problems with oxygenation of the roots and is not a thing as clean when done on a commercial scale. For people who want to enjoy free time for research, demonstration to students about the essentiality of certain chemical elements, even for those who want to grow in a container, a small tub, to grow in spaceships or crops on a large scale, will present various levels of complexity especially if it is to be an economic activity and have low environmental impact. The classification of hydroponics has evolved more recently to form open or closed depending on whether the effluent dumped or reuse the nutrient solution as a form of environmental protection and greater economy in use.

2 Cultural Algorithms

The initial development of Cultural Algorithms (CAs) can be attributed to Reynolds [12] this approach is a complement to the metaphor used by evolutionary algorithms, which had focused on the concepts of genetics and natural evolution. Cultural algorithms are based on the theories of anthropologists, sociologists and archaeologists, who have tried to model the evolution as a process of cultural evolution [5]. The belief space characterizes CAs as evolutionary algorithms, which are used to store the acquired knowledge from previous generations. The information in this space must be accessible to any individual, who may use it to change their behavior and their respective proposed solution. To join the belief space and the population is necessary to establish a communication protocol, which dictates rules of the type of information to be exchanged between spaces. This protocol defines the acceptance and influence functions. The acceptance function is responsible for accepting the information or the experience that individuals have obtained

in the current generation and transport into the belief space. On the other hand, the influence function is responsible for "influencing" variation operators (e.g. crossover and mutation in the case of genetic algorithms). This means that this function set some kind of pressure on resultant individuals from the application of variation operators to reach the desirable behavior, also away from undesirable results, always according to information stored in the belief space.

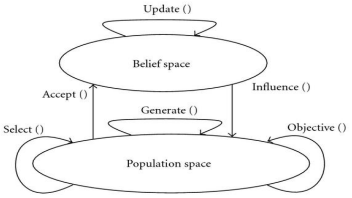


Fig. 1. Different Spaces employed by the Cultural Algorithm

Figure 1 presents the interaction between the belief space and population space. The population space works similar to that of an evolutionary algorithm, i.e. the population consists of a set of individuals where each has an independent feature used to determine their suitability (fitness). The interaction between the two spaces makes the cultural algorithm increases the complexity in the development and computation of the evolutionary algorithm. Below we show the general pseudocode of a cultural algorithm.

```
Begin
  t=0;
  Initialize POP(t); // Initialization of population
  Initialize BLF(t); // Initialization of believing space
  Evaluate POP(t);
  While (Do not condition of term t=t+1)
    Vote (BLF (t), Accept (POP(t))));
    Adjust (BLF (t));
    Evolve(POP(t), Influence(BLF(t)));
    t = t +1;
    Select POP(t) from POP(t-1);
  End While
End
```

3 Selection and Characterization of Seeds in a Hydroponics Project

Hydropnics require of many features to determine the quantity, distributions of seeds and location to improve the commercialization of each specialized space as the shown in the Figure 2.

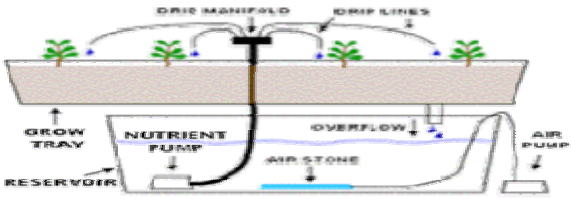


Fig. 2. Components of a Hydroponics System to cultivate different seeds

Table 1. Specifications related with the time to cultvate diverse species

Specie	Weather	Row spacing in cm	Distance between plants in cm	Depth	Germination time day	Harvest time day
Chard	Cold	20	20	1.5	21	70
Celery	Cold	20	20	0.3	33	90
Broccoli	Cold	30	25	1	21	75
Onion	Cold	12	10	0.5	33	80
Chives	Cold	10	8	0.5	33	55
Chile Pepper	Warm	35	30	1	37	80
Coriander	Warm	10	5	2	17	60
Spinach	Cold	17	17	2	20	75
Bean	Warm	15	15	1.5	5	100
Güicoy	Warm	50	40	4	7	120
Little Güicoy	Warm	50	40	3	7	90
Lettuce	Temperate	17	17	5	21	50
Melon	Warm	30	30	3	6	90
Turnip	Temperate	10	10	1	5	80
White Turnip	Cold	10	8	2	17	75
Cucumber	Warm	30	30	3	5	70
Parsley	Cold	15	12	0.5	23	70
Leek	Temperate	10	10	0.5	37	80
Reed Radish	Temperate	8	5	2	4	30
Beet	Temperate	15	10	1	23	85
Rendelí	Temperate	15	10	3	10	120
Cabbage	Cold	30	25	1	33	90
Watermelon	Warm	40	40	4	8	90
Tomato	Warm	35	30	1	20	65
Carrot	Temperate	8	10	3	18	120

We try to improve the next equation 1 using the table 1, in which is P as the productivity reached for cultivate a seed and its respective economical impact according to diverse features to cultivate.

$$P = NL\ [\ IA + (CO - C)^{PS}\ ] \pm CBS \qquad (1)$$

In where:

NL = Number of lines cultivated with the same seed, the unit is a line with until 12 issues.

CO = Commercialization of the harvest when its offers in diverse kind of markets.

C =Represents if the seed can be cultivate during all year, for example the coriander.

IA = Inputs associated with improve of cultivate to produce more with less money.

PS = Sustainability is associated with the reuse of inputs to more of one harvest in the time.

CBS = Social Benefits-Cost related with each seed, in the case to Tomato is associated with improve of healthy in the customers.

Table 2. Multivariable analysis with the information related with the harvest, including index of commercialization, climate and cost-benefit to cultivate

Seed	Cultivate Lines	Commercialization	Climate	Water Consumption	Cost-Benefit
Chard	9	0.778	0.734	0.674	0.814
Celery	13	0.912	0.673	0.728	0.972
Broccoli	6	0.632	0.589	0.682	0.915
Onion	10	0.837	0.715	0.629	0.885
Chives	14	0.944	0.763	0.725	0.723
Chile Pepper	5	0.782	0.488	0.562	0.811
Coriander	4	0.694	0.695	0.679	0.796
Spinach	9	0.778	0.734	0.674	0.814
Bean	13	0.912	0.673	0.728	0.972
Güicoy	6	0.632	0.589	0.682	0.915
Little Güicoy	10	0.837	0.715	0.629	0.885
Lettuce	14	0.944	0.763	0.725	0.723
Melon	5	0.782	0.488	0.562	0.811
Turnip	4	0.694	0.695	0.679	0.796
White Turnip	12	0.726	0.823	0.825	0.837
Cucumber	7	0.926	0.875	0.714	0.862
Parsley	9	0.857	0.742	0.678	0.783
Leek	11	0.796	0.816	0.785	0.827
Red Radish	8	0.814	0.765	0.863	0.799
Beet	14	0.795	0.811	0.835	0.847
Rendelí	5	0.747	0.838	0.842	0.817
Cabbage	8	0.816	0.794	0.803	0.858
Watermelon	9	0.947	0.836	0.828	0.807
Tomato	12	0.877	0.819	0.842	0.805
Carrot	7	0.954	0.797	0.783	0.912

4 Multiple Matching

The multiple matching is a series of seven evaluations according to different combinations of seeds and a batch of 50 runs under different scenarios. In the evaluation phase economics specifications with more similarities will be given a preference, and then these aspects will be selected to compete. Each seed makes a compromise and participates in exactly seven of these evaluations. Seeds must be ranked according to their customers' preferences after tournaments end once the final list of multiple matching is evaluated. The hybrid algorithm sets the right for customers to evaluate a batch according to the organizational needs and the seeds for each comparison assign the seeds list before a new cycle begins. Each evaluation will have all the seeds playing over a schedule of seventeen runs. The hybrid algorithm will be scheduled to set the timing for the comparison of different similarities using a round of multiple matching analyses based in the commercialization assigned to a seed. Then, seeds that qualify for selection in a Model will be

chosen on the following prioritized basis. For the first cycle of similarity, all seeds in the Repository (ie. Beet or Leek seeds) will be invited to participate for different comparisons. Given the organization for each seed and the matches for each round in the algorithm, seeds are asked to state their participation for its evaluation in each of the series. In case any of these seeds decline to participate in the series, the algorithm may nominate one seed to be set as a replacement, and this seed has to be rated amongst the top seeds in the Repository. Based on an average calculation of two decimal places, the rating list in the series of comparisons, before starting a new cycle, ten qualifiers will be selected (excluding the seven seeds that will be compared in the matches). In case seeds have the same average rating, the number of similarities set for the match will be used to determine its ranking. To ensure an active participation in the future, a minimum of twenty-five games are recommended for the four included rating lists and before the main rating list. When a seed does not accept to play into a Multiple Matching series, then the selection process uses the average rating plus number of games played during the rating period. The algorithm repeats this process until reaching the required qualifiers of the Multiple Matching series.

5 Experimentation

In order to obtain the most efficient arrangement of seeds, we developed a cluster for storing the data of each of the representative individuals for each seed. The narrative guide is made with the purpose of distributing an optimal form for each the evaluated seeds [9]. The main experiment consisted in implementing seeds in the Cultural Algorithm, with 500 agents and 200 beliefs into the belief space. The stop condition is reached after 50 runs; this allowed generating the best selection of each kind and their possible location in a specific Model. A location is obtained after comparing the different cultural and economical similarities of each seed and the evaluation of the Multiple Matching Model as in [10]. The vector of weights employed for the fitness function is W_i=[0.6, 0.7, 0.8, 0.5, 0.6], which respectively represents the importance of the particular attributes: Cultivate lines, Commercialization, Climate, Water Consumption and Cost-Benefit. Then, the cultural algorithm will select the specific location of each seed based on the attributes similarity. Each attribute is represented by a discrete value from 0 to 5, where 0 means absence and 5 the highest value of the attribute. The experiment design consists of an orthogonal array test with interactions amongst the attribute variables; these variables are studied within a location range (1 to 400) specific to a coordinates x and y. The orthogonal array is L-N(2**5), in other words, 5 times the N executions. The value of N is defined by the combination of the 5 possible values of the variables, also the values in the location range. In Table 3 we list some possible scenarios as the result of combining the values of the attributes and the specific location to represent a specific issue (seed). The results permit us to analyze the effect of the variables in the location selection of all the possible combinations of values.

Table 3. The orthogonal array test

Cultivate Lines	Commercialization	Climate	Water Consumption	Cost-Benefit
4	1	2	2	3
3	1	2	2	3
2	1	3	2	4
5	1	3	2	5

The use of the orthogonal array test facilitates the reorganization of the different attributes. Also the array aids to specify the best possibilities to adequate correct solutions (locations) for each seed. Different attributes were used to identify the real possibilities of improving a seed set in a particular environment, and to specify the correlations with other seeds (see Figure 3). The locations will be chose based on the orthogonal test array.

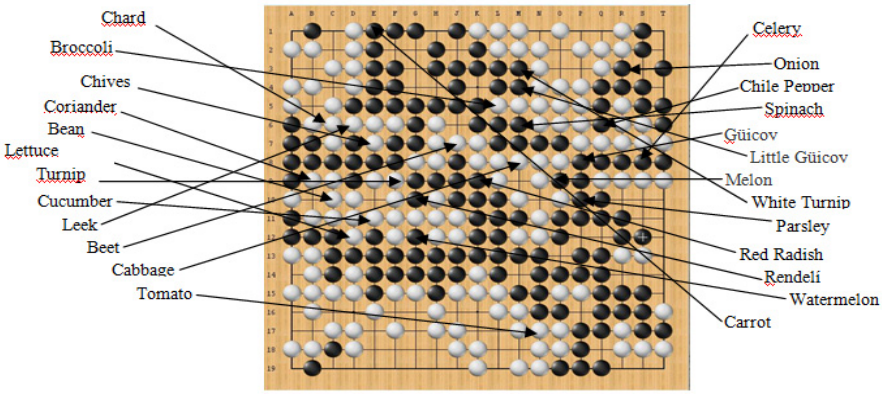


Fig. 3. Location of seeds to different harvest in a small garden based on Hydroponics The white dots represents harvest with less water necessities and the black dots represent harvest with more high water necessities.

6 Conclusions

After our experiments we were able to remark the importance of the diversity of the established economical patterns for each seed. These patterns represent a unique form of adaptive behavior that solves a computational problem that does not make clusters of the seeds. The resultant configurations can be metaphorically related to the knowledge of the behavior of the community with respect to an optimization problem (to culturally select 25 similar seeds [4]). Our implementation related each of the seeds to a specific a location quadrant. The Narrative guide, allowed us to identify changes in time related to one or another seed. Here, we show that the use of cultural algorithms substantially increased the understanding in

obtaining the “best paradigm”. This after the classification of agent communities was made based on a relation that keeps their attributes. Therefore, we realize that the concept of "negotiation" exists based on determining the acceptance function to propose an alternative location for the rest of the seeds [8]. For further implementations we intend to analyze the level and degree of cognitive knowledge for each seed. Additionally, this may help to understand true similarities that share different seeds based in the characteristics to be clustered and also to keep their own biological identity. In a related work [7], it has been demonstrated that small variations go beyond phenotypic characteristics and are mainly associate to tastes and related characteristics developed through the time. On the other hand, CAs can be used in the Evolutionary Robotic field where social interaction and decision is needed, for example in the training phase described in [11], and to organize group of robots for collaborative tasks. Another future work using CAs is related to the distribution of workgroups, social groups or social networking. Finally, CAs can be used in pattern recognition in a social database, for example: fashion styling and criminal behavior.

Acknowledgements. We thank the community of Computational Optimization that for a decade has researched about novel ways to display social topics.

References

1. Desmond, A., Moore, J.: Darwin – la vida de un evolucionista atormentado. Generación Editorial, São Paulo, Brazil (1995)
2. Ochoa, A., et al.: Baharastar – Simulador de Algoritmos Culturales para la Minería de Datos Social. In: Proceedings of COMCEV 2007 (2007)
3. Ochoa, A., et al.: Dyoram’s Representation Using a Mosaic Image. The International Journal of Virtual Reality
4. Memory Alpha (2009), `http://en.memory-alpha.org`
5. Callogerodóttir, Z., Ochoa, A.: Optimization Problem Solving using Predator/Prey Games and Cultural Algorithms. In: NDAM 2003, Reykiavik, Iceland (2007)
6. Hué, T., et al.: The Emergence of Social Network Hierarchy Using Cultural Algorithms. In: VLDB 2006, Seoul, Korea (2006)
7. Vukčević, I., Ochoa, A.: Similar cultural relationships in Montenegro. In: JASSS 2005, England (2005)
8. Dennis, Z.: Culture and Organizations. McGraw-Hill, London (1991)
9. Ponce, J., et al.: Data Mining and Knowledge Discovery in Real Life Applications. Ponce, J., Karahoca, A. (eds.), vol. 438. I-Tech, Vienna (2009) ISBN 978-3-902613-53-0
10. Ustaoglu, Y.: Simulating the behavior of a minority in Turkish Society. In: ASNA 2009, Zurigo, Svizerra (2009)
11. Nolfi, S., Floreano, D.: Evolutionary Robotic: The Biology, Intelligence, and Technology of Self-Organization Machines. MIT Press, MA (2000)
12. Reynolds, R.G., Sverdlik, W.: Problem Solving Using Cultural Al-gorithms. In: International Conference on Evolutionary Computation, pp. 645–650 (1994)

Handling of Synergy into an Algorithm for Project Portfolio Selection

Gilberto Rivera[1], Claudia G. Gómez[1], Eduardo R. Fernández[2], Laura Cruz[1], Oscar Castillo[3], and Samantha S. Bastiani[1]

[1] Madero Institute of Technology, México
riveragil@gmail.com, {cggs71,b_shulamith}@hotmail.com, lcruzr@prodigy.net.mx
[2] Sinaloa Autonomous University, México
eddyf@uas.uasnet.mx
[3] Tijuana Institute of Technology, México
ocastillo@tectijuana.mx

Abstract. Public and private organizations continuously invest on projects. With a number of candidate projects bigger than those ones that can be funded, the organization faces the problem of selecting a portfolio of projects that maximizes the expected benefits. The selection is made on the evaluation of project groups and not on the evaluation of single projects. However, there is a factor that must be taken account, since it can significantly change the evaluation of groups: synergy. This is that two or more projects are complemented in a way that generates an additional benefit to they already own individually. Redundancy, a special case of synergy, occurs when two or more projects cannot be financed simultaneously. Both features add complexity to the evaluation of project groups. This article presents an evaluation of the two most used alternatives for handling synergy, in order to incorporate it into an ant-colony metaheuristic for solving project portfolio selection.

1 Introduction

One of the most important decisions in organizations is to select which projects will be funded for ensuring their growth [2]. This decision is made by one or several people, commonly called the *Decision Maker* (DM). In resources allocating for social projects, the monetary benefits are not the main criteria for selecting projects. Other objectives that measure the social profit are more important. DMs are responsible for selecting a portfolio depending on:

1. Values in the portfolio objectives which have to assure a minimal acceptability level.
2. His/her criteria that shall depend on several personal subjective issues, among which can be cited: his/her belief, experience, policies to follow according to the organization and personal ethics.

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 417–430.
springerlink.com

Typically the *Social Portfolio Problem* (SPP) is modeled as a multicriteria problem [3, 15, 20], and has been solved by means of algorithms that commonly search a Pareto frontier approximation. This guarantees a good level of acceptability (in terms of objective values) in the final solution set. However, the cardinal of this set is often too long; this complicates the selecting process made by the DM. A typical DM will have problems for processing more than 5-9 pieces of knowledge [19], so he/she is unable to select satisfactorily one portfolio even from a relatively-short set of solutions with more than five objectives. To ease the decision making process, the reducing the final solution set is critical. But, the reduced set presented to the DM has to match the preferences of him/her.

Three approaches are commonly used for identifying solutions that match the DM preferences [17]:

1. *Including the DM's preferences after optimization process*, searching a uniform Pareto frontier but presenting only the solutions that better match his/her preferences.
2. *Including the DM's preferences before optimization process*, guiding the search to privileged Pareto frontier zones that better match the DM's preferences.
3. *Interacting with the DM progressively during optimization process*.

In this paper we use the Fernández's model for identifying the DM's preferences about the portfolios. It is a priori articulation of preferences by creating fuzzy outranking relations that allow identifying which solutions match better the DM's preferences (these solutions are called the best compromise).

One feature commonly presents in SPP is synergy, which involves that there are interrelated groups of projects. Such interrelation provokes an additional benefit to organization objectives. So, when the whole synergetic group is supported the benefits are bigger than the sum of benefits of the same projects taken independently.

Basically, there exist two forms when the synergy may be neglected without major problems, when:

1. the interaction is too weak.
2. the interactions affect all the projects in a similar way and constantly.

But in other cases, when no previous characteristic is present, then synergy and redundancy among projects become relevant matters for decision making on project portfolio.

When the DM identifies important synergetic effects, finding an appropriate way for handling synergy and redundancy is not a simple task. Commonly, it is done by means of inclusion of an artificial project, that groups the synergetic set and whose benefits already include the synergy gain. But, extra redundancy relations must be included, between artificial project and each one of the original projects with synergy.

Although previous works on the formation of project portfolio with synergy have been proposed [3, 6, 18, 20], it has made evident the lack of methods that also include the decision maker's preferences. This work presents an Ant Colony

Optimization (ACO) Algorithm that searches a Pareto frontier subset that matches the DM's criteria for selecting a portfolio.

This paper is structured as follows: we formalize the problem in Section 2; in Section 3 we analyze the metaheuristic procedures used to solve it; Section 4 develops our proposal; Section 5 presents the instances and discusses the results; and final section offers the conclusions.

2 Background

In order to establish the theoretical elements needed to pose our contribution, we begin with a formal definition for social portfolio problem, afterward we shall focus on the elements related to the synergy among projects, types and issues inherent, and we conclude with a description of the preference model used in this proposal.

2.1 Social Portfolio Problem

From a set of all proposed projects competing for resources, denoted as X, a portfolio may be defined as a subset of them. Typically is modeled as a binary vector $x = \{x_1, x_2, x_3, \ldots, x_N\}$, where N is the total of project proposals and the variables xi indicate whether the project i is included in the portfolio (if $x_i = 1$) or not (if $x_i = 0$).

There is a total budget that the organization is willing to invest, which is denoted as B, and each project proposal i has an associated cost to carry it out, denoted as c_i, it is understood that the formation of any portfolio x is subject to the constraint:

$$\left(\sum_{i=1}^{N} x_i c_i \right) \leq B \tag{1}$$

It is also common that projects are associated with some form of grouping that influences the decision of the DM. For example in a private company, projects may be associated to departments (e.g. marketing, sales, production or human resources), in this case a balanced DM would try to ensure a minimum of the total budget for each, and would prevent either of them unjustifiably monopolizes most of the budget. For public projects, such groups can be associated with geographical divisions, social groups or public interest areas (e.g. social security, education or public health). In general terms, this form for grouping will be called "areas".

Let L_i and U_i be respectively the minimum and maximum budget that an area i can get ($L_i \leq U_i \leq B$). The area for a project i may be defined as a_i. For each area j, any portfolio x must satisfy the constraint:

$$L_i \leq \sum_{i=1}^{N} x_i g_{i,j} \leq L_u \tag{2}$$

where $g_{i;j}$ may be defined as follows:

$$g_{i,j} = \begin{cases} c_i & \text{if } a_i = j \\ 0 & \text{Otherwise} \end{cases} \tag{3}$$

Suitable values for L and U and the total of areas depend on the problem characteristics, the DM criteria and the organizational policies.

In private organizations is more natural to represent the benefits of projects in monetary units than in public or social organizations. There are also ethical and moral reasons preventing the realization of a monetary equivalent of objectives of public organizations, for example, can raise questions such as: what is the value of a person's health?, and for a child's education?, and for a young man's postgraduate studies?. Due to these reasons, the selection of social projects is done on the multiobjective evaluation of portfolios, and not on a single monetary equivalent of objectives.

So, the benefits for a project i evaluated on p objectives may be modeled as $f(i) = \{f_1(i), f_2(i), f_3(i), \ldots, f_p(i)\}$. And therefore, the quality of a portfolio x is expressed as

$$z(x) = \{z_1(x), z_2(x), z_3(x), \ldots.., z_p(x)\} \tag{4}$$

where $z_j(x)$ is defined as

$$z_j(x) = \sum_{i}^{N} x_i f_j(i) \tag{5}$$

Social Portfolio Problem (SPP) consists in identifying one or more portfolios that solve

$$\arg\max_{x \in X} \{z(x)\}, \tag{6}$$

subject to the constraints expressed in Equations 1 and 2. In this case, the maximization concept is based on Pareto efficiency. In this case, the dimension of its solution space is 2^N, and if taken into account additional considerations such as synergy, partial support, project scheduling or risky conditions, the problem complexity and the number of possible solutions tend to increase.

2.2 Synergy Among Projects

Interactions between projects may affect the evaluation of portfolios. Among synergy types to be found are:

1. *Synergy positive on objective values.* When two or more projects are complemented for increasing profits. For example, a Project *A*: creating a recreational park, and a Project *B*: Paving a road. Each one individually benefits certain amount of people, but if the recreation park (project *A*) is on the road to be

paved (project *B*) both increase their value. The park becomes more accessible to people, and increase the amount of persons that use the road.
2. *Negative synergy on objective values:* The benefit of two or more projects decreases when are together. For example, a project *A* that benefits to 100 people and a Project *B* benefiting 200 people. But if there are 50 people in common between both projects, the benefit of supporting both projects is 250 people instead of 300 as expected.
3. *Redundancy:* Two projects can not be supported simultaneously. For example, a Project *A*: Building a hospital, and a Project *B*: building a school. But if the hospital needs to be built on the same ground that the school, only one of them can receive budget.
4. *Decrease in the cost:* Two projects decrease the costs if both are supported simultaneously. For example, a project *A*: has a total cost of 250 monetary units, of which uses 100 to buy expensive computer equipment, and a Project *B*: it has a cost of 200 monetary units and also need to use similar equipment. If possible, and both projects have no problem sharing the common resource, the cost of financing both resources is 350 monetary units instead of 450 as expected.

Addressing the synergy among projects is a task that is not considered trivial in the forming of social portfolios [14]. Handling of redundancy involves modifying the feasibility conditions of a portfolio. While the handling of synergy that changes the objective values has been handled typically in two forms:

1. Doing an adaptation in the multiobjective function (Equation 4).
2. Adding an artificial project that represents the synergetic set and adding all necessary redundancy relations to prevent the artificial project and any original synergetic projects in this set appear together.

Synergy have been addressed in the social portfolio problem [3, 6, 20], but few research works have also considered the use of a model of preferences [18].

2.3 Preference Model

Although finding Pareto front is important [4, 9], the problem does not is completely solved [5, 11, 12, 13, 23]. Now the DM will choose which portfolio will be selected to receive the budget. Presenting a large set of solutions will complicate the decision process of the DM, even if the solutions found belong to the Pareto front, moreover finding a single function that matches the DM's preference, such as a weighted sum, is not simple and nor guaranteed for real problems [21, 22].

Fernández [10] proposed an *a priori* articulation to guide the search process to areas of the Pareto front that best match the preferences of the DM. Thereby reducing the amount of solutions presented to the DM. The preference model is based on the value of $\sigma(x,y)$, which measures the degree of credibility of the statement "x is at least as good as y", where x and y are portfolios in SPP. To calculate the value of $\sigma(x,y)$ can be used several proven methods, including the ELECTRE [1] and PROMETHEE [8] methods.

Considering the parameters λ, β, and ε ($0 \leq \varepsilon \leq \beta \leq \lambda$), the model identifies one of the following relationships for each pair of portfolios (x, y):

1. $x\mathbf{I}y$, *Indifference:* Corresponds to existence of reasons that justifies equivalence between x and y. $x\mathbf{I}y$ is justified if both conditions are held:
   a. $\sigma(x,y) \geq \lambda \wedge \sigma(y,x) \geq \lambda$.
   b. $|\sigma(x,y) - \sigma(y,x)| \leq \varepsilon$.
2. $x\mathbf{P}y$, *Strict preference*: Represents the existence of reasons that justify significant preference in favor of x. The statement "x is strictly preferred to y" is hold if at least one of following sentences is true:
   a. x dominates y.
   b. $\sigma(x,y) \geq \lambda \wedge \sigma(y,x) < 0.5$.
   c. $\sigma(x,y) \geq \lambda \wedge (0.5 \leq \sigma(y,x) \leq \lambda) \wedge (\sigma(x,y) - \sigma(y,x)) \geq \beta$.
3. $x\mathbf{Q}y$, *Weak preference*: Corresponds to the existence of reasons in favor of x over y, but that are not sufficient to justify strict preference. $x\mathbf{Q}y$ is modeled by the conjunction of the following propositions:
   a. $\sigma(x,y) \geq \lambda \wedge \sigma(x,y) \geq \sigma(y,x)$.
   b. $\neg x\mathbf{P}y$.
   c. $\neg x\mathbf{I}y$.
4. $x\mathbf{R}y$, *Incomparability*: There is no reason that justifies preference or equivalence in favor of x or y. $x\mathbf{R}y$ is the conjunction of
   a. $\sigma(x,y) < 0.5$.
   b. $\sigma(y,x) < 0.5$.
5. $x\mathbf{K}y$, *K-Preference*: There exist reasons for justifying preference in favor of x, but with no significant division established between $x\mathbf{P}y$ and $x\mathbf{R}y$. $x\mathbf{K}y$ is held if the conjunction of following propositions is true:
   a. $0.5 \leq \sigma(x,y) < \lambda$.
   b. $\sigma(y,x) < 0.5$.
   c. $\sigma(x,y) - \sigma(y,x) > \beta/2$

Let O be the set of alternative portfolios found by an algorithm, we can calculate the number of portfolios that are strictly preferred to each $x \in O$ as follows:

$$S(O,x) = \{y \in O \mid y\mathbf{P}x\}, \tag{7}$$

and, from this, the *non-strictly outranked frontier* in O is defined as $NS(O) = \{x \in O \mid S(O,x) = \emptyset\}$. The best compromise is in $NS(O)$. However, it is possible to identify more "weak" preference relations in the set, defining:

$$W(O,x) = \{y \in O \mid y\mathbf{Q}x \vee y\mathbf{K}x\}. \tag{8}$$

Using Equation 8, the *non-weakly outranked frontier* in O is defined as $NW(O) = \{x \in O \mid W(O,x) = \emptyset\}$. The best compromise is in $NS(O)$ and $NW(O)$. Even within the set $NW(O)$, a third measure of preference can be established. Consider the preference flow for an alternative x as:

$$F_n(O,x) = \sum_{y \in O - \{x\}} \sigma(x,y) - \sigma(y,x), \tag{9}$$

note that $F_n(x) > F_n(y)$ denotes a kind of preference relation x on y. Similar to $W(O,x)$ and $S(O,x)$ sets, $F(O,x)$ may be defined as:

$$F(O,x) = \{y \in O \mid F_n(y) > F_n(x)\}, \tag{10}$$

and the *non-net-flow outranked frontier* in O is defined as $NF(O) = \{x \in O \mid F(O,x) = \emptyset\}$. A solution x with $|F(O,x)|=0$ should match with the preferences better than another with a higher value. Thus, identification of the best compromise can be expressed as:

$$x^* = \arg\min_{x \in O} \{|S(O,x)|, |W(O,x)|, |F(O,x)|\}. \tag{11}$$

In Equation 11, the minimization is lexicographical; this involves the minimization of the three objectives in order.

3 Related Works

Several algorithms have been developed to solve SPP with synergy, which range from deterministic heuristic algorithms to more sophisticated metaheuristic techniques. The options in the handling of synergy have been basically two: 1) Modifying the multiobjective evaluation function and 2) Adding artificial projects representing the synergic group, and then add redundancy relations that prevent the artificial project appears at the same portfolio that some project of that group.

P-ACO (*Pareto Ant Colony Optimization*) [6] is an algorithm based on the known metaheuristic of Ant Colony to generate the Pareto front with the most efficient portfolios. Each ant generates a candidate portfolio, and the amount of pheromone deposited by such ant is inversely proportional to the number of solutions that dominate it. The algorithm stores the solutions that have never been dominated, which form an approximation of the Pareto front. P-ACO is able to deal with the synergy between projects, salient characteristic in the algorithm. The algorithm only considers the handling of synergy between groups of two projects. No consideration is made about the DM's preferences and the synergy is handling by modifying the multiobjective function.

APS (*Adaptive Sampling Pareto*) [20] is an algorithm that uses a sampling approach for estimation of the Pareto front, based on Monte Carlo simulation method. APS allows the portfolios selection considering synergy between projects. APS is compared with P-ACO over a hundred instances of test, outperforms it. No preference model is considered and the synergy is handling by modifying the multiobjective function.

RPM (*Robust Portfolio Modeling*) [18] is a support tool for multiobjective decision making for project selection. RPM has an algorithm that develops a comprehensive search with dynamic programming to find all optimal portfolios, which then are indexed according to the DM's criteria through a weighted sum of the objectives. The synergy is handling by means of adding artificial projects and extra redundancy relations.

SS-PPS (*Scatter Search for Project Portfolio Selection*) [3] is an algorithm based on scatter search that addresses the problem of the portfolio selection.

SS-PPS has basically two stages, the first generates a set of efficient starting points using Tabu Search, and the second stage improves the initial set by Scatter Search. No consideration is made about the DM's preferences and the synergy is handling by modifying the multiobjective function.

4 Proposed Algorithm

Our algorithm, ACOS-SPP (*Ant-Colony Outranking System for SPP*) is an ant colony algorithm for solve social portfolio problem. It takes ideas from Dorigo's ACS [7], but incorporates changes in the constructing and updating phases for taking into account the preference model. As most ACO algorithms, ACOS-SPP has two phases: constructing and updating.

During ACOS-SPP constructing phase, each ant selects a portfolio, choosing project by project until the budget is over. The way for selecting next project to portfolio is named *selecting rule*.

Afterwards, the updating phase is performed. All portfolios are evaluated by the preference model (Section 2.2), obtaining the three non-outranked fronts: NS, NW and NF, then pheromone is evaporated and each ant drops pheromone according to non-outranked front of its portfolio.

In this section is described the pheromone structure, and constructing and updating phases for ACOS-SPP.

4.1 Pheromone Structure

The pheromone is modeled as a two-dimensional matrix with a size $N \times N$, where N is the total of projects, so τ has two entries. $\tau_{i,j}$ is the preference of having the projects i and j in the same portfolio. The values of τ is in (0, 1], being one the initial value. The pheromone is decreased with a constant factor, ρ, once at the end of each iteration.

4.2 Constructing Phase

The next project selection depends on the selection rule:

$$j_x = \begin{cases} \arg\max_{i \in X} \{\Omega(x,i)\} & \text{if } \wp \le \alpha_1, \\ \mathcal{A}_{i \in X} \{\Omega(x,i)\} & \text{if } \alpha_1 < \wp \le \alpha_2, \\ \ell_{i \in X} & \text{Otherwise,} \end{cases} \tag{12}$$

where j_x is the next project to be incorporated in portfolio x, X is the project list, $\Omega(x,i)$ is a function that evaluates the expected benefit to incorporate i to portfolio x, $\mathcal{A}$ is the roulette selection function based on $\Omega(x,i)$, ℓ is a function that randomly selects an available project, and $\wp$ is a pseudo-random number between zero and one with uniform distribution. The benefit function $\Omega(x,i)$ may be defined as

$$\Omega(x,i) = w \bullet \eta_i + (1-w)\left(\frac{\sum_{j \in x} \tau_{i,j}}{|x|}\right), \tag{13}$$

where x is the current portfolio, i is the candidate project for being added to x, η_i is known as local knowledge, and is a measure of benefit of the project i, $\tau_{i,j}$ is the pheromone between projects i and j, and w is a weight factor between local knowledge and the pheromone. The local knowledge η_i may defined as

$$\eta_i = \frac{\left(\frac{1}{c_i}\right)\sum_{j=0}^{p} f_j(i)}{\max_{k \in X}\left\{\left(\frac{1}{c_k}\right)\sum_{j=0}^{p} f_j(k)\right\}}, \tag{14}$$

where c_i is the cost of project i, p is the total of objectives, $f_j(i)$ is the objective function of project i in the j-th objective, and X is the candidate project list. A high value η_i indicates that the project i has high objectives values with a low cost.

According to Equation 12, the selection rule has three phases: 1) Exploitation (if $\wp \leq \alpha_1$), the project with highest value of Ω is selected, 2)Exploitation-Exploration ($\alpha_1 < \wp \leq \alpha_2$), a roulette selection function is used to promote the best projects but not totally ignore those with low values in Ω, and 3) Exploration, where a project is randomly selected to promote constructing of new portfolios.

The first term in Equation 13 favors projects with high values in η, and the second term favors projects with high pheromone with the current portfolio which is being constructed. The last term allows sensing the performance of the portfolio when completed.

4.3 Updating Phase

An ant lays pheromone depend on the portfolio created, according to

$$\Delta\tau_{i,j} = \begin{cases} 0.25\left(1-\tau_{i,j}\right) & \text{If } c \in \text{NS}, \\ 0.50\left(1-\tau_{i,j}\right) & \text{If } c \in \text{NW}, \\ 1-\tau_{i,j} & \text{If } c \in \text{NF}, \\ 0 & \text{Otherwise.} \end{cases} \tag{15}$$

where i and j are projects in the portfolio c, NS is the non-strictly outranked front, NW is the non-weakly outranked front and NF is the non-net-flow outranked front.

5 Experimental Results

In this section, tests to verify the quality of the results are reported. This section 1) provides the experimental conditions of the tests, and 2) verifies the quality of the results obtained for each type of synergy handling.

5.1 Experimental Conditions

The following configuration corresponds to the experimental conditions that are common to the tests described in this section:

1. *Software*. Operating system, Microsoft Windows 7; programming language, Java; compiler, JDK 1.6
2. *Hardware*. Computer equipment dual-processor Xeon (TM) CPU 3.06 GHz in parallel and 4 GB RAM.
3. *Instances*. Two groups of 30 instances and nine objectives each one. One group with 25 projects, and the second one with 100 projects.
4. *Addressed synergy types*: Redundancy and synergy in objective functions (negative and positive). For instances of 25 projects, there are between three and six synergetic relations; for instances of 100 projects, between 12 and 24.
5. *Performance measurement*. Performance is measured according to the number of Non-Outranked Solutions(NOS) found by the algorithm. In this work, a portfolio x is considered as NOS if has $S(O,x) = 0$, $W(O,x) = 0$ and $F(O,x) = 0$.
6. *Parameters values*: The values of the preference model parameters are $\lambda = 0.67$, $\beta = 0.1$ and $\epsilon\varepsilon = 0.05$. The values of ACOS-SPP parameters are $w = 0.35$, $\alpha_1 = 0.7$, $\alpha_2 = 0.2$ and $\rho = 0.05$.
7. *ACOS-SPP stop criteria*: ACOS-SPP finishes if any of following conditions occurs:
    a) 1000 iterations are reached.
    b) NS, NW, NF fronts remain unchanged for ten iterations.

5.2 Comparison between Synergy Handling Techniques

For instances of 25 projects, an enumerative search was performed for knowing the NOS. Moreover, two versions of ACOS-SPP were created; the first handles the synergy into the multiobjective function, and the second one adding artificial projects. Figure 1 presents a performance comparison chart between both synergy techniques. As can be observed, except for three instances, both methods find the optimal solution. No relevant difference on execution time was observed.

For instances of 100 projects, it was not possible to perform an enumerative search. The preference model was applied to the results provided by each version

of the algorithm, and the amount of NOS found for each was counted. Figure 2 presents these results. The performance difference in favor of "synergy in the objective function" was 15% on average.

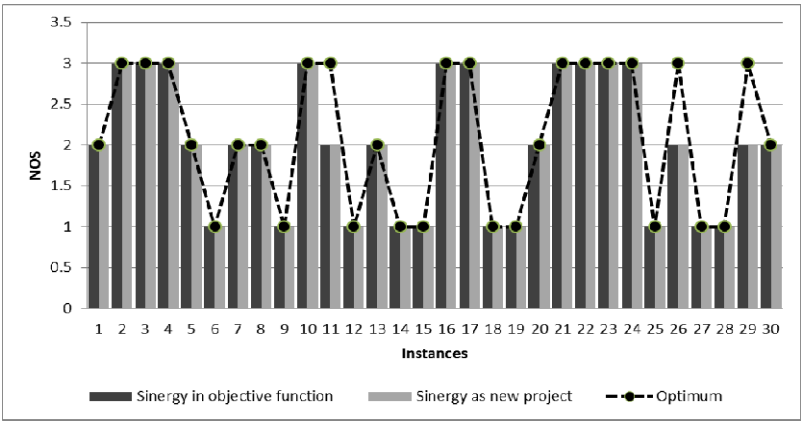


Fig. 1. Performance on instances of 25 projects

Furthermore, significant differences in the execution time were observed. In Figure 3 shows the time run consumed by each version of the algorithm. The time reduction averaged 11% by using "synergy in the objective function".

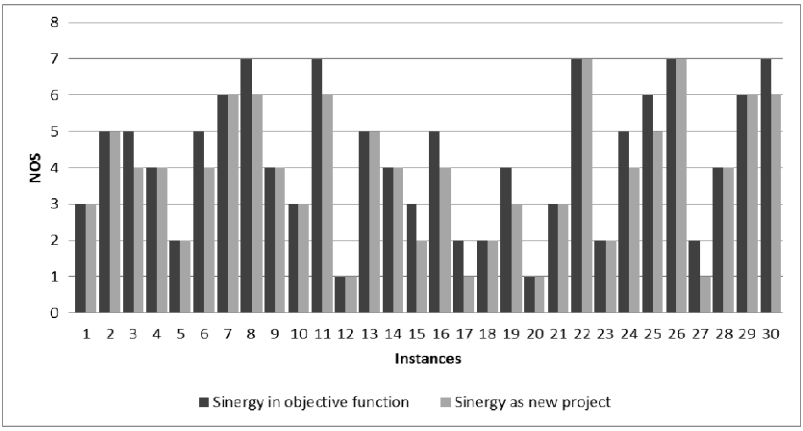


Fig. 2. Performance on instances of 100 projects

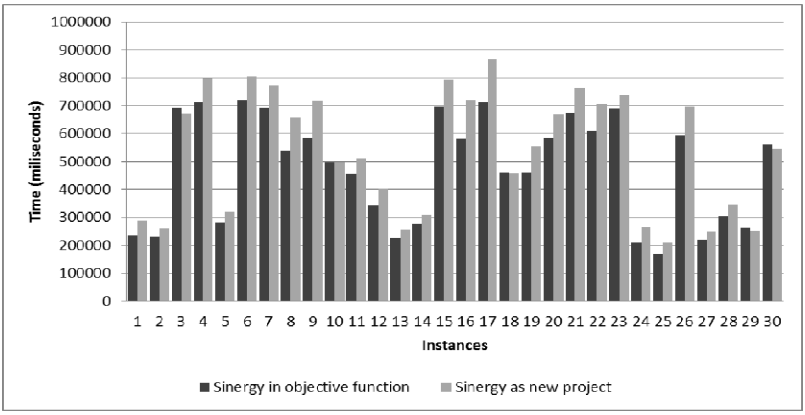


Fig. 3. Execution time on instances of 100 projects

6 Conclusions and Future Work

This article was elaborated to select one of the two most used forms to handle the synergy, and add it into an ant colony algorithm to solve SPP, called ACOS-SPP. The choice had to be made based on experimental evidence. According to Section 5.1, handling of synergy as an adaptation of multiobjective function offers a better performance, in terms of the solution quality and execution time, at least in the instance sets used.

It is important to note that adding artificial projects and redundancy rules is more flexible. In this way, all synergy type presented in Section 2.1 can be addressed, whereas the other needs more specific changes for each synergy type, either in the multiobjective function or in feasibility conditions. Despite its flexibility, it increases the number of unfeasible combinations by adding artificial projects, which can affect the performance of search algorithms, as was the case of ACOS-SPP.

On the test instances used, there was an average increase of 15% in performance, and a 11% reduction of consumed time, by adapting the objective function instead of adding artificial projects.

As future work, we plan to include all synergy types of the Section 2.1. Moreover, to solve other SPP cases, for example partial supporting and scheduling.

References

1. Brans, J., Mareschal, B.: PROMETHEE Methods. In: Multiple Criteria Decision Analysis: State of the Art Surveys, pp. 163–190. Springer, New York (2005)
2. Castro, M.: Development and implementation of a framework for the forming of R & D portfolios in public organizations. Masters Thesis, Nuevo Leon Autonomous University (2007)

3. Carazo, A.F., Gómez, T., Molina, J., Hernández-Díaz, A.G., Guerreo, F.M., Caballero, R.: Solving a comprehensive model for multiobjective project portfolio selection. Computers & Operations Research 37(4), 630–639 (2010)
4. Coello, C., Van Veldhuizen, D.A.C.A., Lamont, G.B.: Evolutionary Algorithms for Solving Multi-Objective Problems. Kluwer Academic Publishers, New York (2002)
5. Coello Coello, C.A., Lamont, G.B., Van Veldhuizen, D.A.: Evolutionary Algorithms for Solving Multi-Objective Problems. In: Genetic and Evolutionary Computation, 2nd edn. Springer (2007)
6. Doerner, K., Gutjahr, W.J., Hartl, R., Strauss, C., Stummer, C.: Pareto ant colony optimization: A metaheuristic approach to multiobjective portfolio selection. Annals OR 131, 79–99 (2004)
7. Dorigo, M., Gambardella, L.: Ant colony system: A cooperative learning approach to the traveling salesman problem. IEEE Transactions on Evolutionary Computation 1(1), 53–66 (1997)
8. Doumpos, M., Marinakis, M., Marimaki, Y., Zopounidis, M.: An evolutionary approach to construction of outranking models for multicriteria classification: The case of ELECTRE TRI method. European Journal of Operational Research 199(2), 496–505 (2009)
9. Durillo, J.J., Nebro, A.J., Coello Coello, C.A., García-Nieto, J., Luna, F., Alba, E.: A study of multiobjective metaheuristics when solving parameter scalable problems. IEEE Transactions on Evolutionary Computation 14(4), 618–635 (2010)
10. Fernández, E., López, E., López, F., Coello Coello, C.A.: Increasing selective pressure towards the best compromise in evolutionary multiobjective optimization: The extended NOSGA method. Information Sciences 181(1), 44–56 (2011)
11. Fernández, E., López, E., Bernal, S., Coello Coello, C.A., Navarro, J.: Evolutionary multiobjective optimization using an outrankingbased dominance generalization. Computers & Operations Research 37(2), 390–395 (2010)
12. Fernández, E., Navarro, J.: A genetic search for exploiting a fuzzy preference model of portfolio problems with public projects. Annals OR 117, 191–213 (2002)
13. Fernández, E., Navarro, J., Bernal, S.: Multicriteria sorting using a valued indifference relation under a preference disaggregation paradigm. European Journal of Operational Research 198(2), 602–609 (2009)
14. Fernández, E., Flerida, L., Mazcorro, G.: Multi-objective optimisation of an outranking model for public resources allocation on competing projects Int. J. Operational Research 5(2) (2009)
15. García, R.: Hyper-heuristic to solve the problem of social portfolio. Master's Thesis, Madero Institute of Technology (2010)
16. Ghasemzadeh, F., Archer, N., Iyogun, P.: A zero-one model for project portfolio selection and scheduling. Journal of the Operational Research Society 50(7), 745–755 (1999)
17. Hakanan, J., Miettinen, K., Sahlstedt, K.: Simulation-based interactive multiobjective optimization in wastewater. In: International Conference on Engineering Optimization, ENGOPT 2008, Río de Janeiro (2008)
18. Liessio, J., Mild, P., Salo, A.: Preference programming for robust portfolio modeling and project selection. European Journal of Operation Research 181, 1488–1505 (2007)
19. Marakas, G.: Decision Support Systems and Megaputer, 2nd edn. Prentice Hall, Upper Saddle River
20. Reiter, P.: Metaheuristic Algorithms for Solving Multiobjective/ Stochastic Scheduling and Routing Problems. Ph.D. Thesis. University of Wien (2010)

21. Roy, B.: The Outranking Approach and the Foundations of ELECTRE methods. In: Reading in Multiple Criteria Decision Aid, pp. 155–183. Spinger (1990)
22. Roy, B.: Multicriteria Methodology for Decision Aiding. Kluwer, Dordrecht (1996)
23. Zitzler, E., Thiele, L.: Multiobjective Evolutionary Algorithms: A comparative case study and the Strength Pareto Evolutionary Algorithm. IEEE Transactions on Evolutionary Computation 3(4), 257–271 (1999)

Part V

Evolutionary Methods and Intelligent Computing

Practical Aspects on the Implementation of Iterative ANN Models on GPU Technology

Mario I. Chacon-Murguia and Jorge A. Cardona-Soto

Visual Perception Applications on Robotic Lab
Chihuahua Institute of Technology, Chihuahua, Mexico
mchacon@itchihuahua.edu.mx, jorge_cardona@ieee.org

Abstract. There has been an increasing use of the graphic processing unit (GPU) in many areas including artificial neural networks (ANN) for several years. However, reported works concentrate on the application itself and not on the methodology used to implement the ANN model in the GPU. This paper presents a set of practical aspect to be considered by new GPU user in the implementation of ANN in GPUs. To illustrate the proposed aspects, the paper describes the realization of the Pulse Coupled Neural Network (PCNN), an iterative model, following these aspects and discusses the problematic of synchronization presented in this and other ANN models that is not treated in other works.

1 Introduction

Nowadays the graphic processing unit, GPU, has become an important tool on different research areas where massive computation is required. GPU technology is currently available in many processing platforms [1] and at affordable cost. Parallel processing is a key point on GPU technology making it very suitable for parallel model realizations which are inherent to bi-inspired algorithms [2]. Based on these considerations the artificial neural network, ANN, area is a very suitable area to obtain the benefits of the GPU technology. During the last years, different ANN works have been reported in the literature. A no serial implementation of a Systemic Computation Architecture is reported in [2]. The GPU development of the SOM and MLP models are presented in [3]. More complex models like CNN, [4], and spiking ANN have been brought to the GPU technology, [5]. Even complete software including different tools and biologically-inspired models to be used for cognitive and developmental robotics research have been published in the literature [6].

Notwithstanding several ANN realizations in GPU technology has been reported in the literature, there exists some gap of information on thes literature. Most papers assume that the reader master the fundamentals of GPU and also the knowledge about the CPU and GPU cooperation [7]. The paper presentation goes directly to the ANN implementation without providing inside hints to the reader about consideration for a good development or a methodology for easy translation

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 433–449.
springerlink.com

of other languages to the GPU. In other cases there is a brief explanation or description of the methodology but in both cases the information provided is not enough for new GPU prospective users. Based on these issues, the present work is intended to provide new information to ANN researchers, who want to incursion into the GPU technology, which now is available in almost any computer.

The contributions of the work are: to establish a set of important suggestions for the realization of ANN on GPU, synchronization of block tasks when working with iterative ANN models, a methodology for Matlab to GPU implementation and the realization of the PCNN ANN in a GPU. The organization of the work is as follows. A set of important suggestions for the realization of ANN in GPU is presented in Section 2. Section 3 describes the implementation of the PCNN model into the GPU architecture following the set of suggestions mentioned in Section 2. Finally Section 4 highlights the results and conclusions of the work.

2 Methodology Suggestions

In these days there exist different development platforms for GPU technologies. In this work we will concentrate on the Matlab, Visual C++ tools and the CUDA Nvidia architecture. The GPU technology can be viewed as a multiprocessor unit or device able to execute several unit of software at the same time. Readers interested in acquired a deeper inside of the hardware of the GPU are encouraged to consult [8]. This section concentrates on the explanation of a set of suggestions in the implementation of ANN into the GPU technology.

2.1 Technical Capability and Concepts

One of the first points to consider in the design of an algorithm in GPU is to verify the requirements to write and compile a GPU program. Therefore, it is necessary to confirm that the computer count on a hardware compatible with CUDA. It is also necessary to install the latest development driver from Nvidia and the CUDA toolkit. This toolkit includes the compiler and the "visual profiler". The software Parallel Nsight may also be installed to configure Visual Studio C++ and also to use debug tools directly on the GPU. Other software is the CUDA SDK tool that provides application examples. In case it is desire to work only on the Matlab environment, the hardware needs to have a CUDA capability at least of 1.3.

2.1.1 Basic Concepts Definition

Some important definitions in the GPU terminology are: CUDA Kernel, it is a parallelizable part of an application that is executed by the GPU processors with an array of threads. Thread, the smallest unit of processing. All threads run the same code and each thread has an ID, threaded.x/y that is used to compute memory addresses and make control decisions. Thread block, set of threads that can cooperate. They have shared memory, and can be synchronized inside the same block.

Each block has its own ID, blockIdx.x/y. 1D, 2D or 3D arrays can be constructed with the blocks. Grid, a set of thread blocks. Device, it is a GPU. Host, it is a CPU. These concepts can be visualized in Fig. 1.

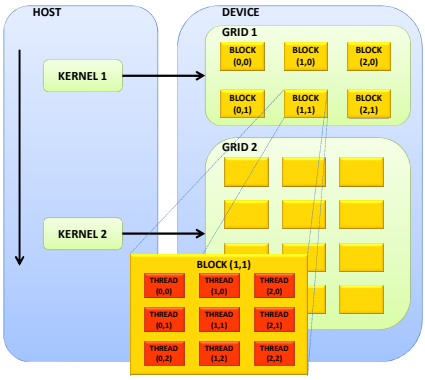


Fig. 1. GPU internal architecture

2.1.2 Memory Model

The threads can access different types of memories, shared or not shared. The memory hierarchy is shown in Fig. 2. It is composed of; the Local Memory, that can be accessed only by its thread. Each block of threads includes a shared memory space for the threads that conforms the block with a life time equal to the existence of block. Next is the global memory that is accessible to all threads. The constant and texture memories are accessible to/from all threads but are read only. Global memories have a life time equal to the existence of the kernel.

2.1.3 Function Type Qualifiers

The function type qualifier indicates where a function is executed or called from. *_devide_* specifies that a function is executed by the GPU. This function can only be called from other GPU functions.

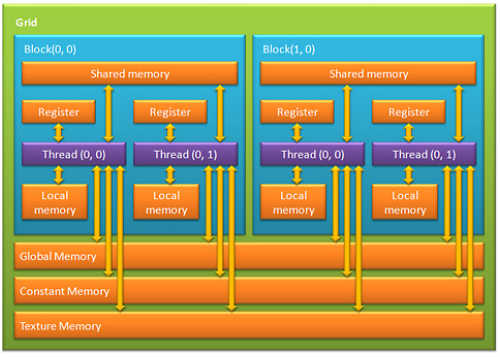


Fig. 2. CUDA memory model

*_globa_* the function is called from the CPU to be executed by the GPU. It cannot be called from GPU functions. It must return void.

*_host_* the function is executed by the CPU and it can only be called from the CPU.

2.1.4 Function Variable Type Qualifiers

The variable type qualifier determines the memory location of the variable as well as its life cycle.

*_device_* is for variables in the global memory space with life cycle corresponding to the complete application, accessible from all threads in the grid and also from the CPU with the CUDA runtime library, i.e. cudaMemcpy.

*_constant_* variable located at the constant memory with life cycle of the complete application. It is accessible to all threads in the grid and from the CPU with the runtime library.

*_shared_* variables in the shared memory of a block with life cycle corresponding to thread block. It is only accessible by the threads in the same thread block.

2.1.5 Execution Configuration

This process consists on a call from the CPU to execute a function, *_global_*,(Kernel) in the GPU. The difference between invoking a kernel and calling a normal C function is the specification of the amount of threads per block and blocks per grid. This is done by

List 1

```
__global__ void Kernel(float *parameters);
//Called as:
Kernel <<< (dim3 Dg), (dim3 Db), (size_t Ns) >>> (parameters);
```

where Dg indicates the grid dimension. Dg.x x Dg.y is the number of blocks in the grid, Db corresponds to the dimension of the blocks. Db.x x Db.y x Db.z is the number of threads in the block, and Ns (optional) is the number of bytes in the shared memory.

2.2 Matlab Algorithm Preparation

If the implementation starts from a Matlab code, it requires a preparation in order to facilitate the codification into CUDA C/C++. For example the sum operator on matrices is overloaded in Matlab, so instead of work with

List 2

```
function Matrix3=sum_matrix(Matrix1,Matrix2)
Matrix3=Matrix1+Matrix2;
```

A better option to visualize the code is

List 3

```
function Matrix3=sum_matrix(Matrix1,Matrix2)
for(i=1:1:size(Matrix1,1))
```

```
 for(j=1:1:size(Matrix1,2))
  Matrix3(i,j)= Matrix1(i,j) + Matrix2(i,j);  end; end;
```

Doing that, the parallelization can be implemented as

List 4

```
    __global__ void sum_matrix(float *Matrix1,float *Matrix2,float *Matrix3,int size_x,int size_y)
{int i=blockIdx.x*blockDim.x+threadIdx.x;
int j=blockIdx.y*blockDim.y+threadIdx.y;
int index = i * N + j;
if (i < N && j << N)
Matrix3[index] = Matriz1[index] + Matriz2[index];}

int main(){
dim3 dimBlock (blocksize, blocksize); //Block thread dimensions
dim3 dimGrid (size_x/dimBlock.x, size_y/dimBlock.x); //Block grid dimensions
//Kernel call
sum_matrix<<<dimGrid, dimBlock>>>( Matrix1, Matrix2, Matrix3, size_x, size_y);
return 0;}
```

2.3 Data Structures and Memory Access

A data structure is a form to organize a set of elemental data in order to optimize their manipulation.

2.3.1 Matrix Data Structure

In this section it will be shown the advantage of using a correct data structure. Assuming a 1000 pixels RGB color image where data may be arranged in two ways by; block, all pixels of one color followed by all pixels of the other two colors, $R_1 \ldots R_{1000}$ $G_1 \ldots G_{1000}$ $B_1 \ldots B_{1000}$ or alternated R_1 G_1 $B_1 \ldots R_{1000}$ G_{1000} B_{1000}. If we consider the transformation of the color image to a gray scale image it can be noticed the influence of the data structure in the computation time as illustrated in the *List 5* block data structure with execution time of 83ms and *List* 6 alternated data structure with 57ms. Both times correspond to an average execution of the code of 10 realizations.

List 5

```
__global__ void RGB_2_GRAY (char *d_charIn, char *d_charOut, int WIDTH, int HEIGHT)
{  int col = blockIdx.x * blockDim.x + threadIdx.x; //Obtain ids
int row = blockIdx.y * blockDim.y + threadIdx.y;
if(col>=0 && row>=0 && col<=WIDTH && row<=HEIGHT) //Limit working area
{  //Define local variables
  float r,g,b;
  WIDTH*=3;  col*=3;
 //Save original image data
 b=(d_charIn[row*WIDTH+col] & 0xff); //Blue
 g=(d_charIn[row*WIDTH+col+102400] & 0xff); //Green
 r=(d_charIn[row*WIDTH+col+204800] & 0xff); //Red
 b=(r+g+b)/3; //Averaging
d_charOut[row*WIDTH+col]=b; d_charOut[row*WIDTH+col+102400]=b;
d_charOut[row*WIDTH+col+204800]=b;}}
```

List 6

```
__global__ void RGB_2_GRAY (char *d_charIn, char *d_charOut, int WIDTH, int HEIGHT)
{
   … same as List 5
// Save original image data
b=(d_charIn[row*WIDTH+col+0] & 0xff); //Blue
g=(d_charIn[row*WIDTH+col+1] & 0xff); //Green
r=(d_charIn[row*WIDTH+col+2] & 0xff); //Red
b=(r+g+b)/3; //Averaging
d_charOut[row*WIDTH+col+0]=b; d_charOut[row*WIDTH+col+1]=b;
d_charOut[row*WIDTH+col+2]=b;}}
```

2.3.2 Memory Allocation

The next example illustrates the effect of memory allocation in Matlab. *List* 7 shows a code without memory allocation and *List* 8 with memory allocation.

List 7

```
for(i=1:1:size(Matrix1,1))
 for(j=1:1:size(Matrix1,2))
  Matrix3(i,j)=Matrix1(i,j)+Matrix2(i,j);%Matrix addition 5kx5K elements
 end;end;
```

List 8

```
Matrix3=zeros(5000,5000); %Memory allocation 5Kx5K
for(i=1:1:size(Matrix1,1))
 for(j=1:1:size(Matrix1,2))
  Matrix3(i,j)= Matrix1(i,j) + Matrix2(i,j);
end; end;
```

It is noticeable the reduction in time from 1.2569 for *List 7* to 0.46956 for *List 8*, just by performing memory allocation. *List 9* illustrates the case of memory allocation in the GPU implementation.

List 9

```
__global__ void sum_matrix(float *Matrix1,float *Matrix2,float *Matrix3,int size_x,int size_y)
{int i=blockIdx.x*blockDim.x+threadIdx.x;
int j=blockIdx.y*blockDim.y+threadIdx.y;
int index = i * size_x + j;
if (i < size_y && j < size_x) //Working region
Matriz3[index] = Matriz1[index] + Matriz2[index];}
int main(){
…
dim3 dimBlock (blocksize, blocksize); //Dimensions of thread by block
dim3 dimGrid(size_x/dimBlock.x,size_y/dimBlock.x);//Block Grid dimension
size_t memSize = 5000 * 5000 * sizeof(float); //5Kx5K
cudaMalloc((void**) &Matriz1,  memSize ); //Memory allocation
cudaMalloc((void**) &Matriz2,  memSize );
cudaMalloc((void**) &Matriz3,  memSize );
    cudaMemcpy(Matrix1,Mat1,memSize,cudaMemcpyHostToDevice);
    cudaMemcpy(Matrix2,Mat2,memSize,cudaMemcpyHostToDevice);
//Kernel call
sum_matrix<<<dimGrid,dimBlock>>>(Matrix1,Matrix2,Matrix3,size_x, size_y);
cudaMemcpy(Mat3,Matrix3,memSize,cudaMemcpyDeviceToHost);
cudaFree(Matrix1); //Release memory allocated
cudaFree(Matrix2);cudaFree(Matrix3); return 0;}
```

2.3.3 Global Memory Access

Memory access by the threads in the GPU must follow a regular pattern as indicated in Fig. 3 because they are design to work in this way. In some cases memory access cannot be achieved as indicated and this reduces the performance of the algorithm. Therefore, if random and not sequential memory access is expected, then it is recommended to use the texture memory because it includes a faster cache memory than the global memory. However, since the texture memory is read only from the kernels, data writing must be done by coping data from RAM memory of the CPU.

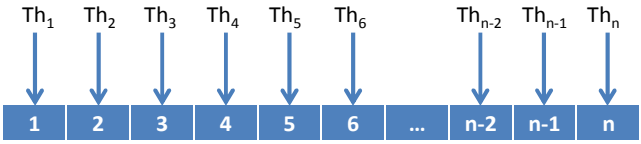


Fig. 3 Thread memory access

2.3.4 Shared Memory Access

Shared memory is useful when data must be reused because shared memory is faster than global memory. However care must be taken to avoid possible conflicts when more than one thread try to access the same location at the same time. If this situation happens then memory latency may occur.

2.4 Parallelize or Not Parallelize?

A key point on the use of GPU is to determine whether or not parallelize. Two paramount points that indicate the necessity of parallelize are the following.

1. If a time consuming code needs to be executed several time during experimentation.
2. For high demand computational (CPU, memory, access disk, etc.) programs. For example ANN implementation.

2.4.1 Consideration at Parallelizing

In order to obtain an efficient parallelization it is important to consider;

1. Computational independency. Identify independent task like for's, do's, subroutines or independent modules.
2. The previous independent task must be where most of the computational load is achieved.

Note that computational independency may imply also data independency. As an example, let consider the code in *List* 10. Lines 1 and 8 are not parallelized because are disk read and write sequential operations. Lines 2 – 4 are memory

allocation and variable initialization, which must be done from the CPU. The computational load in the matrix product in line 5 does not worth to be parallelized. The convolution in line 6 represents the higher computational load and there exist computational independency, thus, it is possible to divide it into parts and in consequence is ideal to be parallelized.

List 10

```
1 A=double(imread('myimagein.jpg'));%BW Image  640x480
2 B=zeros(480,640); % Output image
3 c=ones(3,3)/9;
4 d=rand(3,3);
5 d=c*d; % Matrix product 3x3
6 B=conv2(A,d,'same');%Convolution with image
7 B=(B+A)/2;
8 imwrite(B,'myimageout.jpg');
```

2.5 Synchronization

Synchronization is an inherent issue in any parallel architecture. Ideally, it must be desired to take a serial code, distribute the operations in equal number in the threads and let the CUDA cores to execute them in parallel. However, in some signal processing algorithms as well as in ANN models, it is necessary to use previous output data. This case corresponds to iterative algorithms with data dependency. If the operations of these algorithms are forced to run in parallel without synchronization, they will generate incorrect results, since the variables will be used before its value is calculated or otherwise updated. This condition is called data dependency and is the Achilles heel of any parallel processing architecture.

It should be noted that the essential condition required to successfully parallelize a loop is that each iteration must be independent of other iterations. If a cycle meets this condition, then the execution order of iterations does not matter, they can be executed in any order, and the result remains the same. This feature is derived from data dependence, suggestion 2.4.1. For a cycle with data dependency no iteration of the cycle must write a value to a memory location that is for reading or writing for any other iteration of this cycle. Thus, to determine whether a particular cycle can run in parallel, it is necessary to examine how the variables are used in cycles. It is necessary to pay particular attention to the variables that appear on the left side of assignments, because the data dependency occurs only when the memory locations are modified. If a variable is not changed, there is no data dependency associated with it.

Despite the possible problems of synchronization in ANN and signal processing applications where it becomes necessary to use iterative algorithms with data dependency, it is still a better option to use the parallel architecture in this kind of applications because of the high-density calculations.

When a parallel architecture becomes necessary to use thread synchronization for proper operation, it should be noted that the key for a correct parallel processing, i.e. to affect as little as possible parallelization, is to divide the algorithm into tasks where the need for synchronization is less. The less it is required, the better the results in speed.

Nvidia through CUDA architecture provides two functions that allow synchronization;

_syncthreads(). This function is executed in one kernel and it synchronizes all threads of the same block at one point before they can continue execution. However, it does not work with threads of different blocks. It is ideal to write and then read from shared memory, although it is not convenient to synchronize accesses to global memory.

_threadfence(). Unlike the pervious function, it is used for a global synchronization. It assures that all operations on the global memory inside a block preceding this instruction have been completed before continuing. However this function may generate an overload reducing the computation speed. An alternative to this situation is to copy data from the CPU memory to the GPU, execute the kernel in one iteration, then copy data from an output variable on the GPU to an input variable in the same GPU and execute the kernel again. This process is executed until the number of necessary iteration is reached. At the end, just copy the GPU output variable information to the CPU. This process is illustrated in Fig. 4.

In summary, the best way to synchronize blocks is to execute the kernel in the GPU and make the CPU to wait for all blocks to finish its execution and then start the execution in the GPU again.

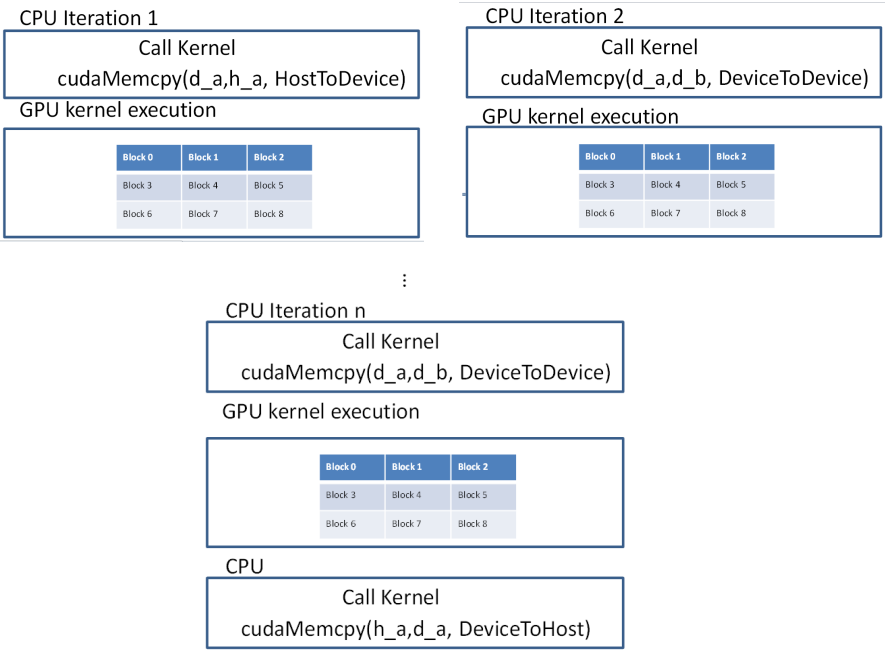


Fig. 4. An alternative to iterative process

2.6 Final Considerations

It is important to keep in mind the bandwidth communication of the GPU with the CPU when transferring data, because it must wait until the transfer is complete before continuing with program execution.

One of the weaknesses of GPU hardware previous to version 1.3 with respect compute capability is that precision of floating point arithmetic operations are sometimes not fully adhere to the IEEE floating point standard. For this reason it is not surprising that sometimes the results of the GPU and the CPU do not match completely.

Among the limitations of the CUDA architecture is that it does not support the display of images on screen. This operation requires the Nvidia primitive libraries (NPP), or to use some library that makes use of these primitives, such as OpenCV or CUVI.

3 PCNN Model Implementation

The implementation of the PCNN model is described in this section. A brief description of the PCNN model is presented first and then follows the GPU implementation including references to the suggestion of Section 2.

The PCNN model implemented is the one proposed in [10] which is briefly described next.

$$F_{ij}(n) = S_{ij} + F_{ij}(n-1)e^{-\alpha_F} + V_F Y_{ij}(n-1) * W \tag{1}$$

$$L_{ij}(n) = L_{ij}(n-1)e^{-\alpha_L} + V_L Y_{ij}(n-1) * M \tag{2}$$

$$U_{ij}(n) = F_{ij}(n)\left[1 + \beta L_{ij}(n)\right] \tag{3}$$

$$\theta_{ij}(n) = \theta_{ij}(n-1)e^{-\alpha_\theta} + V_\theta Y_{ij}(n-1) \tag{4}$$

$$Y_{ij}(n) = \begin{cases} 1 \text{ if } Uij(n) > \theta ij(n) \\ 0 \text{ otherwise} \end{cases} \tag{5}$$

The feeding region is represented by Eq. (1), where S_{ij} is the input (pixel information), α_F is the time constant of the leakage filter of the feeding region, V_F is a potential, $Y(n)$ is the neuron output at time n, (all neuron outputs at time n is termed a pulsation), W is the feeding kernel and * stands for the convolution operation. Eq. (2) describes the linking activity. α_L is the time constant of the leakage filter of the linking region and M is the linking kernel. Equation (3) corresponds to the internal activity of the neuron element. The internal activity depends on the linking and feeding activity. In (3) β is the linking coefficient. β defines the amount of modulation of the feeding due to the linking activity. The dynamic threshold is implemented by (4), where α_θ is the time constant of the leakage filter of the threshold and V_θ is the threshold gain. Finally the output of the neuron is defined by (5).

3.1 GPU Implementation

The proposed model was previously implemented in Matlab code. Therefore the first stage was to apply suggestion 2.2 to that code. The general scheme of the PCNN implementation is illustrated in Fig. 5. This scheme is included in 3 files, main.cpp, cuda1.h and cuda.cu. These files are explained next.

The first step is to read the image to be processed. This can be done with the OpenCV library. It is necessary to include some header files; the first one is the cuda1.h which includes the PCNN object definition. The files cv.h, excore.h and highgui.h are requires in order to use the OpenCV library, *List* 11.

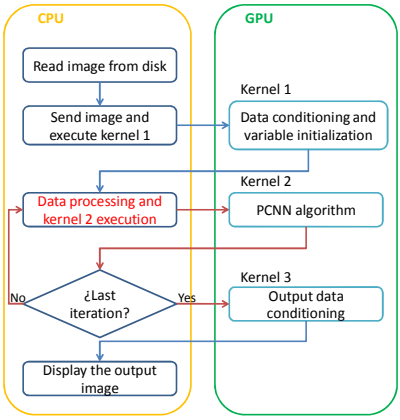


Fig. 5. PCNN implementation in GPU

List 11 file main.cpp

```
//main.cpp
//Open image, process the image and show output image
#include "cuda1.h" //Header with PCNN object
#include "cv.h" //OpenCV.
#include "cxcore.h"
#include "highgui.h"
int key; // Keyboard variable.
int main(int argc, char* argv[])
{
….
IplImage *img = cvLoadImage(filename,0); //Open image.
cvNamedWindow("Window1",1);//Create widows
cvNamedWindow("Window2",1);
cvShowImage("Windows1",img);//Show original image
*Images tt(img->data,&iteracion,img->height,img->width);//Create image object
pcnn(tt);  // Execute pcnn model
cvShowImage("Window2",img);//Show output image
                …
return 0; //Terminamos el programa
}
```

In this code the object *tt* of class *Images* is created to contain the image data, size and number of iteration to perform.

Following the suggestion 2.3.1 data structure, the class *Images* is defined as

List 13 file Cuda1.h

```
//Cuda1.h
class Images {   int *iter,*height,*width; public:   char *x; //image data pointer
 //Construction
 Images(char *x,int *iter,int * height,int *width): x(x), iter(iter), height(height), width(width) {}
```

The function pcnn is declared as friend of Images

List 14

```
friend int pcnn(Images&); };
#endif
```

The file cuda.cu contains the 3 kernels and the *pcnn* function. The file includes the header cuda1.h which contains the class *Images*. The constant definition follows this header

List 15file cuda.cu

```
//cuda.cu
//GPU code
#include "cuda1.h"
#define BETA 1
#define FEED_GAIN 0.1
#define FEED_TIME -0.1
#define LINK_GAIN 1
#define LINK_TIME -0.1
#define THRESH_ALPH 5
#define THRESH_GAIN 5
#define ITERATIONS 25
#define LINKING_KERNEL {0.05, 0.16, 0.05, 0.16, 0.16, 0.16, 0.05, 0.16, 0.05}
#define FEEDING_KERNEL {0.05, 0.16, 0.05, 0.16, 0.16, 0.16, 0.05, 0.16, 0.05}
```

Prototype functions are now declared, for example the error check CUDA function. Notice the GPU functions have the qualifier _global_, suggestion 2.1.2.

List 16

```
// Cuda Error check function
void checkCUDAError(const char *msg);
// Data conditioning and variable initialization function. Kernel 1
__global__ void M_start (char *d_datos, float *d_a , float *d_b, float *d_link, float *d_pulse, float
*d_threshold ,float *d_internal, float *d_feed);
// pcnn function. Kernel 2
__global__ void M_PCNN (float *d_a , float *d_b, float *d_link, float *d_pulse, float *d_threshold
,float *d_internal, float *d_feed);
// Output data conditioning kernel 3
__global__ void M_final (char *d_datos, float *d_b);
```

The function *pcnn* is defined as follows. It starts with variable declaration and defines the dimension of the set of blocks and thread. Since the kernels work with the same dimensions, they are defined once, suggestion 2.1.5.

List 17

```
int pcnn(Imagenes& tt)
{
// Host pointers
float *h_b;
// GPU pointers
float *d_a,*d_b,*d_link,*d_pulse,*d_threshold,*d_internal,*d_feed;
char *d_datos;
// Grid and block size definition
int numThreadsPerBlock=8*8;//image of 256*256 = threadbyblock(8*8) blocks(32*32)
int numBlocks = 32*32;
// blocks and thread dimension configuration
dim3 dimBlock(8,8);
dim3 dimGrid(32,32);
```

The next step is to determine the amount of memory required and to reserve this memory in the video card memory as well as in the RAM, according to the suggestion of data structure and memory allocation 2.3.1.

List 18

```
// host and device memory allocation
size_t memSize = numBlocks * numThreadsPerBlock * sizeof(float);
//Memory allocation for the host variable corresponding to the pulsed image.
h_b = (float *) malloc(memSize);
//GPU memory allocation
cudaMalloc((void**)&d_a,memSize);cudaMalloc((void**) &d_b, memSize );
cudaMalloc((void**)&d_link,memSize);cudaMalloc((void**)&d_pulse, memSize );
cudaMalloc((void**)&d_threshold,memSize);cudaMalloc((void**) &d_internal, memSize );
cudaMalloc((void**) &d_feed, memSize );
```

Call kernel 1 for data initialization M_start.

List 19

```
size_t memSizechar = numBlocks * numThreadsPerBlock * sizeof(char);
cudaMalloc((void**) &d_data memSizechar );
cudaMemcpy(d_data,tt.x,memSizechar, cudaMemcpyHostToDevice);
checkCUDAError("cudaMemcpy");
    //Kernel 1 call
M_start <<< dimGrid, dimBlock  >>> (d_data, d_a, d_b, d_link, d_pulse, d_threshold, d_internal,
d_feed);
cudaThreadSynchronize();
```

Now kernel 2 which includes the PCNN model is executed *List 20*. This part of the code requires synchronization, therefore it is necessary to wait to finish before a new iteration would be started, suggestion 2.5. The code also includes local variable initialization according to suggestions 2.3.3.

List 20

```
// kernel 2 call
M_PCNN <<< dimGrid, dimBlock  >>> (d_a, d_b, d_link, d_pulse, d_threshold, d_internal, d_feed);
cudaThreadSynchronize();// GPU synchronization
printf("Iteration 1\n");
```

Back to the CPU and before the second iteration, data of *Y(n)* in the variable *d_b* is copied to *Y(n-1)* variable *d_pulse*. This memory operation is synchronized and error is checked.

List 21

```
for(int n=1;n<*tt.iter;n++)
{
cudaMemcpy( _pulse,d_b,memSize,cudaMemcpyDeviceToDevice);
cudaThreadSynchronize();
checkCUDAError("cudaMemcpy");
M_PCNN<<<dimGrid,dimBlock>>>(d_a,d_b,d_link,d_pulse,d_threshold,d_internal,d_feed);
// Wait for the GPU to end
printf("Iteration %d\n",n+1);
}
```

Now output data is arranged using the global function M_final corresponding to kernel 3

List 22

```
// kernel 3 call
M_final <<< dimGrid , dimBlock  >>> (d_datos,d_b);
```

Finally, in the CPU, data arranged are copied from the GPU to the CPU. Memory copy is synchronized and errors are checked. At the end, all memory used is released and the result is displayed, suggestion 2.6.

List 23

```
// Copy form device to host and releases memory
cudaMemcpy(tt.x,d_datos,memSizechar,cudaMemcpyDeviceToHost);
cudaThreadSynchronize();
checkCUDAError("cudaMemcpy");
// release device memory
cudaFree(d_data); cudaFree(d_a); cudaFree(d_b);cudaFree(d_link);
cudaFree(d_pulse);cudaFree(d_threshold);cudaFree(d_internal);
cudaFree(d_feed);
// release host memory
free(h_b);
printf("Ok!\n");
return 0;
}//pcnn ends
```

Kernel 1 M_start

Image data is converted to uchar to float and then it is normalized. Following the suggestion 2.3.3 of avoiding access to global memory a local float variable is used in this operation.

List 24

```
//  kernel 1
__global__ void M_start (char *d_data, float *d_a , float *d_b, float *d_link, float *d_pulse, float
*d_threshold ,float *d_internal, float *d_feed)
{
int col = blockIdx.x * blockDim.x + threadIdx.x; //Obtain ids
int row = blockIdx.y * blockDim.y + threadIdx.y;
float temp; //Avoid access to global memory
if(col>=0 && row>=0 && col<640 && row<480) //define working area {
temp=(d_data[row*640+col] & 0xff);
d_a[row*640+col]=temp/255;
d_b[row*640+col]=0.0;//data initialization
d_link[row*640+col]=0.0;
d_pulse[row*640+col]=0.0;
```

```
d_threshold[row*640+col]=0.0;
d_internal[row*640+col]=0.0;
d_feed[row*640+col]=0.0;
}}
```

The PCNN model is contained in Kernel 2 M_PCNN

List 25

```
//  kernel 2 implementation
__global__ void M_PCNN (float *d_a , float *d_b, float *d_link, float *d_pulse, float *d_threshold
,float *d_internal, float *d_feed)
{
int col = blockIdx.x * blockDim.x + threadIdx.x; //Obtain ids
int row = blockIdx.y * blockDim.y + threadIdx.y;
if(col>0 && row>0 && col<639 && row<479)
 // define working area
{
//local variables declaration
float ff, lf, tf, dif;
float X; //2D convolution result
float F_K[9]=FEEDING_KERNEL;
float L_K[9]=LINKING_KERNEL;
int m;
//Exponential initialization
ff=FEED_GAIN*exp(FEED_TIME);
lf=LINK_GAIN*exp(LINK_TIME);
tf=exp(-1.0/THRESH_ALPH);
```

The *for* in the Feed function, Eq. (1) is not recommended for parallelization since it only access 9 data, suggestion 2.4.1. In fact it can be rewritten as nine products.

List 25

```
//Feed Eq. (1)
//Synaptic weights convolution with the image
m=0;
X=0.0f;
for(int i = -1; i < 2; i++)
{
for(int j = -1;j < 2; j++)
 {
  X+=d_pulse[(row+i)*640+col+j]*F_K[m];
  m++;
   }
 }
d_feed[row*640+col]=X+d_a[row*640+col]+ff*d_feed[row*640+col];
```

The same suggestion 2.4.1 for the Linking function, Eq. (2) is considered

List 26

```
//Link Eq. (2)
// Synaptic weights convolution with the image
m=0;
X=0.0;
for(int i=-1;i<2;i++)
{
 for(int j=-1;j<2;j++)
 {
   X+=d_pulse[(row+i)*640+col+j]*L_K[m];
```

```
   m++;
  }
}
d_link[row*640+col]=X+lf*d_link[row*640+col];
```

Next the internal energy, the threshold and pulsation are computed

List 27

```
//Internal energy Eq. (3)
d_internal[row*640+col]=d_feed[row*640+col]*(1+BETA*d_link[row*640+col]);
//Threshold Eq. (5)
d_threshold[row*640+col]=(THRESH_GAIN*d_pulse[row*640+col])+(tf*d_threshold[row*640+col]
);
//Pulsation Eq. (4)
dif=d_internal[row*640+col]-d_threshold[row*640+col];
d_b[row*640+col]=((int)(dif*1000000000)>0) ? 1.0:0.0;}}
```

Kernel 3, M_final

List 28

```
// kernel 3
__global__ void M_final (char *d_data, float *d_b)
{
int col = blockIdx.x * blockDim.x + threadIdx.x; //Obtain ids
int row = blockIdx.y * blockDim.y + threadIdx.y;
if(col>=0 && row>=0 && col<=639 && row<=479)
{ d_datos[row*640+col]=d_b[row*640+col]*255.0;}}
```

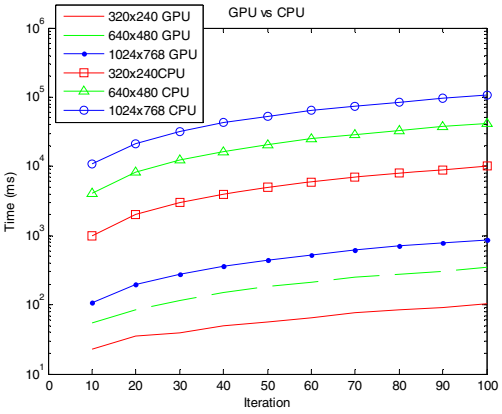


Fig. 6. GPU vs. CPU performances in the PCNN realization considering different image sizes and number of PCNN iterations(Semilog plot)

4 Results and Conclusions

Several suggestions involving technical capability and concepts, Matlab algorithm preparation, data structures and memory access, parallelize or not parallelize, synchronization and final considerations were presented in this work. These

suggestions provided important aspect to consider in the implementation of ANN models as it can be verified in the PCNN realization. This realization was tested with different image sizes and number of iterations to show the reader the benefits of implementing an iterative ANN model with CPU – GPU synchronization. Results of these tests are illustrated in Fig. 6. It is very obvious the tremendous increase of processing when the PCNN is developed in the GPU. The experiments where performed with a GPU Nvidia Quadro FX3800 with 192 Cuda-cores and 1GB DDR3 in a workstation with a CPU Intel(R) Xenon(R) E5620@ 2.40GHz, 4GM RAM running in Win7pro 64-bits.

It is our desire that the presented material in this work may motivate new prospective GPU user to venture into the benefits of implementing ANN models in the GPU technology which represent an inherent alternative for bio-inspired models.

Acknowledgments. The authors thanks to Fondo Mixto de Fomento a la Investigación Científica y Tecnológica CONACYT- Gobierno del Estado de Chihuahua, by the support of this research under grant CHIH-2009-C02-125358.

References

[1] Nickolls, J., Dally, W.J.: The GPU computing era. IEEE Micro 30(2), 56–69 (2010)
[2] Rouhipoura, M., Bentley, P.J.: Fast bio-inspired computation using a GPU-based systemic computer. Parallel Computing 36, 591–617 (2010)
[3] Zhongwen, L., Hongzhi, L., Xincai, W.: Artificial neural network computation on graphic process unit. In: Proceeding of International Joint Conference on Neural Networks, vol. 10, pp. 622–626 (2005)
[4] Dolan, R., DeSouza, G.: GPU-based simulation of cellular neural networks for image processing. In: Proceeding of International Joint Conference on Neural Networks, pp. 730–735 (June 2009)
[5] Ermai, X., McGinnity, M., QingXiang, Jianyong, W.C., Rontaig, C.: GPU implementation of spiking neural networks for color image segmentation. In: 2011 4th Inter. Congress Image and Signal Processing (CISP), vol. 3, pp. 1246–1250 (October 2011)
[6] Peniak, M., Morse, A., Larcombe, C., Ramirez-Contla, S., Cangelos, A.: Aquila: An open-source GPU-accelerated toolkit for cognitive and neuro-robotics research. In: Proceeding of Inter. Joint Conference on Neural Networks, pp. 1753–1760 (August 2011)
[7] Honghoon, J., Anjin, P., Keechul, J.: Neural network implementation using CUDA and OpenMP. In: Proceeding of Computing: Techniques and Applications, pp. 155–161 (2008)
[8] Owens, J.D., Houston, M., Luebke, D., Green, S., Stone, J.E., Phillips, J.C.: GPU Computing. Proceedings of the IEEE 96(5), 879–899 (2008)
[9] Eckhorn, B.R., Reitboeck, H.J., Arndt, M., Dicke, P.: Feature linking via stimulus-evoked oscillations experimental results from cat visual cortex and functions implications from a network model. In: Proceeding of International Joint Conference on Neural Networks, pp. 723–730 (October 1989)
[10] Johnson, J.L., Padgett, M.: PCNN models and applications. IEEE Transactions on Neural Networks 10(3), 480–498 (1999)

High Performance Architecture for NSGA-II

Josué Domínguez, Oscar Montiel, Roberto Sepúlveda, and Nataly Medina

Instituto Politécnico Nacional - CITEDI. Av. del Parque 1310, Tijuana, B., C., México.
{jdominguez,nmedina}@citedi.mx,
{o.montiel,r.sepulveda}@ieee.org

Abstract. NSGA-II is one of the most popular algorithms for solving Multiobjective Optimization Problems. It has been used to solve different real-world optimization problems. However, NSGA-II has been criticized for its high computational cost and bad performance on applications with more than two objective functions. In this paper, we propose a high performance architecture for the NSGA-II using parallel computing, for evaluation functions and genetic operators. In the proposed architecture, the Mishra Fast Algorithm for finding the Non Dominated Set was used. We present results for five different test functions.

Keywords: GA, NSGA, NSGA – II.

1 Introduction

Optimization refers to obtain the values of decision variables, which correspond to the maximum or minimum of one or more objective functions [1]. Many applications consider only one objective function, probably due to the available computational resources; however, most real problems involve one or more objectives, which are very difficult to solve because of its high computational cost. The way of finding optimal solutions of such a problem is known as multiobjective optimization (MOO).

In single-objective optimization, the search space is often well defined. When we try to optimize several objectives at the same time, the search space also becomes partially ordered. A multiobjective optimization problem could be written in the form minimize $[f_1(x), f_2(x),\ldots,f_k(x)]$ for k objective functions $f_i : \Re^n \rightarrow \Re$ subject to some equality and inequality constraints. For $x = [x_1^*, x_2^*,..., x_n^*]^T$, the vector of decision variables, our task is to determine the set F of all vectors which satisfy all the constraints, the particular set of values $x = [x_1^*, x_2^*,..., x_n^*]$ and also yields the optimum values for all the objective functions [2].

If all objective functions are for minimization, a feasible solution x is said to dominate another feasible solution y ($x \succ y$), if and only if, $z_i(x) \leq z_i(y)$ for

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 451–461.
springerlink.com

$i = 1,\ldots,K$, where K is the number of objective functions and z is a set of K objective functions, and $z_j(x) < z_j(y)$ for least one objective function j [3]. One general approach is to determine an entire Pareto Optimal solution set or a representative subset. A Pareto optimal set is a set of solutions that are non-dominated with respect to each other, as shown in Fig. 1.

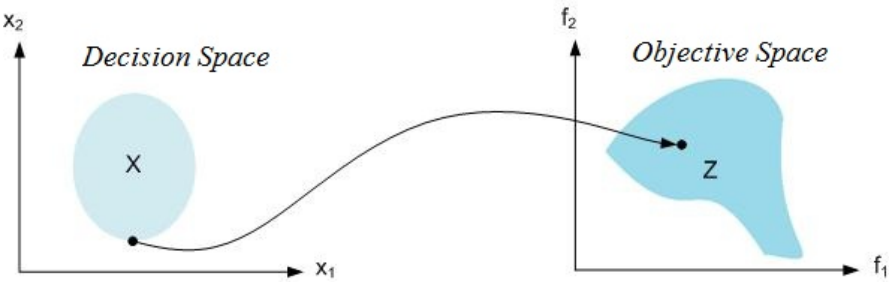


Fig. 1. Decision variable and objective space relationship

The main purpose of a multiobjective optimization algorithm is to identify solutions in the Pareto optimal set. There are three principal methods of dealing with multiple objectives:

1. Combine all the objectives into a single scalar value, typically as a weighted sum, and optimize the scalar value.
2. Solve for the objectives hierarchically, optimizing for a first objective, if there is more than one solution, optimize these solutions for a second objective, and repeat for a third etc., if appropriate.
3. Obtain a set of alternative, non - dominated solutions, each of which must be considered equivalent in the absence of further information regarding the relative importance of each of the objectives.

Generating the Pareto set can be computationally expensive and is often infeasible because the complexity of the underlying application prevents exact methods from being applicable. For this reason, a number of stochastic search strategies such as evolutionary algorithms, tabu search, Ant Colony Optimization have been developed, and they usually find a good approximation, i.e., a set of solutions whose objective vectors are not too far away from the optimal objective vectors.

2 Pareto-optimality

Definition 1. Domination: A decision vector, x_1 dominates a decision vector x_2 (denoted by $x_1 \prec x_2$, if and only if

- x_1 is not worse than x_2 in all objectives, i.e. $f_k(x_1) \leq f_k(x_2)$, $\forall k = 1,\ldots,n_k$, and.
- x_1 is strictly better than x_2 in at least one objective, i.e. $\forall k = 1,\ldots,n_k$, $f_k(x_1) \leq f_k(x_2)$.

Similarly, an objective vector, *f1*, dominates another objective vector *f2*, if *f1* is not worse than *f2* in all objective values, and *f1* is better than *f2* in at least one of the objective values. Objective vector dominance is denoted by $f_1 \prec f_2$. From Definition 1, solution *x1* is better than solution *x2* if $x_1 \prec x_2$ (i.e. *x1* dominates *x2*), which happens when $f_1 \prec f_2$. The concept of dominance is illustrated in Fig. 2.

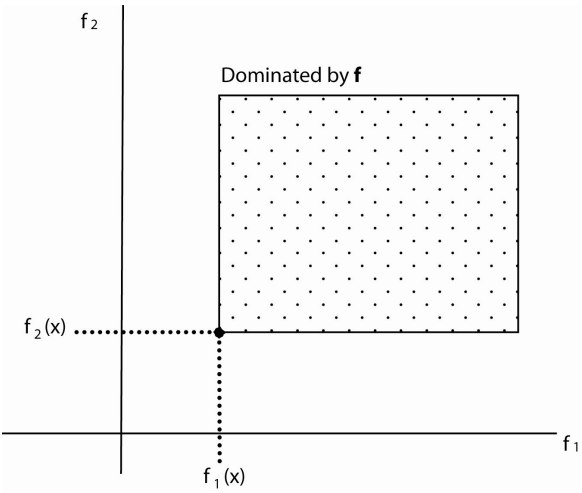


Fig. 2. Dominance concept. Point $(f_1(x),f_2(x))$ dominates all other points

Definition 2. Pareto-optimal: A decision vector $x^* \in \Omega$ is Pareto - optimal if there, does not exist a decision vector $x \neq x^* \in \Omega$ that dominates it. An objective vector, $f^*(x)$, is Pareto-optimal if *x* is Pareto-optimal.

3 Genetic Algorithms

Holland and his colleagues proposed the concept of Genetic Algorithms (GA) in the 1960s and 1970s [4]. GA is inspired by the evolutionist theory explaining the origin of species. In nature, weak and unfit species within their environment are faced with extinction by natural selection. The strong ones have greater opportunity to pass their genes to future generations via reproduction. In the long run, species carrying the correct combination in their genes become dominant in their population. Sometimes, during the slow process of evolution, random changes may occur in genes. If these changes provide additional advantages in the challenge for survival, new species evolve from the old ones. Unsuccessful changes are eliminated by natural selection.

A solution vector $x \in X$ is called an individual or a *chromosome*. Chromosomes are made of discrete units called *genes* and each gene controls one or more features of the chromosome [5]; genes are assumed to be binary bits.

An operate with a collection of chromosomes, called a population and is normally randomly initialized. As the search evolves, the population includes fitter and fitter solutions, and eventually it converges, meaning that it is dominated by a single solution.

3.1 Operators for Solution Generation

GA uses two operators to generate new solutions from existing ones: *crossover* and *mutation*. In crossover, generally two chromosomes, called *parents*, are combined to form new chromosomes, called *offspring*. The parents are selected among existing chromosomes in the population with preference towards fitness so that offspring is expected to inherit good genes, which make the parents fitter. By iteratively applying the crossover operator, genes of good chromosomes are expected to appear more frequently in the population, eventually leading to convergence to an overall good solution.

The mutation operator introduces random changes into characteristics of chromosomes. A mutation is generally applied at the gene level. In typical GA implementations, the mutation rate (probability of changing the properties of a gene) is very small and depends on the length of the chromosome. Therefore, the new chromosome produced by mutation will not be very different from the original one. Reproduction involves selection of chromosomes for the next generation; the fitness of an individual determines the probability of its survival for the next generation. There are different selection procedures in GA, depending on how the fitness values are used, such Proportional section, ranking, and tournament selection are the most popular selection procedures.

The general procedure for GA given in Algorithm 1 summarizes a general GA.

Algorithm 1 GA general procedure

```
Let t = 0 be the generation counter;
Create and initialize an n_x − dimensional population: P(0);
while Stopping condition != true do
    Evaluate the fitness, f(x_i), of each individual, x_i, in the population, P(t);
    Perform cross-over to produce offspring;
    Perform mutation on offspring;
    Select population P(t + 1) of new generation;
    Advance to the new generation, i.e. t = t + 1;
end while
```

4 NSGA-II

The NSGA-II, Non Dominated Sorting Genetic Algorithm is an improved version of the algorithm NSGA proposed by Srinivas and Deb [9]. This is a scheme for solving multiobjective optimization problems using the concept of non-dominance introduced by Goldberg [8], and a genetic algorithm to produce new solutions.

The schematic of this algorithm is shown in Fig 3 which shows that the first randomly generates a population P_t with N individuals and from them creates a R_t set of offspring solutions using the genetic operators of crossover and mutation.

Parents and offspring generated are added to a new set called R_t, in this set is made a non-dominated sorting with which are obtained different fronts of non-dominated, after each solution is assigned a crowding distance. A new parent population is formed from the fronts with better range of non-dominance and the last frontier choosing solutions with best crowding distance. The pseudocode for the NSGA-II is given in the algorithm 2:

Algorithm 2 NSGA-II pseudocode

$R_t = P_t \bigcup Q_t$
$F = fast - nondominated - sort(R_t)$
while $|P_{t+1}| < N$ **do**
  $crowding - distance - assignment(F_i)$
  $P_{t+1} = P_{t+1} \bigcup F_i$
end while
$Q_{t+1} = make - new - pop(p_{t+1})$
$t = t + 1$

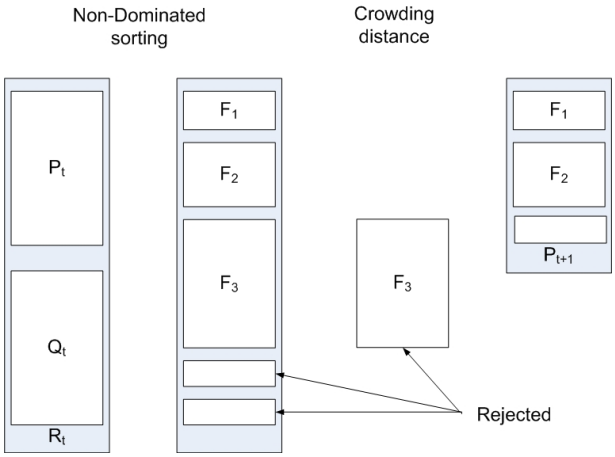


Fig. 3. Illustration of the NSGA-II process

The fast non-dominated sorting procedure separates the population into different fronts according to their level of dominance for that Deb suggested the following algorithm:

First, for each solution is calculated two entities: *i)* n_i, which is the number of solutions that dominate the solution i, and *ii)* S_i is the set of solutions that i dominates. It identifies all the points that have $n_i = 0$ and put on the list F_1. It is named to the list F_1 current front. For each solution *(i)* of the front current visit, each member *(j)* for a whole (S_i) and is reduced by one the number of individuals dominating n_j. By doing this, if some member of j the counter becomes zero $(n_j = 0)$, is put on a new list H. When all members of the current front have been revised, declare the members of the front F_1 as the first front and the front H becomes our current front.

This algorithm to maintain diversity of the population in each non-dominated frontier using a density estimator for each individual who is named *Crowding Distance,* which is an average separation between two contiguous individuals, individuals who are at the ends are assigned infinite distance the pseudocode for this procedure shown in algorithm 3.

Algorithm 3 Crowding distance procedure

```
l = |F|
for all i ∈ F do
    d_i = 0
    for all objetivo m do
        I = Sort (l, m)
        d_1^m = ∞ d_l^m = ∞
        for j ← 2, (l − 1) do
            d_j^m = d_j^m + F_{j+1}^m − F_{j−1}^m
        end for
    end for
end for
```

5 PNSGA-II

The PNSGA-II is an algorithm based in the NSGA-II structure with different sorting procedure, elite mechanism and genetic operators. The first step is initialize a population of size *2N* where *N* is a population size, each individual is an array of real numbers of size *k+m+2* as shown in the Figure, where *k* is the number of variables, m is the objective functions and other tow places for Pareto Rank (R), and Crowding Distance (d). The compute of their objective values is realized in parallel form.

Fig. 4. Encoding of individuals in a population

The next step is assign Pareto rank to each individual using the parallel sorting procedure execute using the following step.

- **Step 1.** Sort all the solutions R in decreasing order of their first objective function (S_i)
- **Step 2.** Divide the population R in equal number of the parallel workers $R_1, R_2 \ldots, R_W$.
- **Step 3.** For each parallel worker, assign Pareto Rank to R_W and return the subpopulation P_W.
- **Step 4.** Synchronize the workers and join the population P

To assign Pareto Rank to each individual of the subpopulation R_W the first step is finding the non-dominated set of R_W using the **Mishra fast algorithm**, assigns Pareto Rank 1 to its set add to new P_W and delete from R_W. For the new population R_W, find the non-dominated set, add to P_W and delete from R_W, this process continue until R_W is an empty set.

In this algorithm, one of two strategies to archive truncation is applied, the first choose the solutions with better Pareto Rank, for these strategies first sort P in ascendant order of their Rank, if some solution of the first N has Pareto Rank greater than 1 delete the last N individual and assign crowding distance, if this conditional are not satisfied preserve only the solutions with Rank 1 and compute de crowding distance.

If the size of the population P is bigger than N, the second archive truncation strategies is applied. For this strategies sort, P in decreasing order of their crowding distance and preserve only the first N solutions.

The crowding distance is uses to a diverse front by making sure each member stays a crowding distance apart. This keeps the population diversity and helps the algorithm to explore the fitness landscape. For compute de crowding distance to de population P each solution of Rank i is moved to a new subset F_i. For each F_i sort in ascendant order of their first objective function and assign infinite distance at the first and the last solutions, for the other solutions compute de average between adjacent solutions, $d_j^m = d_j^m + F_{j+1}^m - F_{j-1}^m$, this process is iterative for each objective function.

Applying the elitism strategy, we have N Parents P with the better Rank and/or Crowding distance. With the crowding comparator, we choose the better solutions and move to the mating pool for generate new offspring solutions.

The crowding comparator is a binary tournament that compare two solutions randomly selected from P, and the winner has the better Rank. If the Rank is equal, the winner have the lower crowding distance and moved to the mating pool B this process continue until B have N solutions.

To create new solutions the blend crossover (BLX-0.5) [18] and Non-uniform mutation are used, [19], these GA operators are executed in parallel form with W workers at the same time. For this process, the mating pool B is a global memory and each worker generate N/W offspring and after this are synchronized to the offspring population Q.

Each worker chooses two random solutions from B, after generate two random numbers c (crossover probability) and m (mutation probability), if p is bigger than

the selected crossover probability the worker choose other parents, else apply the BLX-5. If *m* is bigger than the mutation probability the new solution is added to the offspring population *Q;* else, the non-uniform is applied, and the new solution is added to the offspring population *Q*, this process continue until have *N*/*W* offspring. If some offspring is out of the variable boundaries, it is replaced for some parents.

6 Results

This algorithm was tested with the function SCH1, SCH2 [20], ZDT1, ZDT2 and ZDT1 [21] (minimization of both objectives).

$$SCH1\begin{cases} f_1(x) = x^2 \\ f_1(x) = (x-2)^2 \end{cases} \tag{1}$$

$$SCH2\begin{cases} f_1(x) = \begin{cases} -x, x \le 1 \\ x-2, 1 \le x \le 3 \\ 4-x, 3 \le x \le 4 \\ x-4, x > 4 \end{cases} \\ f_2(x) = (x-5)^2 \end{cases} \tag{2}$$

$$ZDT1\begin{cases} f_1 = x_1 \\ g = 1 + \dfrac{9}{n+1} \quad (\sum_{i=2}^{n} x_i) \\ f_1 = 1 - \sqrt{x_1 / g} \end{cases} \tag{3}$$

$$ZDT2\begin{cases} f_1 = x_1 \\ g = 1 + \dfrac{9}{n+1} \quad (\sum_{i=2}^{n} x_i) \\ f_1 = 1 - (x_1 / g)^2 \end{cases} \tag{4}$$

$$ZDT3\begin{cases} f_1 = x_1 \\ g = 1 + \dfrac{9}{n+1} (\sum_{i=2}^{n} x_i \\ f_1 = 1 - \sqrt{x_1 / g} - (x_1 / g)\sin(10\pi x_i) \end{cases} \tag{5}$$

For these functions the Pareto optimal is known, for SCH1 when $x \in [0,2]$, for SCH2 the Pareto optimal occurs $x \in [1,2] U [4,5]$, for the function ZDT1-3 the Pareto optimal corresponds to $x_1 \in [0,1]$ and $x_i = 0$ *for* $i = 2,3,\ldots,30$.

Fig. 5 show the non-dominated solutions obtained by PNSGA-II on SCH1 after 10 generations using crossover probability of 0.8 and mutation probability of 0.01, these probabilities are the same for the other experiments, the population size is 100. With these parameters, the mean of non dominated solution is 0.966260381, in $x \in [0.002789492, 2.002688216]$ the standard deviation of 0.578461005 and the variance of 0.334617134. In Fig. 5 show the non-dominated solutions obtained for SCH2 using the same configuration than for SCH1, the mean of non dominated solution is 2.079552681, in $x \in [0.999171542, 5.001348663]$ the standard deviation of 1.218250089 and the variance of 1.48413328.

This algorithm has good results for the ZDT functions like show in Table 1 and in Figs.5 and 6. The population size is 100, the results presented were obtained after 200 generation, the non dominated set is near to the Pareto front known for these problems and its spread in the non-dominated front is good. The convergence to Pareto Front is faster than other algorithms.

Table 1. Statistical analysis of ZDT functions

Function		x_1	x_2- x_{30}
ZDT1	Mean	0.43942224	0.005943853
	Range	[5.7005E-05, 0.99961505]	[0.003354647, 0.010692929]
	Deviation	0.3200986	0.002217063
	Variance	0.10246311	4.91537E-06
ZDT2	Mean	0.60248299	0.00430547
	Range	[0.00016236, 0.99863189]	[0.00186554, 0.00952807]
	Deviation	0.266575	0.00173071
	Variance	0.07106223	2.9954E-06
ZDT3	Mean	0.42721361	0.02621265
	Range	[7.1923E-05, 0.85166161]	[0.01333688, 0.13884388]
	Deviation	0.27739178	0.02283645
	Variance	0.0769462	0.0005215

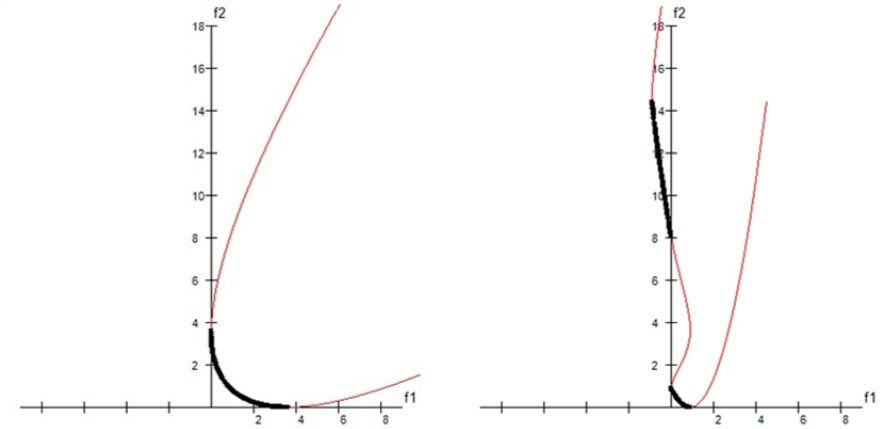


Fig. 5. Non-dominated solutions obtained by PNSGA-II on SCH1 and SCH2

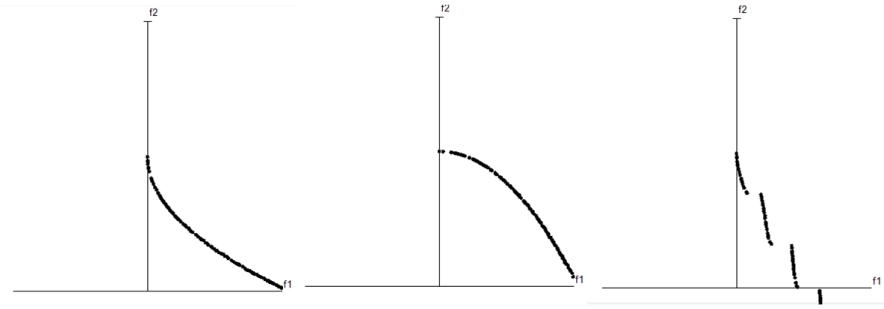


Fig. 6. Non-dominated solutions obtained by PNSGA-II on ZDT1, ZDT2 and ZDT3

7 Conclusions

The proposed scheme for applying genetic operators in parallel can be used for any genetic algorithm, the suggestion to use shared memory allows maintaining the diversity of the population, but this strategy only works if parallelization is used for large granularity.

The elitism strategy used is an improvement to the algorithm NSGA-II because the area of MOOEA has been given great importance to the spread of solutions in the Pareto optimal front. Being the only criterion of unemployment the number of generations, if the algorithm is iterated for more generations needed to reach the Pareto frontier, the only thing is that it will occur to improve the spread.

Acknowledgment. The authors would like to thank the Instituto Politécnico Nacional (IPN), Comisión de Operación y Fomento de Actividades Académicas (COFAA), and the Mexican Consejo Nacional de Ciencia y Tecnología (CONACYT) for supporting our research activities.

References

1. Rangaiah, G.P.: Multi-Objective Optimization: Techniques and Applica-tions in Chemical Engineering. World Scientific Publishing CO. Pthe. Ltd (2009)
2. Abraham, A., Jain, L.C., Goldberg, R.: Evolutionary Multiobjective Optimization: Theoretical Advances And Applications. Springer (2005)
3. Konak, A., Coit, D.W., Smith, A.E.: Multi - objective optimization using genetic algorithms: A tutorial. Reliability Engineering and System Safety 91 (2006)
4. Holland, J.H.: Adaptation in natural and artificial systems. University of Michigan Press, Ann Arbor (1975)
5. Mitchell, M.: An introduction to Genetic Algorithms. MIT Press, Cambridge (1998)
6. Goldberg, D.E.: Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley Longman Publishing Co., Inc. (1989)
7. Deb, K., Pratap, A., Agarwal, S.R., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation (2002)
8. Goldberg, D.E.: Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley Longman Publishing Co., Inc. (1989)
9. Deb, K., Pratap, A., Agarwal, S.r., Meyarivan, T.: A fast and elitist mul-tiobjective genetic algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation (2002)
10. Srinivas, N., Deb, K.: Multiobjective Optimization Using Nondo-minated Sorting in Genetic Algorithms. Evolutionary Computation (1994)
11. Corne, D.W., Jerram, N.R., Knowles, J.D., Oates, M.J.: Martin J: PESA-II: Region-based Selection in Evolutionary Multiob-jective Optimization. In: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2001 (2001)
12. Li, M., Liu, L., Lin, D.: A fast steady-state epsilon-dominance multi-objective evolutionary algorithm. Comput. Optim. Appl. Kluwer Academic Publishers (2011)
13. Deb, K.: Multi-Objective Optimization using Evolutionary Algorithms. Wiley (2001)
14. Coello, C.A.: Evolutionary Algorithms for Solving Multi-Objective Problems. Springer (2007)
15. Abido, M.A.: Multiobjective evolutionary algorithms for electric power dispatch problem. IEEE Trans. Evolutionary Computation (2006)
16. Formiga, K.T.M., Chaudhry, F.H., Cheung, P.B., Reis, L.F.R.: Optimal Design of Water Distribution System by Multiobjective Evolutionary Methods. In: Fonseca, C.M., Fleming, P.J., Zitzler, E., Deb, K., Thiele, L. (eds.) EMO 2003. LNCS, vol. 2632, pp. 677–691. Springer, Heidelberg (2003)
17. Ahmed, F., Deb, K.: Multi-objective path planning using spline representation. In: ROBIO (2011)
18. Eshelman, L.J., Schaffer, J.D.: Real-coded genetic algorithms and interval-schemata. Foundation of Genetic Algorithms 2, 187–202 (1993)
19. Michalewicz, Z.: Genetic algorithms + data structures = evolution programs, 3rd edn. Springer (1996)
20. Schaffer, J.D.: Some experiments in machine learning using vector evaluated genetic algorithms (artificial intelligence, optimization, adaptation, pattern recognition). Vanderbilt University (1984)
21. Zitzler, E.: Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications (1999)

Natural Language Interfaces to Databases: An Analysis of the State of the Art

Rodolfo A. Pazos R., Juan J. González B., Marco A. Aguirre L.,
José A. Martínez F., and Héctor J. Fraire H.

Instituto Tecnológico de Ciudad Madero, Mexico
{r_pazos_r,automatas2002}@yahoo.com.mx,
jjgonzalezbarbosa@hotmail.com, marco@marcoaguirre.com.mx,
jose.mtz@gmail.com

Abstract. People constantly make decisions based on information, most of which is stored in databases. Accessing this information requires the use of query languages to databases such as SQL. In order to avoid the difficulty of using these languages for users who are not computing experts, Natural Language Interfaces for Databases (NLIDB) have been developed, which permit to query databases through queries formulated in natural language. Although since the 60s many NLIDBs have been developed, their performance has not been satisfactory, there still remain very difficult problems that have not been solved by NLIDB technology, and there does not yet exist a standardized method of evaluation that permits to compare the performance of different NLIDBs. This chapter presents an analysis of NLIDBs, which includes their classification, techniques, advantages, disadvantages, and a proposal for a proper evaluation of them.

1 Introduction

Information is a fundamental factor in decision making. Currently the largest sources of information are stored in databases (DBs). Information stored in databases has been growing constantly, which has brought about the need for creating systems that permit users to obtain information easily, fast and reliably. In order for a user to obtain information from a database, he/she needs formulate a query in such a way that the computer interprets and generates the correct answer (usually in a database query language such as SQL). Unfortunately, only computing professionals can formulate queries in such way. Therefore, natural language interfaces to databases (NLIDBs) emerge as an alternative that permit users accessing information stored in databases through a query in natural language (NL).

The need for NLIDBs has grown popular nowadays as a consequence that many users request accessing information in different types of systems (computers, cell phones, PDAs) [8, 4].

Unfortunately, despite that fact that NLIDBs have been developed for more than 50 years, they have not attained a high enough performance (percentage of

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 463–480.
springerlink.com

correctly answered queries close to 100%) to make them widely used in business applications. Most of the NLIDBs that have achieved good results (success rate around 95%) are domain dependent; i.e, those that can only be used to query a single database; however, domain-independent NLIDBs still have deficiencies in the translation process (form NL to a DB query language) and have attained a success rate in the interval 80-90%.

It is clear that the development of effective NLIDBs continues to be an open research topic, for the simple fact that no domain-independent NLIDB has achieved a success rate close to 100%.

This analysis of the state of the art in NLIDBs aims at showing the state of progress of this technology to date. To this end, the most relevant architectures, NL processing techniques and types of GUIs that have been used for implementing NLIDBs are described together with their advantages and disadvantages, as well as examples of NLIDBs that have used them. This chapter includes a description of the most relevant problems that have prevented NLIDBs achieving a level of performance good enough to make them widely used in business applications. Finally, since evaluation is an important aspect for assessing the progress of technologies, this chapter includes an overview of the databases, query corpuses and metrics that have been used for evaluating NLIDBs, as well as a proposal for standardizing their evaluation.

2 History

The first NL querying systems were developed in the 60s, and they were basically NLIs for expert systems tailored for specific domains, being some of the most famous BASEBALL [2] and LUNAR [3]. Most of those NLIDBs were developed keeping in mind a particular database, and consequently they could not be easily modified for querying different databases or they were difficult to port to different application domains. These systems had a database or a knowledge system created manually by domain experts. Some of the first NLIDBs were based on pattern matching techniques, described in Subsection 4.1.1.

Other interfaces that were developed in the late 70s were PLANES [36], REL [36] and LADDER [36]. These systems used a semantic grammar, described in Subsection 4.1.3. A boom of NLIDBs development occurred in the 80s, when numerous research prototypes were implemented, such as CHAT-80 [3], DATALOG [3], EUFID [3], and TELI [3].

The next generation of NLIDBs used an intermediate representation language, which expressed the meaning of user queries in terms of high level concepts, independently of the database structure [3]. TEAM [20] is an experimental and portable NLIDB developed in 1985, which consists of two main components: expression mapping from NL to formal representations and transformation of these expressions to a DB query language, which permits the separation of the linguistic process from the mapping process in the knowledge base.

Masque/pro translates English queries to expressions in a logic language called LQL, which are evaluated against Prolog databases. Later on, since most existing databases are relational, the authors decided to modify this system, from which

Masque/sql evolved, which is a portable NLIDB that supports SQL databases. The main adaptation consisted of translating LQL queries to SQL expressions [3].

Several NLIDBs were developed which were commercialized. Some of the most successful were INTELLECT, Q&A, English Wizard and English Query. However, when graphical user interfaces were introduced, these NLIDBs were gradually discontinued. Their performance was not very satisfactory, due to the existence of several problems: the language coverage of the system was not obvious for users; i.e., the system might be able to answer the query but it would not generate the correct result; the origin of its mistakes could not be discerned; i.e., the user could not guess if the problem was caused by a language coverage limitation or the conceptual scope of the database; and when the NL queries were ambiguous, usually the result was not what users expected.

3 Recent Work

The boom of NLIDBs development occurred in the 80s. Despite the development of many systems at that time, gradually their use decreased due to several problems that affect their performance. In this section some of the most relevant of recent works are presented; moreover, several of the systems described in this section were included because they use translation techniques different from the traditional ones.

PRECISE [30] introduces the notion of "semantically tractable sentences", which are those that can be translated to a unique semantic interpretation. Its accuracy rate for tractable sentences is impressively high (100%), however its recall rate is just 78-95%.

NLPQC [34] is a NLIDB that is used for querying the virtual library CINDI. This system is constituted by a run-time preprocessor. The preprocessor builds a conceptual knowledge base of the DB schema using WordNet, which is used at run time for analyzing the user query according to several predefined frameworks and for generating an SQL expression.

WYSIWYM [29] uses a semantic graph inspired by the one described in [37]. The interface helps users formulate queries using predefined frameworks, through a query interface, where the user completes the framework choosing from a list of search values for the columns involved in the query.

DaNaLIX [18] is a NLIDB to query XML databases. In this system the domain knowledge can be obtained from the interaction with the user. Query analysis focuses on keywords and its relationships with the XML elements, which are used for translating the query to XQuery.

C-PHRASE [22] has an authoring tool for editing the data dictionary of the NLIDB for a specific domain. Query analysis focuses mainly on recognizing noun phrases, and it uses a semantic grammar based on rules and patterns, which are used for translating the query to SQL.

In [11] a NLIDB that uses Sequence and Tree Kernels (STKs) and some variants is presented. In this system sets of data are constructed that consist of pairs, where each pair is constituted by a NL query and its translation into SQL. The NLIDB represents these pairs through syntactic trees, and uses kernel functions and support vector machines for carrying out the translation from natural language to SQL.

4 Classification of NLIDBs

In this section three types of classification of NLIDBs are presented; according to: their type of architecture, language processing technique, and type of graphical interface.

4.1 Classification by Architecture

4.1.1 Pattern Matching Systems

The first NLIDBs used this type of method, which employs a simple technique based on patterns or rules that are applied to the NL query [3]. In pattern-matching systems, their inner workings are closely related to the peculiarities of the database to be queried; consequently, these systems are limited to specific databases and the complexity of the patterns considered. In order to exemplify the simplicity of the method, let us consider a rule that could be used for the ATIS database [33]:

Pattern of NL query:
"Flight Code" of <A Flight Number>
SQL expression that corresponds to the preceding pattern:

SELECT flight_code.flight FROM flight WHERE flight_number = <A Flight Number>

The advantage of this technique is that it is neither necessary to implement a syntactic parsing nor interpretation modules for translating the query, which facilitates their implementation. The main disadvantage is that the simplicity and superficiality of the implementation (because of the limitation of the rules defined) frequently causes an incorrect translation of the NL query to a DB query language. Additionally, the coverage of the system is limited by the number of patterns it uses.

Some NLIDBs like the system described in [1] and gNarLI use this type of technique.

4.1.2 Syntax-Based Systems

Syntax-based systems use a syntactic parser for generating the parse tree corresponding to NL queries [3]. The parse tree is mapped to the DB query language based on predefined syntactic grammars.

The syntactic parser is a program that parses NL sentences. Parsers are implemented from a set of grammar rules and lexicons. Parsers are generally designed for a specific natural language due to the idiosyncrasies of different languages regarding grammar rules and lexicons.

The result of parsing a sentence is the syntactic representation of the sentence. The representation obtained consists of three types of information: part of speech of words, phrases, and relationships between words or phrases. The part of speech of a word in a sentence is its syntactic category (verb, noun, adjective, preposition, etc.). A phrase is a sequence of two or more words that are grammatically linked without subject and predicate.

Parsers also generate a representation for a sentence, called constituent tree. Relationships between words are represented in the constituent tree. Parsers use a set of rules based on the constituent tree for generating the syntactic relationships between words and phrases.

Once a user query is parsed into a constituent tree, syntax-based systems use a syntactic grammar for mapping user queries. An example of a possible grammar for a query to the GeoBase database looks as shown below.

- S → WQ VP
- WH → "what" | "which"
- WQ → WH "river" | WH "state"
- VP → V DP
- V → "passes through" | "borders"
- DP → "Illinois" | "Missouri" | "Indiana"

Considering the following query:

Which river passes through Illinois?

According to the syntactic grammar, the system could map this query to the following logical query to the database (where X denotes a variable):

? (river(x)∩flow_through(X,Ilinois)).

The advantage of using this technique is that it facilitates mapping from the syntactic tree to the DB query language, because the tree contains detailed information of the sentence of the user query. The disadvantages are related to problems concerning domain independence and DB query language. Problems such as semantic ellipsis may cause ambiguity when the query refers to a column that belongs to several DB tables. Other problems are related to the possibility of generating several syntactic trees, which could lead to several interpretations of the user query. There also exists the difficulty of mapping the sentence directly to a DB query language (such as SQL) due to the rules defined. The syntactic structures are deeply rooted to the database information; therefore, the systems that use this technique are difficult to port to another database.

An example of NLIDB that uses this translation technique is LUNAR.

4.1.3 Semantic Grammar Systems

The semantic grammar systems are similar to syntax-based systems: syntactic parsing is used for generating the constituent tree of the user query. This technique uses predefined grammars for mapping from the constituent tree to an SQL expression. An example of a possible semantic grammar is shown below [3].

S → River_question Flow_through
River_question → "which river"
Flow_through → "passes through" State
State → "Illinois"

Semantic grammars offer more-semantically-enriched information than syntactic grammars, which extends the scope of application of such systems. Semantic information about the domain knowledge is related to the semantic grammar. Semantic grammar categories are usually selected to enforce semantic constraints.

The advantage of this type of architecture is that the semantic representation does not have a complex syntactic tree. The structures permit assigning semantic information to the tree nodes, which reduces the elliptical problems of user queries.

The disadvantage of this method is that the semantic information about the knowledge domain is hard-wired into the semantic grammar (a new semantic grammar has to be written whenever the system is configured for a new domain), which makes difficult to port NLIDBs from one database to another. Moreover, the structures of the syntactic tree can hardly be useful for other databases. Some systems intend to obtain information from the domain knowledge by interacting with the user or extracting such information from a corpus of queries formulated by users.

Examples of NLIDBs that employ this translation technique are LADDER, PLANES, REL, PHILIQA1, EUFID, ASK, DATALOG, Nchiql, QBI, CoBASE, PRECISE, NLPQC, WYSIWYM, and the system presented in [11].

4.1.4 Intermediate-Representation Language Systems

Currently most NLIDBs use this technique for translating a NL user query to an intermediate logical query (in some intermediate meaning representation language). Intermediate logical queries express the meaning of user queries in terms of high-level-world concepts, which are independent of the database structure [3].

In this type of systems a sentence is mapped first to a logical query language, and subsequently this query is translated to a DB query language. This type of systems may include more than one meaning representation language.

In intermediate-representation language systems, a system can be divided into two parts: the first generates the logical query and the second generates the DB query from the logical query. The use of logical query languages permits adding reasoning capabilities to the system through the incorporation of reasoning modules between the semantic analyzer and the DB query generator. A recent variation of this architecture consists of using an internal structure (such as a semantic network) instead of an intermediate-representation language.

The advantage of using this technique is that the part that generates the logical queries is independent of the DB management system. This technique also permits the domain to be independent of the system modules and the addition of reasoning modules that enables the system to carry out reasoning based on information stored in the database. A disadvantage is that most of these kind of NLIDBs use deductive databases, which are much less used than relational databases; thus, the scope of application is limited.

Examples of systems that use this technique are RENDEZVOUS, CHAT-80, DIALOGIC, TEAM, DIAGRAM, JANUS, TELI, TAMIC, EDITE, and C-PHRASE.

4.2 Classification by Language Processing Techniques

4.2.1 Symbolic Approach

The symbolic method was prevalent in the first decades of NLIDBs development. Words are symbols that represent objects in the real world, and they are subject to well-defined grammar rules.

In this type of approach, language is coded in rules or other forms of representation. Language is analyzed at several levels for obtaining information, from which rules are obtained that are applied for achieving the intended linguistic functionality for the system. Similarly to the human language capabilities, a rule-based reasoning is carried out which is supported by symbolic processing [2].

The advantage of this approach resides on its simplicity, since it does not require a complex implementation. Its disadvantage is that its functionality and scope are limited to the rules defined for interpretation.

Many NLIDBs use this technique since they use rules for answering user queries. Some systems that use this technique are LUNAR, LADDER, PLANES, REL, PHILIQA1, EUFID, ASK, DATALOG, TAMIC, EDITE, NLPQC, and C-PHRASE.

4.2.2 Empirical Approach

Statistical analysis is the main tool for this type of approach, since it is based on text corpora analysis. The corpora are used as source of information for the natural language [2]. In this approach the syntactic parsing is carried out on a training corpus, in which the ellipsis and ambiguity problems can be resolved considering the context of the information obtained.

The advantage of using this approach is that, unlike the symbolic approach, this method automatically carries out the acquisition and learning of more complex knowledge. Its disadvantage is that its effectiveness depends on the training corpus, and it also needs some processing time that depends on the corpus size.

Examples of NLIDBs that use this approach are PRECISE, WYSIWYM, DaNaLIX and the system presented in [11].

4.2.3 Connectionist Approach

Artificial neural networks permit modeling language processing. Instead of symbols, this method is based on distributed representations that correspond to statistical regularities in natural language [2].

The advantage of this approach consists of permitting high-level representations of knowledge, which allows more information and possibly larger domain independence. The disadvantage is that it probably needs more processing capacity that the other techniques.

An example of the use of this method is presented in [11]. Additionally, systems such as Nchiql, QBI, CoBase, and WYSIWYM use semantic graphs for interpreting user queries. In Owda's NLIDB [25] knowledge representation trees are used.

4.3 Classification by Type of GUI

4.3.1 Command Line Based

This is a method that permits users to input NL queries through a simple text line. The first NLIDBs used this type of approach because of the existing software and hardware limitations.

Though inexperienced users easily learn to interact with graphic interfaces, a well-designed command line interface has several advantages over other interface types. Since the interface does not have complex components, its simplicity and uniformity make it easy to use. In general, the execution of a command can be faster than using a graphic interface.

A disadvantage of this type of approach is that it might be complicated to minimize user mistakes. Though this type of interfaces have uniformity in the interaction with the user, since they do not have other tools such as clarifying dialogues, the results obtained are not always satisfactory because there is no feedback from the NLIDB.

An example of NLIDB that uses this type of interface is NLPQC.

4.3.2 Menu-Based

In this type of systems users formulate queries choosing from menus of phrases and search values. The menus are displayed on the screen and are constantly updated. The user can browse the menus to find out the queries that the system can answer.

The advantage of this kind of systems is its ease of use and the guarantee that queries are correctly worded. A major disadvantage is that the coverage of the system is limited by the number of phrases included in its menus. Another disadvantage is the heavy use of computer resources due to the constant updating of the system menus. Additionally, query formulation may occasionally require searching through a lot of information, which may cause a lengthy search in the menus.

Some examples of menu-based NLIDBs are NLMenu System, TEAM, and WYSIWYM, as well as the systems presented in [15] and [19].

4.3.3 Conversation/Dialogue-Based

Conversation-based systems use conversation agents that permit users to interact with the NLIDB, in such a way that the users may refine their queries through dialogues until they get the desired result. There exists another type of dialogues used by NLIDBs, which are called clarifying dialogues. These dialogues, like conversation-based dialogues, are used after the query has been preprocessed in order to clarify the meaning of a query that has not been fully understood (usually asking for omitted words in the query wording).

The advantage of this type of interface is its reliability and friendliness, since it permits the user to refine or clarify the query for attaining the desired result. Interaction with the user makes possible to integrate learning modules that permit feedback to the system, which in turn leads to improved performance. With a good dialogue design, this type of interfaces guarantees obtaining the desired results with better accuracy.

However, it might be necessary to configure the system so it adequately responds and interprets the responses obtained from the user as a result of the interaction with the system. There might also occur superfluous interactions with the user. Another disadvantage is that the interaction with the system could be lengthy when dealing with complex queries or elliptical queries (those in which words crucial for their interpretation are omitted). Due to the processing and coverage of dialogue cases, the system might require a larger amount of computational resources.

Some examples of conversation/dialogue-based NLIDBs are SHRDLU, WISBER, and TAMIC, as well as the systems presented in [15] and [25]. Other systems such as IR-NLI support spoken dialog.

4.3.4 Multimode-Based

This kind of interfaces has sophisticated selection and navigation techniques for selecting objects and context-sensitive menus for manipulating them. This technique could be considered as a combination of menu-based interfaces and dialogue-based interfaces, which in some cases includes the use of checkboxes, lists, objects, etc., and even speech for interacting with the system.

Because of the combination of techniques, multi-mode interfaces share their advantages and disadvantages. Among their advantages, it can be mentioned that users can formulate queries at a higher level, since queries can be formulated at different levels of construction, additionally they permit clarifying details of the user query.

The main disadvantage if this type of interface is that it requires a larger use of computational resources because of the larger amount of constituent elements. Occasionally, user-system interaction could be tedious and its use might possibly be difficult when the user ignores the operation of the system.

Some examples of multimode-based NLIDBs are JANUS, QBI, CoBase, infoVis, DaNaLiX, and C-PHRASE.

5 Problems in NLIDBs

For over 50 years a large number of NLIDBs prototypes and commercial systems have been developed; however, to date they have not attained satisfactory results; i.e, recall rate close to 100%.

Interfaces such as PRECISE [30] (considered one of the most successful), only answers queries that are considered "tractable", where tractable is defined as "easy to understand, questions where the words or phrases correspond to database

elements or constraints on join paths." Unfortunately, in complex databases (like ATIS), many queries involve semantic ellipsis which makes queries difficult to understand; for example in the query corpus for ATIS, the percentage of elliptical queries is approximately 85%.

The pitiful situation of commercial NLIDBs gives an indication of the backward state of NLIDB technology; most of the commercial products have been discontinued due to their poor performance and the difficulty for porting and configuring them. For example, LanguageAccess was discontinued by IBM, English Query (developed by Microsoft) was included for the last time in SQL Server ver. 8.0 in 2000, and English Wizard (by Linguistic Technology Corporation) was discontinued several years ago. ELF [5] (reputed as being one of the best NLIDBs and still available for sell) does not provide anymore technical support and it has not been updated nor improved its operation for many years.

Though many ILNBDs problems still remain unsolved, most researchers that have worked in this area, have moved to other research topics and have not continued to improve their prototypes performance.

Some of the main problems in which most of the researchers agree, are those related to portability (domain independence) and translation from NL to a DB query language [22, 30].

5.1 Domain Independence

A major issue in domain-independent NLIDBs is that it is very difficult to attain a recall rate above 90%, and usually this requires a tedious configuration process for adapting an interface to a specific database. The NLIDBs that offer domain independence need an initial configuration process before the final users can introduce their queries.

Configuring a NLIDB consists of supplying the system with the words and concepts needed for the new domain, which are related to the information stored in the database. Usually DB administrators need high levels of expertise and a long time for configuring the system [22]. Unfortunately, NLIDBs will not be widely used in businesses until their configuration is so easy that any computer professional could be able to perform it.

As a result of the difficulty of developing domain-independent NLIDBs that achieve recall rates closer to 100%, in recent years many researchers have focused their efforts in developing NLIDBs for specific domains and for languages different from English, some of which are described in [1, 7, 9, 10, 12, 14, 16, 17, 19, 24, 26, 31, 32].

Currently some NLIDBs that are considered domain-independent, have not shown supporting experimental results (CoBase [37] and InBASE [6]), have been tested with a single domain (NLPQC, STEP [22], C-PHRASE, and [23]), and other NLIDBs have been tested with databases that are not complex (ENLIGHT [13], NaLIX [18], DaNaLIX, and the system presented in [11]).

Table 1. Types of problems that occurs in queries

Cases	Problems
1	**Use of words or phrases of different syntactic categories**
1.1	Use of nouns or nominal phrases for referring to tables or columns. Example: List ***number of seats*** on D9S.
1.2	Use of verbs or verbal phrases for referring to tables or columns. Example: What time does flight 102136 ***leave*** ATL to DFW?
1.3	Use of prepositions or prepositional phrases for referring to tables or columns. Example: Give me an economy class flight ***from*** DFW ***to*** BWI one-way.
1.4	Use of adjectives or adjectival phrases for referring to tables or columns. Example: How ***fast*** can the Concorde fly?
1.5	Use of temporal adverbs. Example: List fares for all flights leaving ***after*** twelve o'clock noon from BOS to BWI.
1.6	Use of conjunctions. Example: Flights exiting Fort Worth ***and*** entering Dallas.
2	**Semantic ellipsis**
2.1	Lacking information of tables or columns. Example: List ***fares*** for all flights leaving after twelve o'clock noon from BOS to BWI. Note: there exist two columns related to the word "fare": "one_way_cost" and "rnd_trip_cost", thus it is not clear which of these columns is being referred to.
2.2	Lacking information of tables or columns referred to by some value. Example: How much is ***Delta*** flight ***539***?
2.3	Lacking information of tables or columns about the information requested. Example: ***All flights*** from ATL to SFO on Delta first class.
3	**Covering of the capability of SQL**
3.1	Queries that involve several tables. Example: Give me an ***economy*** class ***flight*** from DFW to BWI ***one-way*.** Note: the columns referred to by “economy”, “flight” and “one-way” belong to different tables.
3.2	Queries that involve aggregate functions and the Group By clause. Example: Which flight from Philadelphia to Dallas has the ***cheapest fare***?
4	**Other type of problems**
4.1	Search values that involve two or more columns. Example: Give me the hire date of the employee ***Margaret Peacock***. Note: the value "Margaret Peacock" involves two columns: “FirstName” and “LastName.”

Table 1. *(continued)*

4.2	Search values constituted by two or more words. Example: Give me the postal code and city of the supplier "***Exotic Liquids***".
4.3	Incomplete search values. Example: What is the name of the store where is "***the busy".***
4.4	Inexistent tables or columns. Example: Get me a ***date*** on flight 294 leaving ATL to Washington. Note: the table or column "date" does not exist in the database.
4.5	Spacing, punctuation and formatting mistakes. Example: Flights between SFO and Dallas between noon and ***5:00 P.M.*** Note: the ATIS database uses the military format for time.
4.6	Imprecise search values. Example: Show me the Atlanta to Dallas flights in the ***morning***.

5.2 Translation Process

The translation process is one of the most important aspects of a NLIDB, since it deals with understanding the query and translating it to a DB query language. In Sections 4.1 and 4.2 several translation approaches were described.

Translation issues are caused by problems that occur in queries. From an analysis carried out on query corpora involving three databases (ATIS, Northwind and Pubs), four general types of problems (i.e., those that usually are found in queries for most databases) were identified and classified [29]:

1. The use of words or phrases of different syntactic categories (such as nouns, verbs, adjectives, and prepositions) for referring to tables or columns of the database.
2. Semantic ellipsis that occurs when words that are necessary for clearly understanding the query are omitted.
3. Coverage of the capabilities of SQL, such as involving several tables of the database and the use of aggregate functions.
4. Other type of problems related to human errors, such as nonexistent information in the database, words that indicate imprecise values, etc.

The classification obtained and some examples are shown in Table 1. It is important to point out that the classification carried out according to the problems found in the corpora mentioned can be applied to most databases.

5.3 Desirable Features of a NLIDB

Obviously the main characteristic that any NLIDB must have is that it should answer correctly all the user queries. Additionally, a survey of the literature on ILNBDs revealed that, regarding the issues faced by ILNBDs, other characteristics that are considered the most important are the ones mentioned next.

Ease of configuration. The interface should be easily and swiftly configured (requiring a minimal user intervention). It should preferably be auto-configurable.

Operability. The system should be easy to use (friendly interface). Preferably, users should be guided through the querying process.

Authoring. The interface should have a tool that permits carrying out modifications of the knowledge base or data dictionary.

Habitability. The capacity of an interface for meeting user expectations, without surpassing the system linguistic capacity (limited grammar and linguistic coverage).

Transparency. The capabilities and limitations of the system should be evident to its users.

Robustness. Capacity of answering all the user queries.

Efficiency. The system should answer quickly.

Accuracy. The system should answer correctly all the queries.

Intelligence. The system should be able to answer deductive and temporal queries. It should also be able to improve its performance through information feedback obtained from user-system interaction.

Multimodality. Since most NLIDBs only support user-system interaction through keyboard, which is little practical, it is expected that the interface uses different means for inputting queries (such as speech, menus, forms, graphic objects, etc.) that permit users to obtain the advantages from the use of several input media.

Independence. The interface should be independent in four aspects:

1. Domain independence. The interface should be able to be ported to any database, so that it can answer queries concerning the domain at hand.
2. DBMS independence. The interface should extract information from the database regardless of the database management system (Oracle, Sybase SQL Server, PostgrestSQL, Microsoft SQL Server, Access, DB2, and Informix, and MySQL).
3. Natural language independence. The interface should be multi-lingual.
4. Hardware and software independence. The interface should permit porting it to any computing platform.

Handling of linguistic phenomena. The interface should consider the linguistic problems that may affect the meaning of the syntactic elements involved in user queries. Some of the most important are: anaphora, ellipsis and ambiguity.

6 Evaluation of NLIDBs

Currently, evaluating a NLIDB is a disturbing task, because there are no generally accepted evaluation benchmarks that can be used for comparing the performance of different NLIDBs, unlike other more mature research fields. Some characteristics of the evaluations of most NLIDBs are presented next, regarding databases used, test corpora and evaluation metrics.

6.1 Databases Used for Evaluation

Despite the large number of NLIDBs developed to date, there are only a few databases that have been used for evaluation.

Many of the NLIDBs use three databases developed by L. Tang and R. Mooney (used in [35]): Geobase, Restbase and Jobdata, relative to geography, restaurants and jobs domains. It is important to point out that these databases are not relational (they are programmed in Prolog) and their structure is simple (they contain from 1 to 8 entity classes). Researchers that have used these databases to test their NLIDBs, had to convert manually these databases to relational databases.

The first NLIDBs were designed for this type of databases, called deductive databases, because they contain rules for deducting facts from logic predicates. NLIDBs for deductive database never became popular because the vast majority of the databases used in businesses are relational.

Some of the databases used for testing are Northwind (traders database) [21], Pubs (books database) [21], CINDI (database of the virtual library of the Concordia University) [34], and in some cases student databases, which do not have a complex structure, and for which there does not exist a real-life or a corpus of complex queries for testing.

Finally, ATIS, a database that handles information on airline flights, is one of the most complex relational databases used for testing. Due to the complexity of its structure (27 tables, 123 columns and a corpus with 85% of elliptical queries), only few NLIDBs have dared to test their performance using this database.

6.2 Query Corpora Used for Evaluation

Similarly to the fact that there is no generally accepted benchmark database, neither does exist a benchmark corpus of queries for testing NLIDBs. From the query corpora mentioned in the literature, only the ATIS, Geobase, Restbase and Jobdata have corpora of real-life queries. Unfortunately, despite the large number of queries in ATIS (the largest corpus) many queries are repeated (i.e, they have the same structure but different search values), and the queries do not encompass all the types of problems that occur in queries.

The corpora for Geobase, Restbase and Jobdata are constituted by deductive queries because the databases are deductive. The queries of these corpora may have a high degree of logical complexity, but their databases do not have a complex structure (as compared with ATIS, for example).

The ATIS corpus, though it has many repeated queries, it has a large variety of queries that involve the most common interpretation problems found in real life. Moreover, the complexity of the database adds further complexity to some queries.

6.3 Evaluation Metrics

There exist many metrics that could be used for evaluating ILNBDs; however, the most used ones by researchers are the following:

$$\text{accuracy} = \frac{\text{total number of correct queries}}{\text{total number of parsed queries}} \times 100 \quad (1)$$

$$\text{recall} = \frac{\text{total number of correct queries}}{\text{total number of queries}} \times 100 \quad (2)$$

As shown by expression (1), accuracy is the percentage of correctly answered queries with respect to the number of translated queries. As shown by (2), recall is the percentage of correctly answered queries with respect to the number of queries input to the interface.

Unfortunately, there is no uniformity on what "correct query" means. In some NLIDBs evaluations, queries whose results include additional information to that requested in the query, are counted as "correct queries"; while in other NLIDBs evaluations "correct queries" include only those queries whose results include just the requested information. This lack of uniformity makes impossible to compare the performance of different NLIDBs.

6.4 Proposed Evaluation

As it has been mentioned, evaluating a NLIDB (by researchers and potential users of commercial interfaces) might be a disturbing task, since there is no established benchmark that permits standardizing the tests to carry out for a NLIDB. This situation makes impossible to compare the performance of different NLIDBs and to assess the progress of NLIDB technology.

As a result of the survey carried out on different state-of-the-art NLIDBs, we propose some testing means for guaranteeing a meaningful and useful evaluation of NLIDBs, which includes databases, query corpora, and evaluation metrics.

We propose to evaluate interfaces using corpora that include queries that involve the problems listed in Table 1, in order to find out how many of such problems are dealt with correctly by an interface.

Concerning databases, the use of at least three relational databases is proposed, whose structure has a complexity degree from medium to high (regarding number of tables and relations among tables). The ATIS database is a good candidate for the benchmark, since it has 27 tables and a complex relations structure. However, additional databases are needed, since the ATIS database is not adequate for formulating queries that involve certain problems; for example, queries that involve the Group By clause. By the way, the group of benchmark databases should permit formulating queries that involve all the problems listed in Table 1.

Additionally, we propose that only the recall rate should be used to assess performance, and to this end, each query in the benchmark corpora should be accompanied with the equivalent expression in SQL, in order to verify the correct translation of the query.

We claim that **the performance of a NLIDB can not be fully described by a single recall number, but by a group of numbers**, one for each aspect to evaluate. For example, a test of ELF carried out by us on a set of five complex queries

that involve the Group By clause yielded 0% of correctly answered queries; however, when tested with another query corpus ELF's recall rate is around 75% [28]. Therefore, to fully assess the performance of a NLIDB, a recall percentage should be measured for queries that involve words or phrases of different syntactic categories (nouns, verbs, adjectives, prepositions), a recall percentage should be measured for elliptical queries, another recall percentage for queries that involve Group By, and so on.

7 Concluding Remarks

This analysis of the state of the art in NLIDBs aims at showing the state of progress in this field to date. Our conclusion is that, despite the large number of ILNBDs that have been implemented for more than 50 years, these systems have not attained a high enough success rate (recall close to 100%) to make them widely used in business applications. We claim that for NLIDBs to be useful for such applications, they have to reach a success rate greater than or equal to the success rate of a good computing professional. Therefore, the good news for NLIDB researchers is that there is plenty of work ahead.

The main issues that hinder the advancement of NLIDB technology are: achieving high recall rates with domain-independent systems, developing easily configurable NLIDBs, and lack of benchmarks for evaluating NLIDB performance.

From the experience gained during the development of two NLIDB prototypes during the past 11 years [27, 28], we have arrived at the following conclusion: the process of correctly translating from a NL query to SQL is an extremely difficult problem, since it involves many issues (most of which are summarized in Table 1). We think that new efforts should focus on thoroughly studying and fully solving each of those problems.

Finally, like other more mature research fields, there is the need for an established benchmark that permits obtaining a meaningful evaluation of the performance of NLIDBs. To this end, we propose: the development of a benchmark consisting of two or three databases (ATIS could be a good candidate), the development of query corpora for the benchmark databases consisting of queries that involve the problems presented in Table 1, and the use an overall recall rate together with a recall rate for each of the different problems that occur in queries (those presented in Table 1).

References

1. Ahmad, R., Abid, M., Ali, R.: Efficient Transformation of Natural Language Query to SQL. In: Proc. 2nd Conference on Language and Technology, pp. 53–60 (2009)
2. Akerkar, R., Joshi, M.: Natural Language Interface Using Shallow Parsing. International Journal of Computer Science & Applications 5, 70–90 (2008)
3. Androutsopoulos, I., Ritchie, G., Thanisch, P.: Natural Language Interfaces to Databases - An Introduction. Natural Language Engineering 1, 29–81 (1995)
4. BBC, Bill Gates: Mouse is out, Touch Screen and Natural Language Interface are in (2008),

http://news.bbc.co.uk/player/nol/newsid_7170000/newsid_7174300/7174330.stm?bw=bb&mp=wm&asb=1&news=1&bbcws=1 (accessed June 10, 2012)
5. Bhootra, R.: Natural Language Interfaces: Comparing English Language Front End and English Query. Dissertation, Virginia Commonwealth University (2004)
6. Boldasov, M., Sokolova, E., Malkovsky, M.: User Query Understanding by the In-BASE System as a Source for a Multilingual NL Generation Module. In: Proc. 5th International Conference on Text, Speech and Dialogue, pp. 33–40 (2002)
7. Boonjing, V., Hsu, C.: A New Feasible Natural Language Query Method. International Journal on AI Tools 15, 323–330 (2006)
8. Cimiano, P., Haase, P., Heizmann, J.: Porting Natural Language Interfaces between Domains: an Experimental User Study with the ORAKEL System. In: Proc. 12th International Conference on Intelligent User Interfaces, pp. 180–190 (2007), doi:10.1145/1216295.1216330
9. Djahantigui, F., Norouzifard, M., Davarpanah, S., Shenassa, M.: Using Natural Language Processing in Order to Create SQL Queries. In: Proc. International Conference on Computer and Communication Engineering 2008, vol. 13, pp. 600–604 (2008)
10. El-Mouadib, F., Zubi, Z., Almagrous, A., El-Feghi, I.: Interactive natural language interface. Journal WSEAS Transactions on Computers 8, 661–680 (2009)
11. Giordani, A., Moschitti, A.: Semantic Mapping Between Natural Language Questions and SQL Queries via Syntactic Pairing. International Conference on Applications of Natural Language to Information Systems, pp. 207–222 (2010), doi:10.1007/978-3-642-12550-8_17
12. Jain, H., Bathia, P.: Hindi Language Interface to Databases. Journal of Global Research in Computer Science 2(4) (2011)
13. Joshi, M.: Intelligent Natural Language Interface. Dissertation, Maharshtra University (2006)
14. Karande, N., Patil, G.: Natural Language Database Interface for Selection of Data Using Grammar and Parsing. In: Proc. World Academy of Science Engineering Technology, vol. 59, pp. 484–487 (2009)
15. Kim, H.: A Dialogue-Based NLIDB System in a Schedule Management Domain. In: Proc. 33rd conference on Current Trends in Theory and Practice of Computer Science, pp. 869–877 (2007), doi:10.1007/978-3-540-69507-3_75
16. Kovacs, L.: SQL Generation for Natural Language Interface. Journal of Computer Science and Control Systems 2(18), 19–22 (2009)
17. Le, T.: A Frame-based Approach to Text Generation. In: The 21st Pacific Asia Conference on Language, Information and Computation, vol. 21, pp. 192–201 (2007)
18. Li, Y., Chaudhuri, I., Yang, H., et al.: DaNaLIX: a Domain-adaptive Natural Language Interface for Querying XML. In: Proc. 2007 ACM SIGMOD International Conference on Management of Data, pp. 1165–1168 (2007), doi:10.1145/1247480.1247643
19. Majeed, F., Shoaib, M., Ashraf, F.: An Approach to the Optimization of Menu-based Natural Language Interfaces to Databases. IJCSI International Journal of Computer Science Issues 8(4) (2011)
20. Martin, P., Appelt, D., Grozs, B., Pereira, F.: TEAM: An Experimental Transportable Natural-Language Interface. IEEE Database Eng. Bull 8(3), 10–22 (1985)
21. Microsoft Northwind and Pubs Sample Databases for SQL Server 2000 (2004), http://www.microsoft.com/en-us/download/details.aspx?id=23654 (accessed June 10, 2012)

22. Minock, M., Olofsson, P., Näslund, A.: Towards Building Robust Natural Language Interfaces to Databases. In: Kapetanios, E., Sugumaran, V., Spiliopoulou, M. (eds.) NLDB 2008. LNCS, vol. 5039, pp. 187–198. Springer, Heidelberg (2008)
23. Nguyen, A.K., Le, H.T.: Natural Language Interface Construction Using Semantic Grammars. In: Ho, T.-B., Zhou, Z.-H. (eds.) PRICAI 2008. LNCS (LNAI), vol. 5351, pp. 728–739. Springer, Heidelberg (2008)
24. Nihalani, N., Motwani, M., Silaka, S.: Natural Language Interface to Database using semantic matching. International Journal of Computer Applications 31, 29–34 (2011), doi:10.5120/3942-5552
25. Owda, M., Bandar, Z., Crockett, K.: Conversation-Based Natural Language Interface to Relational Databases. In: Proc. 2007 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology, pp. 363–367 (2007)
26. Pakray, P.: Keyword Based Multilingual Restricted Domain Question Answering. Dissertation. Jadavpur University (2007)
27. Rangel, R.A.P., Joaquín Pérez, O., Juan Javier González, B., Gelbukh, A., Sidorov, G., Rodríguez M., M.J.: A Domain Independent Natural Language Interface to Databases Capable of Processing Complex Queries. In: Gelbukh, A., de Albornoz, Á., Terashima-Marín, H. (eds.) MICAI 2005. LNCS (LNAI), vol. 3789, pp. 833–842. Springer, Heidelberg (2005)
28. Pazos R., R.A., Rojas P., J.C., Santaolaya S., R., Martínez F., J.A., Gonzalez B., J.J.: Dialogue Manager for a NLIDB for Solving the Semantic Ellipsis Problem in Query Formulation. In: Setchi, R., Jordanov, I., Howlett, R.J., Jain, L.C. (eds.) KES 2010, Part II. LNCS, vol. 6277, pp. 203–213. Springer, Heidelberg (2010)
29. Pazos R., R.A., González B., J.J., Aguirre L., M.A.: Semantic Model for Improving the Performance of Natural Language Interfaces to Databases. In: Batyrshin, I., Sidorov, G. (eds.) MICAI 2011, Part I. LNCS, vol. 7094, pp. 277–290. Springer, Heidelberg (2011)
30. Popescu, A., Armanasu, A., Etzioni, O., et al.: Modern Natural Language Interfaces to Databases: Composing Statistical Parsing with Semantic Tractability. In: Proc. 20th International Conference on Computational Linguistics, vol. 141 (2004), doi:10.3115/1220355.1220376
31. Rao, G.: Natural Language Query Processing Using Semantic Grammar. International Journal on Computer Science and Engineering 2(2), 219–223 (2010)
32. Sarhan, A.: A Proposed Architecture for Dynamically built NLIDB Systems. International Journal of Knowledge Based Intelligent Engineering Systems 13(2), 59–70 (2009)
33. Sri International, Air Travel Information Service (1990), `http://www.ai.sri.com/natural-language/projects/arpa-sls/atis.html` (accessed June 10, 2012)
34. Stratica, N.: Using Semantic Templates for a Natural Language Interface to the CINDI Virtual Library. In: Data & Knowledge Engineering, vol. 55, pp. 4–19. Elsevier Science Publishers, The Netherlands (2005), doi:10.1016/j.datak.2004.12.002
35. Tang, L.R., Mooney, R.J.: Using Multiple Clause Constructors in Inductive Logic Programming for Semantic Parsing. In: Flach, P.A., De Raedt, L. (eds.) ECML 2001. LNCS (LNAI), vol. 2167, pp. 466–477. Springer, Heidelberg (2001)
36. Templeton, M., Burget, J.: Problems in Natural-Language Interface to DBMS with Examples from EUFID. In: ANLC 1983 Proc. First Conference on Applied Natural Language Processing, pp. 3–16 (1983), doi:10.3115/974194.974197
37. Zhang, G., Chu, W., Meng, F., et al.: Query Formulation from High-Level Concepts for Relational Databases. In: Proc. 1999 User Interfaces to Data Intensive Systems, pp. 64–75 (1999)

A Genetic Algorithm for the Problem of Minimal Brauer Chains

Arturo Rodriguez-Cristerna and Jose Torres-Jimenez

Information Technology Laboratory, CINVESTAV-Tamaulipas Km. 5.5 Carretera Cd. Victoria-Soto la Marina, 87130, Cd. Victoria Tamps., Mexico
arodriguez@tamps.cinvestav.mx, jtj@cinvestav.mx

Abstract. Exponentiation is an important and complex task used in cryptosystems such RSA. The reduction of the number of multiplications needed during the exponentiation can significantly improve the execution time of cryptosystems. The problem of determining the minimal sequence of multiplications required for performing a modular exponentiation can be formulated using the concept of Brauer Chains.

This paper, shows a new approach to face the problem of getting Brauer Chains of minimal length by using a Genetic Algorithm (GA). The implementation details of the GA includes a representation based on the Factorial Number System (FNS), a mixture of Neighborhood Functions (NF) and a mixture of Distribution Functions (DF). We compare the proposed GA approach with another relevant solutions presented in the literature by using three benchmarks considered difficult to show that it is a viable alternative to solve the problem of getting shortest Brauer Chains.

Keywords: Addition Chains, Genetic Algorithms.

1 Introduction

Exponentiation is an important and complex task used in cryptosystems such as RSA. The reduction of the number of multiplications needed during the exponentiation can significantly improve the execution time of cryptosystems [2]. The problem of determining the minimum operations required for the exponentiation of x^n have been searched with different strategies and a *naive* solution is to apply a sequence of *n-1* multiplications of x such that $x^n = x \cdot x \cdot \ldots \cdot x$. For example, if we want to compute x^{23} using the naive way we have to apply 22 multiplications. Another possibility to compute x^n is by applying the *binary method* [11], which is showed in a recursive description in the Equation 1.

$$x^{\alpha} \begin{cases} x & \text{if } \alpha = 1 \\ x^{\frac{\alpha}{2}} \cdot x^{\frac{\alpha}{2}} & \text{if } \alpha \text{ is even} \\ x^{\alpha-1} \cdot x & \text{otherwise} \end{cases} \quad (1)$$

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 481–500.
springerlink.com

For the example of the exponentiation x^{23}, using the *binary method* we only need to perform 7 multiplications, as can be seen next:

$$xx = x^2, x^2x^2 = x^4, xx^4 = x^5, x^5x^5 = x^{10}, xx^{10} = x^{11}, x^{11}x^{11} = x^{22}, xx^{22} = x^{23}$$

But it is possible to reduce further the number of multiplications needed for the exponentiation than those used by the *naive* way or the *binary method*, and there have been reported different strategies to simplify this complex task like the *m-ary method* and the *window-based* method [5,11].

Also, the problem of determining the minimal sequence of multiplications required to perform an exponentiation can be formulated with the concept of Addition Chain (AC). The addition chain-based methods for exponentiation use a sequence of positive integers such that the first number of the chain is 1 and the last number is the exponent *n*. Therefore, the length of an addition chain *n* is equal to the corresponding number of multiplications required for the computation of x^n. Thus, the smallest of such multiplications, given by the chain length *l(n)*, makes the exponentiation task faster.

Given a positive integer *n*, an AC is a sequence of integers $C = \{\alpha_0, \alpha_1, \dots, \alpha_r\}$ such that:

$$\alpha_i = \begin{cases} 1 & \text{if } i = 0 \\ \alpha_j + \alpha_k & \text{if } i > 0 \text{ for some } j, k < i \end{cases} \quad (2)$$

The length of the AC is $|C| = r$, the chain length *l(n)* is the minimal length of all possible ACs for *n*, and the smaller addition chain is called Minimal Addition Chain (MAC).

An example of an AC is the sequence of numbers shown in the Figure 1, where can be see how each non root member in the chain are composed by two previous members and the element *r* is an integer equal to *n*.

In an AC every set of values *{j,k}* is called step, and according with their properties along the chain, the steps and the complete chain takes some particular name. A first instance is if *j* and *k* are equal to *i*-1, it its called "double step". Another instance is when *j* is equal to *i*-1 and *k*<*j*, i.e. if one of the addends is the previous member, then it is called a star step, and "an addition chain that consists entirely of star steps is called a star chain" [20] or Brauer Chain (BC) [12] in honor of Brauer (1937). We denote $l^*(n)$ the minimal BC length for a number *n*, and where a BC *C* has the smallest length *r* for a number *n* we can say that *C* is a Minimal Brauer Chain (MBC) for *n*.

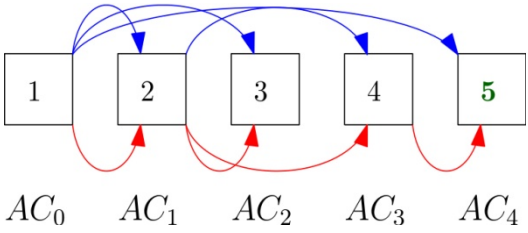


Fig. 1. Example of an AC for the number 5

An example of a BC is (1,2,4,5,9,18,23), which leads to the following scheme for the computation of x^{23} using only 6 multiplications:

$$xx = x^2, x^2x^2 = x^4, xx^4 = x^5, x^5x^4 = x^9, x^9x^9 = x^{18}, x^{18}x^5 = x^{23}$$

In this paper we propose a Genetic Algorithm (GA) to face the problem of getting MBCs, using a representation based on the FNS, a mixture of neighborhood functions and a mixture of distribution functions. The remaining of this paper is organized as follows. Section 2 gives a brief description of a variety of approaches proposed to find MACs and MBCs, Section 3 describes our proposed approach and the parameter configuration used, Section 4 shows the results obtained and finally Section 5 gives the reached conclusions.

2 Relevant Related Work

In the last years, it has been shown that metaheuristic strategies find near optimal solutions for a wide variety of combinatorial problems in a reasonable time. In this section are described some approaches that have been designed for the problem of finding MACs and MBCs.

Bleichenbacher and Flammenkamp [2] search for MACs by using Direct Acyclic Graphs (DAG) to represent the chains and a BackTracking (BT) search to construct the graph. Their backtrack approach uses a stage where special cases of addition chains are replaced with another equivalent ones in order to get smaller ACs.

Thurber in 1999 [20] proposed a BT algorithm to find MACs, by using a representation based on a tree of k levels and branch and bound methods to explore the tree of a size at least $k!$.

Nedja and Moruelle in 2002 [14] designed an approach based in the m-ary method using a parallel implementation to compute MACs by decomposing the exponent, in its binary representation, in blocks (also called windows) containing successive digits of ones that results in variable length zero-partitions and one-partitions. They found ACs with a lower number of elements than the binary method does.

Nedja and Moruelle in 2003 [17] used large windows inside a genetic algorithm. Their optimal parameter settings found were: 50 individuals; a double-points crossover; a mutation rate between 0.4 and 0.7; and a mutation degree of about 1% of the last value in the binary encoding sequence.

Cruz-Cortes *et. al* in 2005 [5] proposed a GA to solve the problem of finding MACs with the following features: each solution is represented as a sequence of integers, where each gene is related to one step of the AC, so each time that their algorithm apply a crossover operator it have to assure that the resulting sequence is a valid AC and the fitness is the AC length. The remaining features of their proposed GA are: a "non elitist" survivor selection; a population size of 100; a number of generations of 300; a mutation rate of 0.5; a selection of parents pairs to be recombined by a binary tournament; and a one point crossover. Their results suggest that this kind of algorithm can be a good alternative to solve the problem of finding MBCs.

Nedja and Moruelle in 2004 [15] and 2006 [16] proposed Ant Colony Systems (ACS) to obtain MACs that use a bi-dimensional triangular array to store the ACS global memory. The local memory is divided in two parts: a vector of length n and the fitness of the path traveled by the ant to construct the AC.

Gelgi and Onus in 2006 [10], proposed some heuristics approaches for the problem of getting an MBC. They present five approaches: three greedy heuristics and two dynamic programming approaches. They found empirically, that their dynamic heuristic approach has an approximation ratio (obtained length / minimum length) of 1.1 with $0 \leq n \leq 20000$.

Cruz-Cortes *et. al* in 2008 [4] presented an algorithm using the metaheuristic approach known as Artificial Immune System (AIS) to tackle the problem of finding short BCs, which uses a cloning operator and a hypermutation operator (the hypermutation operator is inversely proportional to the clones fitness) over the best solutions; and an "elitist" selection. The values of the parameter used are: a population size of 45; a number of best individuals to be cloned of 11; a number of replaced antibodies of 4 and 25 iterations of the main cycle. Also they combine their AIS system with a slide-window method to deal with large numbers. They use a function named *fill* to search for a valid BC, but its important to say that they do not mention about how much effort is done inside the *fill* function either in terms of time or evaluations.

Osorio-Hernandez *et. al* in 2009 [18] design a GA to find minimal length Brauer Chains, despite the name of the work ("A Genetic Algorithm with repair and local search mechanisms able to find minimal length addition chains for small exponents"). They use a repairing process; local search; and integer representation. They were able to find short BCs with the following parameters: population size of 200; maximum number of generations of 300; and binary tournament selection. However they don't were able to find all minimal length BCs with only one configuration and they had to use two different configurations to find some of them. Other important aspects of this work are that: the parameter configuration was obtained by trial and error, and there is no mention about how many operations are used or how much time is used inside the *fill* function (which is used to search for a valid BC).

Dominguez-Isidro *et. al* in 2011 [6,7] designed an algorithm to face the problem of finding MBCs using an evolutionary programming approach, where mutation is the only variation operator. Each individual k in the current population generates by mutations t mutants, and the best of them is chosen as k's offspring. As parameter values they use: population size of 100; maximum number of generations of 230; 4 mutants per individual; and a survivor selection based on a tournament of size 10.

Jose-Garcia *et. al* in 2011 [13] designed an interesting algorithm based on Simulated Annealing (SA) approach for the problem of finding MBCs, although the name of the work mentions addition chains, which uses a mixture of 4 neighborhood functions and a representation based on a chain in FNS. This algorithm works only with one solution at a time due to Simulated Annealing specification and the parameter values were: an initial temperature of 10, a cooling rate of .85 and a length of a Markov chain of 10. Also, with the purpose of reducing consuming time of the experimentation they performed and used a parallel test scenario.

Rodriguez-Cristerna *et. al* in 2011 [19] reports a Mutation-Selection (MS) algorithm, based on the general scheme of an evolutionary algorithm but without the recombination stage to find MBCs. This algorithm uses: a representation based on FNS, which is adequate to apply genetic operators and always generate a valid solution; a fitness function based in the n achieved and the length of the chain; a "non elitist" survivor selection; $3\log_2 n$ parents; $7\log_2 n$ mutated children per each parent; and $n \times 1000$ iterations.

Clift in 2011 [3], designed a BT strategy based on a graph representation and a novel prune criterion. This approach was able to find all $l(n)$ with $n \leq 2^{32}$ using a computational time of about a month using 12 processors.

3 Proposed Approach

3.1 Genetic Algorithm

In order to present the GA proposed, a brief description of how it works is given. GA uses one or more points in the search space, called *parent-points*, to generate multiple points through recombination and then its applied a mutation process (many times implemented as a local search procedure) to the new points. The generated points are called *children-points* and are evaluated in search of an optimal point. When no optimal point is found, the whole process is repeated replacing the *parent-points* with some points of the population according to a rule, which is called survivor selection. This cycle is repeated until an optimal point is found or certain termination criterion is met.

The algorithm proposed is based on the general scheme of an evolutionary algorithm [8], and its pseudocode is showed below.

```
General scheme of the GA algorithm proposed.
INITIALIZE parents
EVALUATE parents
REPEAT
  SELECT pairs of parents
  RECOMBINE pairs of parents
  MUTATION through local search with the
resulting offspring
  EVALUATE new candidates
  parents =  survivor selection
UNTIL the number of evaluations functions are
done
```

Contextualizing the GA for MBC computation, we have to address the next points:

– The representation and the search space used by the proposed algorithm (described in Subsection 3.2).

- The distribution functions used to select a position in a BC (in FNS representation) and the distribution function used to pickup values for a given position in the BC (described in Subsection 3.3).
- The *survivor selection* methods used (described in Subsection 3.4).
- The *children-points* generated through selection and recombination (detailed in Subsection 3.5).
- The local search process with children-points using Neighborhood Functions (detailed in Subsection 3.6).
- The evaluation function used to measure the quality of the potential solutions and the termination condition (described in Subsection 3.7).

3.2 Representation and Search Space

The representation used is based on the FNS and the search space is $r!$ where r is the length of the BC. We use a lower bound denoted by φ and an upper bound denoted by ψ, defined in the Equations 3 and 4 respectively. The lower bound is defined as the barrier of minimum number of multiplications that are needed if we could apply only double steps and the upper bound is defined as the maximum number of multiplications needed by the binary method.

$$\varphi = \lfloor \log_2(n) \rfloor \tag{3}$$

$$\psi = 2 \cdot \lfloor \log_2(n) \rfloor \tag{4}$$

In FNS we can describe a BC C with a chain of numbers, that we refer as C', by taking a value from the set $\{0,1,...,i\text{-}1\}$ for each node of C' with an index position i greater than 0, because the representation in FNS for the node 0 of C has no value. We can rebuild the original BC from the chain C' applying the Equation 5. The Figure 2 shows an example of how to represent a BC C for n=23 using a chain of numbers C' in the FNS.

It's necessary to mention that the population is initialized only with double steps nodes, and the chain C' is always translated to a BC until the last node which its translation is equal or lower than n.

index	0	1	2	3	4	5	6
C	1	2	4	5	9	18	23
C'	-	0	1	0	2	4	3

Fig. 2. Example of representation of a BC C for n=23 in a chain of numbers C' using the FNS

$$C(i) = \begin{cases} C(i-1) + C\big(C'(i)\big) & \text{if } i > 0 \\ 1 & \text{if } i = 0 \end{cases} \tag{5}$$

3.3 Distribution Functions for Selecting a Position in the BC

The distribution functions allows to select an i position in a C' chain, such that $1 \leq i \leq r$, with a non Gaussian distribution in two steps: first is calculated a random value x with $0 \leq x \leq \tau$ (Equation 6) and second is used one of the two distribution functions in Equation 7 to calculate the selected i position.

The main purpose of the two distribution DF_1 and DF_2 is the selection of a point in the BC. DF_1 allows to select with more probability the left most positions while the distribution function DF_2 allows to select with more probability the right most positions. The importance of using DF_1 and DF_2 is to establish a balance between exploration and exploitation. DF_1 enables exploitation and DF_2 enables exploration. The distribution functions DF_1 and DF_2 also has the property to select with a small probability a higher position, but not more than one unit far from the chain length, which increases the possibility of modifying the chain length.

$$\tau = \frac{(r+1) \times (r+2)}{2} \tag{6}$$

$$i = \begin{cases} DF_1 = \left\lfloor \frac{1+\sqrt{1+8(x+1)}}{2} \right\rfloor - 1 \\ DF_2 = r + 3 - \left\lfloor \frac{1+\sqrt{1+8(x+1)}}{2} \right\rfloor \end{cases} \tag{7}$$

Once defined the criteria for selecting a position within the BC, we are able to define the criterion to choose a valid value for the positions. The steps that follow a neighborhood function to select a valid value for an i position in a BC are two: first is selected a random x' such that $0 \leq x' \leq \tau'$ (**Equation 8**); and second is used the distribution function DF_1. This methodology ensure that the i chain member has a value from the set {0,1,...,i-1} and also has the property to set in the i position a double step with high probability.

$$\tau' = \frac{(i+1) \times i}{2} \tag{8}$$

3.4 Survivor Selection

It is important to select the best survivor selection type because it allows to manage the memory of the GA in different ways. For that reason we explore the use of three types of survivor selection: the first one is called (μ+λ) and takes the best points from the set of *parent-points* and the *children-points* using a random criterion as a tiebreaker.

The second one is called (μ+λ) with no repetitions, which takes the best points from whole population but discards the points with the same fitness. In the cases where the *parent-points* can not be full filled with different individuals, the *parent-points* are completed with the the best remaining individuals.

The third one is called (μ,λ) and takes the best points from the set of the *children-points*, in case of ties its used a random selection criterion. The *parent-points* are randomly replaced in case that they were fewer *children-points* than *parent-points*.

3.5 Selection and Recombination

The selection of parents pairs to be recombined were explored with the following strategies: random selection, tournament of size two and tournament of size three. In the random selection both parents are randomly chosen. For the tournament of size two, there are selected two groups of two parents points and only the best one of each group is selected to be recombined. Finally for the tournament of size three, there are selected two groups of three parents points and the best parent point for each group is selected.

For the recombination, we use a simple recombination of one point scheme. It means, that if we choose the point t for the recombination, the *children-point* C''_1 is constructed with the first t elements of the *parent-point* C'_1 and the remaining elements of the *parent-point* C'_2, in complement, the *children-point* C''_2 is constructed with the first t elements of the *parent-point* C'_2 and the remaining elements of the parent-point C'_1. This process is exemplified in the Figure 3.

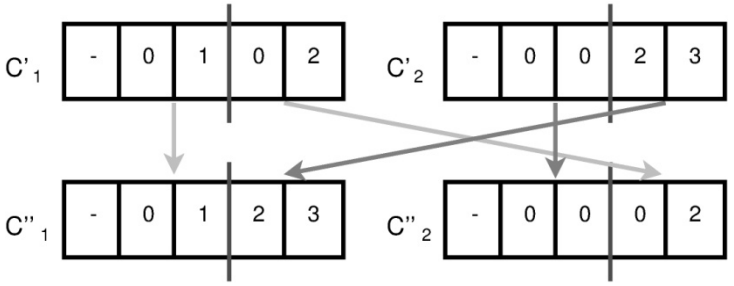


Fig. 3. Example of recombination of *parents-point* C'_1 and C'_2

To pickup the t point, we explore five mixtures of use of the distribution functions DF_1 and DF_2:

1. 0% DF_1 - 100% DF_2
2. 25% DF_1 - 75% DF_2
3. 50% DF_1 - 50% DF_2
4. 75% DF_1 - 25% DF_2
5. 100% DF_1 - 0% DF_2

3.6 Neighborhood Functions and Local Search

Inside of the local search we explore the use of a mixture of four neighborhood functions, which are described below:

- $NF_1(s)$. Select a random index position i from s, and pickup another FNS value different from the original with the use of DF1, the change is made only if the resulting BC is better than the original.
- $NF_2(s)$. Select a random index position i from s, and pickup a FNS value different from the original in i. Then select another different random position j from s, and pickup a FNS value different from the original in j. The change is applied only if the resulting BC is better than the original.
- $NF_3(s)$. Select a random index position i from s, and pickup the best FNS value.
- $NF_4(s)$. Select a random index positions i and j from s, and pickup the best FNS values.

To select the i and j positions inside the pairs of neighborhood functions NF_1- NF_3 and NF_2- NF_4, we test the use of a mixture of distribution functions as in Section 3.5, because the use of different distribution functions modify the behavior of the neighborhood functions. For example, the use of the distribution function DF_1 implies exploitation while the use of the distribution function DF_2 implies exploration. Additionally NF_1 and NF_2 use the DF_1 to pickup a value for the selected position.

The local search consists in applying iteratively the neighborhood functions. To explore the behavior of the amount of resources wasted during the local search in the algorithm, we try five limits of number of iterations: 1, $0.10 \cdot r+1$, $0.20 \cdot r+1$, $0.50 \cdot r+1$, $r+1$.

3.7 Evaluation Function and Termination Condition

The evaluation function ς used in the GA is shown in Equation 9.

$$\varsigma = |n' - n|(r + 1) + r \tag{9}$$

In Equation 9 r represents the length of the BC that is evaluated, n' is the value of the evaluated chain in its r position and n is the searched value. In this way, solutions whose n' is near to the n searched, have an evaluation only determined by its length, but solutions whose difference between n' and n, have an evaluation determined by its difference multiplied by its length plus its length which mean that those solutions are penalized, making possible to discriminate between the quality of chains.

The termination condition is met when certain limit number of evaluation functions are done. We established the limit of evaluations functions as $\log_2(n)^2 * 26^{\log 2(\log 2(n))}$. The evaluation functions are considered according to each time that is checked the fitness in the chain, thus NF_3 and NF_4 use a number of evaluation functions that grows with the position or positions that their are

modifying while NF_1 and NF_2 only use one evaluation function each time that are called. This criterion make us able to distinguish the potential of the different configurations of the full system using the same computational resources.

3.8 Parameters Used

As a result of a fine-tuning process the best configuration obtained was set to:

- 0% of use DF_1 and 0% of use DF_2 inside NF_1-NF_3.
- 100% of use DF_2 and 0% of use DF_2 inside NF_1-NF_3.
- 25% of use DF_2 and 75% of use DF_2 inside inside of the recombination strategy.
- r+1 iterations of neighborhood functions during the local search.
- Random *parent-point* selection for the recombination strategy.
- $\lceil 3\log_2 n \rceil$ *parent-points* and *children-points*.
- 20% of use of NF_3 and 80% of use of DF_4.

3.9 Implementation Note

The proposed GA was coded using C language and compiled with GCC 4.3.5 with -O3 optimization flag. The algorithm has been run on a single core of a cluster with 4 processor six-core AMD® 8435 (2.6 Ghz), 32 GB RAM, and Operating System Red Hat Linux Enterprise 4.

4 Results

In order to measure the performance of the GA proposed, the first experiment was to search the MBC for a benchmark made of 27 different numbers n that satisfy the restriction of be $c(r)$, available in [9] along a database of many $l(n)$ values computed by Neill Clift [3], and to get statistical significance each experiment was tested 31 times with different random seeds. $c(r)$ is the smallest number which have an addition chain of lenght r, and the set of numbers that accomplish the $c(r)$ property are a special class of numbers for which is hard to obtain the minimum addition chains.

The main results of the first experiment are shown in Table 1, where we can see the set of n's tried; the minimal, average and maximum length obtained; the average and standard deviation of the time to conclude each experiment; and the hits (times that a MBC was found). It can be seen that for the first experiment (Table 1) all the MBC was obtained and the average length and the standard deviation obtained indicates the reliability of the GA. The behavior of the proposed GA for the first experiment also can be seen in the Figure 4 that shows the difference between minimum, average and maximum length with their respective standard deviation versus optimal lengths, where is evident that our approach can get easily most of the Brauer Chains with $l^*(n)<22$ and for the other

cases the average length obtained are not too far from the optimal. Table 3 shows the MBCs found for $n \in \{2211837, 4169527, 7624319, 14143037\}$ where can be verified the restriction of the BC's.

A second benchmark, taken from [18], is composed by 20 different numbers hard to optimize, because its minimal addition chains currently has not been generated by deterministic methods (i.e. binary method or window-based) or some other non deterministic methods. For the second benchmark, every instance was tried 31 times with different random seeds and the results are showed in Table 1 where can be seen the set of *n*'s tried; the minimal, average and maximum length obtained; the average and standard deviation time to finish the tests; and the hits. In the second benchmark only for one instance was not obtained its MBC (3926651), however the minimum size found is not to far from the optimal, in fact it is only at one unit of distance. Additionally, with the complete results we observe that the worst cases and the average cases are not far from the optimal solution, getting in the worst case BC's only three units far from the optimal and in average at 1.4 units plus the optimal length. One of the MBCs found for the numbers 3459835, 3493799, 3704431 and 3922763. are presented in Table 4.

In order to contrast the obtained results, Table 5 presents a comparison between the obtained results of our proposed approach in the first experiment and other five approaches reported in the literature: BackTracking [2], Genetic Algorithm [5], Artificial Inmune System [4], Mutation Selection [19] and Simulated Annealing [13]. The comparison is made using the minimum length obtained, the average time (in seconds) and the average hits, where the symbol $\oplus$ means that it is a worse result than the one given by our approach in both time and/or number of hits. It can be observed that our results have the same or better quality than the procedures presented in the state of the art. It could be useful a more detailed comparison against one of the best reported works [4], however they don't report time spent; number of evaluation functions; or number of hits for each case.

A third experiment, consisted in calculate the accumulated addition chain length for the range of exponents $[1,Z]$ for $Z \in \{512, 1000, 1024, 2000, 2048, 4096\}$, where each exponent was tested 31 times with different random seeds. The best results are calculated according to Equation 10 and the worst case according to Equation 11, where GA is a function that receives an integer number and returns a BC generated through our proposed approach. The average results are computed according to Equation 12 and the standard deviation is based on the Equation 13. *CV* is the coefficient of variation calculated as *CV=average/stddev* and δ is the difference between the best case and the optimal case.

$$\sum_{i=1}^{i=Z} \min \{GA_j(i) | 0 \leq j \leq 30\} \tag{10}$$

$$\sum_{i=1}^{i=Z} \max \{GA_j(i) | 0 \leq j \leq 30\} \tag{11}$$

$$average\left\{AS_j = \sum_{i=1}^{i=Z} GA_j(i)\min|0 \le j \le 30\right\} \tag{12}$$

$$stddev\left\{AS_j = \sum_{i=1}^{i=Z} GA_j(i)\min|0 \le j \le 30\right\} \tag{13}$$

Table 6 presents the results of the third experiment where it is observed the optimal case calculated using the public database of addition chains of Achim Flammenkamp [9]; the best case; the worst case; the average case; and the standard deviation. Additionally it is presented the accumulated time spent for all the cases together with the standard deviation and the difference between the optimal case and our best case (δ). Our results for $Z \in \{512, 1000, 1024\}$ exhibits a δ=0, meaning that all the Minimal Brauer Chains were constructed, but δ is greater than 0 for $Z \in \{2000, 2048, 4096\}$ where the list of numbers (with their corresponding $l(n)$) for which the GA was not able to construct their MBC were: (1063,13), (1143,13), (1387,13), (2011,14), (2087,14), (2091,14), (2135,14), (2151,14), (2251,14), (2285,14), (2507,14), (2617,14), (2647,14), (2774,14), (2957,14), (3199,15), (3559,15), (3707,15), (3801,15), (3803,15), (3819,15), (3829,15), (4051,15) and (4070,15). However this doesn't mean that our approach with different configurations cannot reach that MBCs. Also is visible in Table 6 that the best case is very close or equal to optimum case, the average case is also competitive and the coefficient of variation is small which means that our results have a high degree of confidence and are not dependent on the random seeds.

Table 7 allows to see a comparison between classic approaches ([1]) and the GA proposed for the calculation of the accumulated addition chains where n is in the range [1,1000]. There is a remarkable difference between the accumulated addition chain computed by the binary and dichotomic method versus the optimal accumulated sum, however both are fast options to compute addition chains since they are deterministic approaches that do not imply any heuristic search or backtrack.

A comparison between the GA proposed and the more recently metaheuristics are presented in Table 8 and it is done using the best results for an accumulated addition chain in the range $[1,Z]$ for $Z \in \{512, 1000, 1024, 2000, 2048, 4096\}$, where Δ is the difference between the best reported results and our best result. The metaheuristics chosen for the comparatison, that are part of the best strategies to construct Minimal Addition Chains or Minimal Brauer Chains, are: AIS [4], SA [13], EP [6,7], GA [5] and GA [18].

In Table 8 we can see with the symbol ⊕ when the difference between the accumulated addition chain and the results of other strategy is positive, meaning that our result is better. It can be seen that the quality of our result are better than the best reported in the state of art, however it is not possible to conclude that our approach have a better efficiency than the approaches used for the comparison due to lack of necessary information. Here it's necessary to remark the difference that we only use one configuration for the reported results, and the complexity of the

system as the number of times that is checked the fitness of a BC, taken as as evaluation function or the test of how long is the chain and what is the obtained number in the r position, is limited to $\log_2(n)^2 * 26^{\log 2(\log 2(n))}$. A more detailed comparison can be done between our approach and other approaches, but they do not report accumulated time or how many operations are done inside the working chain or chains. It is need to say that: AIS [4], EP [6,7] and GA [18] do not report evaluations functions used, however in [6,7] are used 92000 comparison between individuals and in GA [18] are used 300000 comparisons.

But it is not only important the best results, it is also is important how much confidence we can have in the average case, taken as the coefficient of variation. For this reason we have made a comparison using the coefficient of variation between the GA proposed and the metaheuristics compared in Table 8. The comparisons of coefficients of variation for the accumulated addition chains lengths is presented in Table 9, where can be observed that all approaches have a high degree of confidence, and the one with the highest confidence is EP [6,7]. Also, with $\oplus$ are marked the coefficients of variation that have less confidence than our results.

Table 1. Summary of results to compute MBC for some n for which is hard to find their MBC (part 1)

id	n	Min. r	Average r	Max. r	Std. Dev. r	Average time (s)	Std. Dev. time (s)	hits
1	7	4	4	4	0	0	0	31
2	11	5	5	5	0	0.0003	0	31
3	19	6	6	6	0	0.0010	0	31
4	29	7	7	7	0	0.0023	0	31
5	47	8	8	8	0	0.0055	0	31
6	71	9	9	9	0	0.0107	0.0001	31
7	127	10	10	10	0	0.0259	0.0021	31
8	191	11	11	11	0	0.0442	0.0006	31
9	379	12	12	12	0	0.1031	0.0007	31
10	607	13	13	13	0	0.1760	0.0017	31
11	1087	14	14	14	0	0.3241	0.0076	31
12	1903	15	15	15	0	0.5503	0.0034	31
13	3583	16	16.0323	17	0.1767	0.1767	0.0075	30
14	6271	17	17	17	0	1.5671	0.0117	31
15	11231	18	18	18	0	2.4806	0.0176	31
16	18287	19	19	19	0	3.5497	0.0250	31
17	34303	20	20.1613	21	0.3678	5.5663	0.0553	26
18	65131	21	21	21	0	8.5120	0.1364	31
19	110591	22	22.2903	23	0.4539	11.9941	0.1586	22
20	196591	23	23.0323	24	0.1767	17.0883	0.2624	30
21	357887	24	24.1613	26	0.4470	24.0470	0.3448	27
22	685951	25	25.6129	27	0.5493	34.3607	0.6828	13
23	1176431	26	27.0645	29	0.6689	46.5622	1.0647	5
24	2211837	27	27.9677	30	0.6949	63.2554	0.8555	7
25	4169527	28	28.3226	29	0.4675	86.3890	2.1981	21
26	7624319	29	30.0968	34	1.1175	114.9951	2.2392	11
27	14143037	30	30.9677	32	0.6949	152.4645	2.2392	8

Table 2. Summary of results to compute MBC for some n for which is hard to find their MBC (part 1)

id	n	Min. r	Average r	Max. r	Std. Dev. r	Average time (s)	Std. Dev. time (s)	hits
1	2948207	27	28.2258	30	0.5512	73.3253	0.9955	1
2	3093839	27	28.2903	30	0.6812	75.2953	0.9979	3
3	3167711	27	27.9355	30	0.7155	74.5330	1.0593	7
4	3182555	27	27.7419	28	0.4376	75.1344	0.7311	8
5	3190511	27	27.9355	29	0.5643	74.7987	1.0623	6
6	3230591	27	27.9677	29	0.3094	75.5409	0.7089	2
7	3234263	27	27.3548	29	0.5983	75.3779	0.7897	22
8	3235007	27	28.0645	30	0.6188	75.8259	0.9108	2
9	3243679	27	28.0000	29	0.5080	75.8844	0.8510	4
10	3243931	27	28.0000	29	0.5680	75.7768	0.8483	5
11	3266239	27	28.3548	30	0.6500	76.5093	1.1414	1
12	3287999	27	28.1935	30	0.6435	76.5596	1.0754	3
13	3325439	27	28.3871	30	0.6052	77.0552	1.0106	1
14	3352927	27	27.6774	29	0.5895	76.5714	0.9955	12
15	3440623	27	28.0968	30	0.7769	78.6647	0.9695	7
16	3459835	27	28.0968	29	0.3898	78.6540	0.8115	1
17	3493799	27	28.0968	29	0.5301	79.6454	0.9094	3
18	3704431	27	27.9355	30	0.5643	81.2270	1.3532	5
19	3922763	27	28.4839	30	0.6154	84.6750	1.3315	1
20	3926651	28	28.3548	29	0.4785	84.2965	1.4677	0

Table 3. Some MBCs found (part 1)

n	MBC found	$l^*(n)$
2211837	1→2→4→8→16→32→33→65→130→260→520→ 1040→1073→2146→4292→8584→8617→17234→34468→ 68936→137872→137937→275809→551618→ 552691→1105382→1106455→2211837	27
4169527	1→2→3→6→7→14→28→56→112→113→226→ 452→904→1808→3616→7232→7238→14476→28952→ 28955→57910→115820→231640→260595→ 521190→1042380 →2084760→4169520→4169527	28
7624319	1→2→4→6→7→14→28→30→58→116→232→464→ 928→1856→3712→7424→14848→29696→29724→ 59448→118896→237792→475584→951168→953024→ 1906048→1906078→3812156→7624312→7624319	29
14143037	1→2→3→5→10→13→26→52→104→208→ 416→832→858→1716→1726→3452→6904→ 13808→27616→55232→110464→220928→441856→ 883712→1767424→3534848→3535706→3535758→7071516→ 14143032→14143037	30

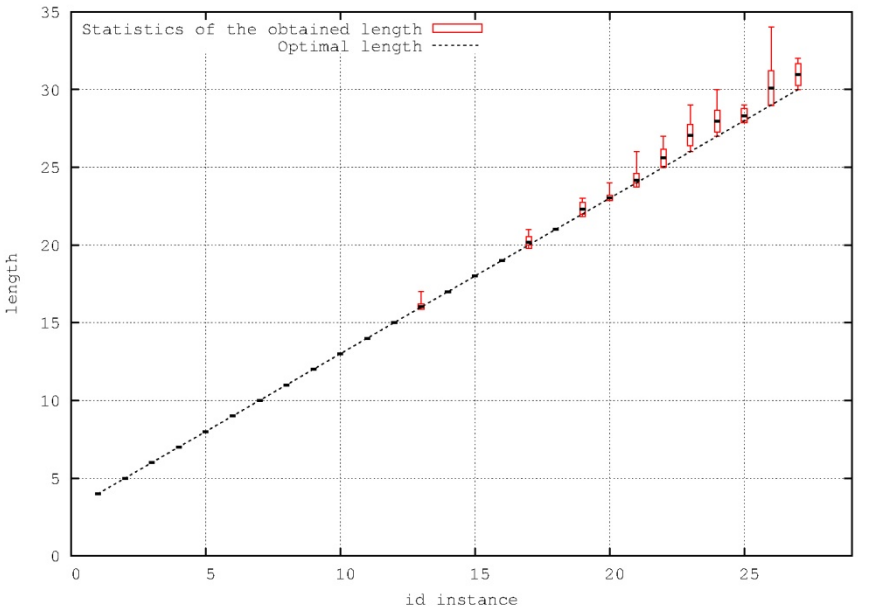


Fig. 4. Comparison between minimum, maximum and average found versus optimal lengths for some n's for which is hard to find their MBC (part 1)

Table 4. Some MBCs found (part 2).

n	MBC found	$l^*(n)$
3459835	1→2→4→8→16→32→33→66→99→ 198→396→792→1584→1683→1691→3374→6748→ 13496→26992→53984→107968→215936→216332→432268→ 864536→1729072→3458144→3459835	27
3493799	1→2→3→5→8→13→26→27→53→106→ 212→424→848→1696→3392→6784→13568→27136→ 54272→108544→217088→434176→868352→868365→875149→ 1309325→2184474→3493799	27
3704431	1→2→4→6→12→13→25→50→63→113→ 226→452→904→1808→3616→7232→14464→28928→ 28940→57880→115760→231520→463040→463052→926092→ 1852184→3704368→3704431→	27
3922763	1→2→4→8→9→17→34→68→136→272→ 281→349→698→1396→1532→3064→6128→12256→ 24512→49024→98048→196096→392192→784384→784665→ 1569049→2353714→3922763	27

Table 5. Comparison of results to compute hard MBC

id	n	$l(n)$	BT [2]	GA [5]	AIS [4]	MS [19] (time s.) (hits)	SA [13] (hits)	GA proposed hits (time s.) (hits)
3	19	6	6	6	6	⊕6 (9.629s) (20)	6 (32)	6 (0.001s) (31)
4	29	7	7	7	7	⊕7 (7.470s) (27)	7 (32)	7 (0.002s) (31)
5	47	8	8	8	8	⊕8 (9.477s) (10)	8 (32)	8 (0.005s) (31)
6	71	9	9	9	9	⊕9 (13.408s) (8)	9 (32)	9 (0.01s) (31)
7	127	10	10	10	10	⊕10 (12.282s) (4)	⊕10 (30)	10 (0.025s) (31)
8	191	11	11	11	11	⊕11 (14.816s) (2)	⊕11 (30)	11 (0.044s) (31)
9	379	12	12	12	12	⊕12 (14.057s) (29)	⊕12 (30)	12 (0.103s) (31)
10	607	13	13	13	13	⊕13 (21.635s) (21)	⊕13 (28)	13 (0.176s) (31)
11	1087	14	14	14	14	⊕14 (26.326s) (8)	⊕14 (13)	14 (0.324s) (31)
12	1903	15	15	15	15	⊕15 (18.538s) (26)	⊕15 (6)	15 (0.550s) (31)
13	3583	16	16	17	16	-	⊕16 (1)	16 (0.973s) (30)
14	6271	17	17	17	17	⊕17 (23.032s) (9)	⊕17 (1)	17 (1.567s) (31)
15	11231	18	18	18	18	⊕18 (30.650s) (20)	⊕18 (1)	18 (2.480s) (31)
16	18287	19	19	19	19	⊕19 (28.136s) (4)	⊕19 (2)	19 (3.549s) (31)
17	34303	20	20	20	20	⊕20 (29.623s) (0)	⊕20 (1)	20 (5.566s) (26)
18	65131	21	21	21	21	⊕21 (32.396s) (3)	⊕21 (2)	21 (8.512s) (31)
19	110591	22	22	22	22	-	⊕22 (2)	22 (11.994s) (22)
20	196591	23	23	23	23	-	⊕23 (2)	23 (17.088s) (30)
21	357887	24	24	24	24	-	⊕24 (1)	24 (24.047s) (27)
22	685951	25	25	25	25	⊕25 (37.696s) (2)	⊕26 (0)	25 (34.360s) (13)
23	1176431	26	26	⊕27	26	⊕27 (46.717s) (0)	⊕26 (1)	26 (46.562s) (5)
24	2211837	27	27	⊕28	27	-	⊕27 (2)	27 (63.255s) (7)
25	4169527	28	-	⊕29	28	28 (33.291s) (1)	⊕28 (1)	28 (86.3890s) (21)
26	7624319	29	-	⊕30	29	29 (42.237s) (1)	⊕30 (0)	29 (114.9951s) (11)
27	14143037	30	-	⊕31	30	30 (64.844s) (1)	⊕30 (2)	30 (152.4645s) (8)

Table 6. Accumulated addition chain length and time obtained with the GA proposed for n in the ranges [1,512], [1,1000], [1,1024], [1,2000], [1,2048] and [1,4096]

n ∈	[1,512]	[1,1000]	[1,1024]	[1,2000]	[1,2048]	[1,4096]
AS						
Optimal	4924	10808	11115	24063	24731	54408
δ	0	0	0	3	4	24
Best	4924	10808	11115	24066	24735	54432
Average	4926.2581	10817.0645	11124.3226	24106.2581	24777.6129	54572.5791
Worst	4932	10838	11147	24185	24860	54807
Std. Dev.	1.1062	1.9333	1.8384	3.8266	4.1557	11.5808
CV	0.0002	0.0001	0.0001	0.0001	0.0001	0.0002
Time (s)						
Best	33.4870	138.9694	146.0097	567.8268	595.2965	2291.0749
Average	34.4035	141.9297	149.0738	577.4052	605.2810	2326.0918
Worst	39.0730	153.831	161.2811	602.2832	630.98	2338.8750
Std. Dev.	0.1024	0.3386	0.3412	0.8298	0.8331	2.1918
CV	0.0029	0.0023	0.0022	0.0014	0.0013	0.0009

Table 7. Comparison between classics approaches and the GA proposed for accumulated addition chains in the range [1,1000]

Strategy	Total length	Difference with l(n)
Binary [1]	11925	1117
Factor [1]	11088	280
Dichotomy [1]	11064	256
Fermat [1]	10927	119
Dyadic [1]	10837	29
Total [1]	10821	13
Proposed GA	10808	0
Optimal	10808	-

Table 8. Comparison of the best results of accumulated addition chains lengths

n ∈	optimal	AIS [4]	SA [13]	EP [6,7]	GA [5]	GA [18]	Proposed	Δ
[1,512]	4924	4924	-	4924	4925	4924	4924	0
[1,1000]	10808	10808	10823⊕	10808	-	10809⊕	10808	0
[1,1024]	11115	11120⊕	-	11115	-	-	11115	0
[1,2000]	24063	24108⊕	-	24070⊕	24124⊕	24076⊕	24066	4
[1,2048]	24731	4778⊕	-	24737⊕	-	24748⊕	24735	2
[1,4096]	54408	54617⊕	-	54487⊕	54648⊕	54487⊕	54432	55
total Δ								61

Table 9. Comparison of coefficient of variation (*CV=average/stddev*) of the of accumulated addition chains lengths

n ∈	AIS [4]	SA [13]	EP [6,7]	GA [5]	GA [18]	Proposed
[1,512]	0.0001	-	0	0.0009⊕	0	0.0002
[1,1000]	0.0002⊕	0.0002⊕	0	-	0.0001	0.0001
[1,1024]	0.0002⊕	-	0	-	-	0.0001
[1,2000]	0.0002⊕	-	0	0.0002⊕	0.0001	0.0001
[1,2048]	0.0002⊕	-	0	-	0.0001	0.0001
[1,4096]	0.0002	-	0	0.0002⊕	0.0001	0.0002

5 Conclusions

In this paper, we have presented a novel approach to find Minimum Brauer Chains based on a GA with the following features: a representation based in FNS which has proven to be well suited to the problem of addition chains [13,19]; the use of distribution functions used by the neighborhood functions to focus their behavior; and a limit of evaluation functions according to the BC searched.

The use of a representation based on the FNS allows the implementation of neighborhood functions and recombination without a repairing process, giving the advantage to construct hard MBCs in a fast way, without modifying the search space. This properties allow to preserve the characteristics of individuals inside the recombination strategy and the local search. In this sense, the use of FNS open the possibilities of using other neighborhood functions or recombination strategies, like recombination of two points or uniform crossover.

Each one of the neighborhood functions used have distinct properties: NF_1 and NF_2 can be named as explorations functions while NF_3 and NF_4 can be seen as exploitation functions. Also NF_1 and NF_3 produce less dramatic changes than NF_2 or NF_4.

The use of distribution functions allowed to focus the section of the chain where its necessary to make a change with more probability. For this reason we used a mixture of distribution functions inside the recombination strategy and neighborhood functions. In the case of the single point recombination, the position of point used could determine if it is going to be an exploration or an exploitation so the use of a distribution functions let center the main work that need to be done by the crossover.

The experimental results of the three benchmarks used demonstrated the strength of the GA proposed, in terms of the quality of the solutions. For the first benchmark consisting of 27 hard instances (that meet the requirement to be $c(r)$), we obtained the same or better results in all the instances as others competitive approaches reported in the literature, in fact all $l(n)$ was reached.

For the second experiment which consisted on 20 instances hard to optimize, we could reach 19 chains with length equal to $l(n)$.

For the third experiment which was done through calculating the accumulated addition chain length for a sequence of number in the range $[1,Z]$ for $Z \in \{512, 1000, 1024, 2000, 2048, 4096\}$, our proposed approach gives a competitive

solution against the classics approaches: binary, factor, dichotomic, fermat, dyadic and total [1].

Additionally our approach is strong versus new metaheuristic strategies, achieving better quality results than: BackTracking [2], Genetic Algorithm [5], Artificial Inmune System [4], Mutation Selection [19] and Simulated Annealing [13].

Due to overall results in the three benchmarks, we can conclude that the proposed approach is a feasible solution to get MBC, however the search of efficient strategies to get MACs and MBCs is not finished and need more research and that the practitioners of cryptosystems use MBC in a more thoroughly way.

Acknowledgments. The authors acknowledge the support of access to the infrastructure of high performance computing in the Laboratory of Information Technologies Unit from CINVESTAV-Tamaulipas.

This research was partially funded by the following projects: CONACyT 58554 - Cálculo de Covering Arrays, 51623 - Fondo Mixto CONACyT y Gobierno del Estado de Tamaulipas.

References

1. Bergeron, F., Berstel, J., Brlek, S.: Efficient computation of addition chains. Journal de Théorie des Nombres de Bordeaux 6(1), 21–38 (1994)
2. Bleichenbacher, D., Flammenkamp, A.: An efficient algorithm for computing shortest addition chains. SIAM Journal of Discrete Mathematics 10(1), 15–17 (1997)
3. Clift, N.: Calculating optimal addition chains. Computing 91, 265–284 (2011), http://dx.doi.org/10.1007/s00607-010-0118-8, doi:10.1007/ s00607-010-0118-8
4. Cruz-Cortes, N., Rodriguez-Henriquez, F., Coello Coello, C.: An artificial immune system heuristic for generating short addition chains. IEEE Transactions on Evolutionary Computation 12(1), 1–24 (2008)
5. Cruz-Cortés, N., Rodríguez-Henríquez, F., Juárez-Morales, R., Coello Coello, C.A.: Finding Optimal Addition Chains Using a Genetic Algorithm Approach. In: Hao, Y., Liu, J., Wang, Y.-P., Cheung, Y.-m., Yin, H., Jiao, L., Ma, J., Jiao, Y.-C. (eds.) CIS 2005. LNCS (LNAI), vol. 3801, pp. 208–215. Springer, Heidelberg (2005)
6. Domínguez-Isidro, S., Mezura-Montes, E., Osorio-Hernández, L.: Addition chain length minimization with evolutionary programming. In: Proceedings of the I Congreso Internacional de Electrónica, Instrumentación y Computación, Minatitlan, Veracruz, México (2011)
7. Dominguez-Isidro, S., Efren, M.M.: Addition chain length minimization with evolutionary programming. In: Proceedings of the 13th Annual Conference Companion on Genetic and Evolutionary Computation, Dublin, Ireland, pp. 59–60. ACM (2011)
8. Eiben, A., Smith, J.: Introduction to evolutionary computing. Springer (2003)
9. Flammenkamp, A.: Shortest addition chains repository (June 2012), http://wwwhomes.uni-bielefeld.de/achim/addition_chain.html
10. Gelgi, F., Onus, M.: Heuristics for Minimum Brauer Chain Problem. In: Levi, A., Savaş, E., Yenigün, H., Balcısoy, S., Saygın, Y. (eds.) ISCIS 2006. LNCS, vol. 4263, pp. 47–54. Springer, Heidelberg (2006), http://dx.doi.org/10.1007/11902140_7

11. Gordon, D.M.: A survey of fast exponentiation methods. Journal of Algorithms 27(1), 129–146 (1998), http://www.sciencedirect.com/science/article/pii/S0196677497909135
12. Guy, R.: Unsolved problems in number theory, 3rd edn., vol. 1. Springer (2004)
13. Jose-Garcia, A., Romero-Monsivais, H., Hernandez-Morales, C.G., Rodriguez-Cristerna, A., Rivera-Islas, I., Torres-Jimenez, J.: A Simulated Annealing Algorithm for the Problem of Minimal Addition Chains. In: Antunes, L., Pinto, H.S. (eds.) EPIA 2011. LNCS, vol. 7026, pp. 311–325. Springer, Heidelberg (2011), http://dx.doi.org/10.1007/978-3-642-24769-9_23
14. Nedjah, N., de Macedo Mourelle, L.: Efficient Parallel Modular Exponentiation Algorithm. In: Yakhno, T. (ed.) ADVIS 2002. LNCS, vol. 2457, pp. 405–414. Springer, Heidelberg (2002), http://dx.doi.org/10.1007/3-540-36077-8_43
15. Nedjah, N., de Macedo Mourelle, L.: Finding Minimal Addition Chains Using Ant Colony. In: Yang, Z.R., Yin, H., Everson, R.M. (eds.) IDEAL 2004. LNCS, vol. 3177, pp. 642–647. Springer, Heidelberg (2004), http://dx.doi.org/10.1007/978-3-540-28651-6_94
16. Nedjah, N., de Macedo Mourelle, L.: Towards minimal addition chains using ant colony optimisation. Journal of Mathematical Modelling and Algorithms 5, 525–543 (2006), http://dx.doi.org/10.1007/s10852-005-9024-z
17. Nedjah, N., Mourelle, L.: Efficient Pre-Processing for Large Window-based Modular Exponentiation Using Genetic Algorithms. In: Chung, P.W.H., Hinde, C.J., Ali, M. (eds.) IEA/AIE 2003. LNCS, vol. 2718, pp. 165–194. Springer, Heidelberg (2003), http://dx.doi.org/10.1007/3-540-45034-3_63
18. Osorio-Hernandez, L., Mezura-Montes, E., Cruz-Cortes, N., Rodriguez-Henriquez, F.: A genetic algorithm with repair and local search mechanisms able to find minimal length addition chains for small exponents. In: IEEE Congress on Evolutionary Computation, CEC 2009, pp. 1422–1429 (May 2009)
19. Rodriguez-Cristerna, A., Torres-Jiménez, J., Rivera-Islas, I., Hernandez-Morales, C.G., Romero-Monsivais, H., Jose-Garcia, A.: A Mutation-Selection Algorithm for the Problem of Minimum Brauer Chains. In: Batyrshin, I., Sidorov, G. (eds.) MICAI 2011, Part II. LNCS, vol. 7095, pp. 107–118. Springer, Heidelberg (2011), http://dx.doi.org/10.1007/978-3-642-25330-0_10
20. Thurber, E.: Efficient generation of minimal length addition chains. SIAM Journal on Computing 28(4), 1247–1263 (1999)

Type-2 Fuzzy Grammar in Language Evolution

Juan Paulo Alvarado-Magaña[1], Antonio Rodríguez-Díaz[1],
Juan R. Castro[1], and Oscar Castillo[2]

[1] Autonomous University of Baja California, Faculty of Chemical Sciences and Engineering, Tijuana, Baja California, Mexico
[2] Tijuana Institute of Technology, Tijuana, Baja California, Mexico

Abstract. This paper proposes a new approach to simulating language evolution, it expands on the original work done by Lee and Zadeh on Fuzzy Grammars and introduces a Type-2 Fuzzy Grammar. Ants in an Ant Colony Optimization algorithm are given the ability of embedding a message on the pheromone using a Type-2 Fuzzy Grammar. These ants are able to gradually adopt a foreign language by adjusting the grades of membership of their grammar. Results that show the effect of uncertainty in a language are given.

Keywords: ACO, Language Evolution, Type-2 Fuzzy Grammar.

1 Introduction

How humans developed language can be explain from two different points of view. Some researchers believe that human linguistic abilities are innate ([7], [22], [23], [24]), this is what is called the Nativist point of view, which sustains that language is rooted in the brain's biology, in other words, the brain has an organ dedicated to language. The second point of view is known as Non-nativist, which claims that language is a byproduct of general intellectual abilities ([9], [26], [27]), this point of view doesn't assume the existence of certain characteristics in human biology but claims that language is an emergent response to evolutionary pressure applied to human ancestors.

Having these two points of view has created the Nativist versus Non-nativist divide. One example that supports Nativist is the existence of a critical period where children can learn a language; if a child is exposed to a language before the age of six he is able to learn it fluently regardless of intellectual and environmental circumstances, while an adult requires a greater amount of e ort to learn a foreign language [21].

The origin of the Nicaragua Sign Language is another example of language nativism. The first school for deaf children in Nicaragua was opened in 1977, before then deaf people lived isolated with their immediate family and communicated

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 501–515.
springerlink.com

using signs specific to their home. It wasn't until they opened these schools that children had the opportunity to socialize with others with similar abilities. Through this interaction they took each other's dialects and formed a more comprehensive vocabulary by which all children could communicate. Re-searchers noted that younger children could conjugate verbs in ways that older children could not and it was also found that younger children were introducing new structures to the grammar as they were learning it ([17], [18], and [25]).

In [12] researchers found that in order to enhance a child's ability to learn adults adjust their language level when talking to them by using simpler grammatical forms and vocabulary, which suggest that humans are instinctively good at teaching language, which is a Non-nativist method of transmitting language.

To close the Nativist vs. Non-nativist division simulations of language evolution can be applied. As explained in [21] simulations should explore three different aspects: the Nativist vs. Non-Nativist perspectives, syntax evolution and finally the evolution of communication (usage of words as symbols).

There have been many attempts to simulate this phenomenon which have shown good results in formal languages ([2], [3], [13], [14], [15]); most common are those that use genetic algorithms ([3], [13], [15]) and neural networks ([14]), which will be further explain in later sections.

This paper explores grammar evolution and language acquisition using a Type-2 Fuzzy Grammar, which is an extension of Lee and Zadeh's original work in [19] on Fuzzy Grammars. A modified Ant Colony Optimization algorithm is used to simulate the social interactions required in a communicating society and results that support the validity of this approach are given, as well as a detailed explanation of how the simulation operates.

This paper has four sections organized as follows: section two is a collection of previous work. Section three is an explanation of the simulation itself; it includes a formal definition for Fuzzy Grammars as well as the modified ACO algorithm. Results are presented in section four and finally conclusions are shared in section five.

2 Previous Work

2.1 Emergent Vocal System in an Agent Society

This is an example of a Non-nativist simulation in which a language changes due to the constant interaction between participating agents. In [2], de Boer simulates a sound system organization using imitation games. To achieve this he proposes an experiment in which agents are added or removed from a population, each having a device that synthesizes sounds similar to the human voice and another device that receives and decodes sounds in real time.

During a game, an agent randomly generates a sound that is added to its lexicon, a second agent perceives this sound and tries to decode it and reproduce it, if

the first agent finds that the imitation is sufficiently similar it gives positive feedback to the second agent. If however the sound is very different, the second agent tries to modify it to better match the original.

Using this method de Boer shows that a system of shared sounds can emerge through adaptive imitation games.

2.2 Development of Shared SYMBOLS

Another case in which the simulation takes a Non-nativist approach is as follows. Hutchins and Hazelhurst in [14] tried to weaken the assumption that a lexicon used by agents must be provided externally by the researcher. They started with agents with no innate knowledge of a lexicon and prepared them with a finite set of situations they may experience. In order to identify them, each agent has a network that represents a situation as an activation of nodes (the network's connection weights are initialized at random).

Agents take turns performing roles as emitters and receivers. Both are presented with the same situation, and a pattern is formed within each network, the emitter then transmits its pattern to the receiver, and since both perceive the same situation the receiver can use the emitter's pattern as an additional input in order to adjust its own network.

After a certain number of iterations agents converge on a shared vocabulary, which means that when agents perceive the same situation the patterns produced by the network are equal for all agents.

2.3 The Bioprogram Hypothesis and the Baldwin Effect

Using a multi-agent system Briscoe simulated the propagation of language using genetic algorithms [3], this is an example of a simulation taking the Nativist point of view in consideration. Agents are created with a grammar capable of analyzing a sequence of categories or statements. The grammar is partly innate and contains categories with some of the ways they can be combined.

One agent generates a sequence and another tries to interpret it using its internal grammar, if a derivation tree can be generated then the interaction is successful.

The genetic part of the algorithm comes into play when agents are selected for reproduction. Each agent has a degree of fitness that depends on its success in interactions, the expressibility of the language and the amount of memory used to make derivations.

When an agent is unsuccessful in its interaction it can modify its grammar. The fittest agents are chosen for reproduction, the o spring's grammar is formed by both parents' grammar.

This method proves that through genetic assimilation one grammar will eventually dominate over an entire population.

2.4 Grammar Emergence in Communication Agents

Ikegami and Hashimoto demonstrated that a grammar can increase complexity, and therefore be more expressive, using an evolutionary method [13].

In this method, agents have an internal grammar they use to generate a string of ones and zeros that is then transmitted to other agents. Agents who receive the string try to use their internal grammar to replicate it, each agent keeps track of how many steps it needs to derive the string from its grammatical rules.

For selection, agents that interpret long chains in fewer steps randomly change their grammatical rules and are allowed to reproduce. Agents that fail to derive the string are removed from the population and since only the most successful agents are kept in the population this could be seen as another example of a Nativist simulation.

Through this process a regular grammar eventually transforms into a context-free grammar. It is known that a context-free grammar can generate more words than a regular grammar.

2.5 Evolution of Communication Agents in the Predator-Prey Pursuit Game

Jim and Giles in [15] use the Predator-Prey pursuit game as a case study. During a game, agents communicate with each other by writing a string of ones and zeros to a message board. Once all messages have been posted, each agent reads all the strings and concatenates them into a single input that is passed to a finite state machine to determine the next move.

In order to evolve the language each predator is encoded in a chromosome. The initial generation usually doesn't capture the prey, but as generations advance the lengths of the strings grow and improve agent performance.

Jim and Giles find that there must be a minimum size language to solve such problems.

2.6 Comparison with Previous Work

The proposed method is a new approach to the study of language evolution, it differs from those discussed earlier in that it doesn't use traditional crisp grammars but instead opts to use Type-2 Fuzzy Grammars. It also differs from previous experiments [1] in that Fuzzy Grammars are extended into a Type-2 Fuzzy Grammar.

As noted in [19], Fuzzy Grammars are a midway point between the precision of formal languages and the ambiguity of natural languages. They provide the flexibility to partially integrate new rules to a grammar and maintain seldom used rules without completely eliminating them. Type-2 Fuzzy Grammars also provide a level of uncertainty that in this case will model the mastery level of an individual with a new language.

3 Simulation

Dorigo's Ant Colony Optimization [10] also known as ACO, was selected as a case study. ACO is a meta-heuristic inspired by the foraging habits of ants where each ant communicates with other members of the colony by placing a chemical substance, called a pheromone, on the ground. This experiment takes advantage of this feature since successful interaction with other ants is key to finding a solution.

During the experiments each ant is able to embed a message to the pheromone using a Fuzzy Grammar, other ants need to parse it in order to follow the trace. Classic ACO algorithms use a homogeneous colony to find a solution, the algorithm is extended by including more than one group of ants who are segregated by a different Fuzzy Grammar, and this allows multiple groups of ants to work on the same problem space to find multiple solutions. During the simulation the colony will attempt to minimize De Jong's function, which has one global minimum.

The experiments show how two groups of ants find the same solution by acquiring each other's language through the use of Fuzzy Grammars.

3.1 Ant Colony Optimization

Social interaction is one of the most important aspects of ant survival, this interaction is modeled by Dorigo in the Ant Colony Optimization algorithm [10]. In ACO, individual ants leave a pheromone deposit to mark a solution in a problem space, doing so allows other ants to follow the pheromone trace and arrive at similar solutions. Pheromone intensity is either reinforced as more ants visit the same solution or it evaporates as bad solutions get discarded. Eventually the highest concentration is found around the best solutions.

The pheromone in ACO is a form of communication, by embedding a word in it ants may pass messages between themselves. Ants are divided into different groups, such that ants will only follow a pheromone trace left behind by other ants of the same group, this is done by providing each group with its own language as described by a Fuzzy Grammar.

3.2 Type-2 Fuzzy Grammar

In 1975 Zadeh introduced a concept called Type-2 Fuzzy sets [30], which is an extension that permits the inclusion of uncertainty about the membership functions of traditional fuzzy sets. Ever since then there have been many contributions that use this concept, for example in Fuzzy Logic [5][20] and machine learning [4].

Lee and Zadeh's original definition for Fuzzy Grammars [19] is extended into a Type-2 Fuzzy Grammar by implementing the concept of a Type-2 fuzzy set as follows:

Definition 1. *A Type-2 Fuzzy Grammar is a quadruple* $G = (V_N; V_T; P'; S)$ *in which* V_T *is a set of terminals,* V_N *is a set of non-terminals, P is a set of fuzzy productions and S 2 VT is the set of starting variables*

The elements of *P'* are all productions in the form

$$\mu(\alpha \rightarrow \beta) = \omega', \omega' = [\underline{\omega}, \overline{\omega}] \tag{1}$$

Where $\alpha \rightarrow \beta$ expresses a re-writing rule, and are in $(V_N\ U\ V_T)$ and ω' is the grade of membership given in an interval [0, 1] of β given α. A fuzzy production where ω = [0; 0] is assumed to not be in *P'*.

A string of terminals *x* is said to be in the fuzzy language *L(G)* if and only if *x* is derivable from the starting variable *S*. The grade of membership of *x* in *L(G)* is given by

$$\begin{aligned} \mu_G(x) &= [\underline{\mu}_G(x), \overline{\mu}_G(x)] \\ = [&supmin(\underline{\mu}(S \rightarrow \alpha_1), \underline{\mu}(\alpha_1 \rightarrow \alpha_2), ..., \underline{\mu}(\alpha_n \rightarrow x)), \\ &supmin(\overline{\mu}(S \rightarrow \alpha_1), \overline{\mu}(\alpha_1 \rightarrow \alpha_2), ..., \overline{\mu}(\alpha_n \rightarrow x))] \end{aligned} \tag{2}$$

The uncertainty of a string of terminals *x* is given by $\Delta(x)$

$$\Delta(x) = \overline{\mu}_G(x) - \underline{\mu}_G(x) \tag{3}$$

3.3 The Algorithm

The simulation has two groups of ten ants (twenty in total, at the beginning ants share the same Fuzzy Grammar with members of their group (these grammars are given in Chomsky Normal Form). Both groups will attempt to minimize De Jong's function which is as follows:

$$f(x) = \sum_{i=1}^{n} x_i^2 \tag{4}$$

In this case ants will find the point in which equation 4 evaluates to zero in a two dimensional space, so $n = 2$. In other words, ants walk across a plane such that each coordinate pair is used as input in De Jong's function. As ants explore the problem space they will evaluate De Jong's function looking for the global minimum, finding it means they have found a food source and thanks to ACO other ants will be able to follow the pheromone trace to it. It is expected that after some iterations the number of ants that find the food source will increase. It is important to note that the two dimensional plane is discretized to allow ants to take steps in controlled increments.

Each point in the problem space is a deposit that holds pheromones. Since many ants can pass over the same deposit, different pheromone levels are tracked. Before an ant can decide on a new position the dominant pheromone is determined (the pheromone with the highest intensity is the dominant one).

The simulation is divided into epochs, during each epoch ants must choose a route based on the dominant pheromone left behind during the previous epoch. The main body of the simulation is as follows.

```
1:        ant[n] is an array of n ants
2:        Each ant is assign one of two grammars
given in Chomsky Normal Form
3:        Each ant is placed randomly in the two
dimensional plane
4:        for epoch = 0 to maximum number of epochs
do
5:        for i = 0 to n do
6:        ant[i] chooses a route
7:        end for
8:        update pheromone intensity
9:        update messages in the pheromone
10:       end for
```

An ant forms a route incrementally by stepping into a new deposit and then selecting from the eight possible adjacent spaces, a route is completed once it has reached a maximum number of allowed movements. The probability of choosing deposit *i* is:

$$P(i) = \left(\frac{\tau(i)}{\sum_{j=1}^{8} \tau(j)} \right) (1 - \Delta(i)) \tag{5}$$

where $\tau(i)$ is the pheromone intensity at deposit *i* and Δ is the uncertainty of the message in *i* as given by 3. The following pseudo-code illustrates how a route is generated.

```
1:        route[n] is an array of deposits of size n
2:        best is the best solution found so far for
De Jong's function
3:        Set route[0] = best
4:        for j = 1 to n do
5:        position[8] is an array of eight possible
positions an ant can choose as its next step
6:        Set position with the message and intensity
of the dominant pheromone
7:        for i = 0 to 8 do
```

```
8:        Set route[j] = position[i] with probability
given by equation 5
9:        end for
10:       if route[j] is a better solution then best
when evaluated under De Jong then
11:       Set best = route[j]
12:       end if
13:       end for
```

In order to parse the embedded message an algorithm given by Cocke [6], Younger [26] and Kasami [14], also known as the CYK algorithm, is used. One common implementation of the CYK algorithm uses a three dimensional boolean matrix to store true or false values as the parsing tree is built. In order to calculate G(x) as given in 2 a modification that allows storage of grades of membership of each production rule is made to the CYK algorithm.

```
1:        S = a1a2...an is the message to parse of
length n
2:        G(S) is the grade of membership of S in the
fuzzy grammar
3:        The fuzzy grammar contains r non terminal
variables
4:        P[n, n, r] is a three dimensional matrix
with real values, each position stores both the lower
and upper grades of membership of a production rule
5:        for i = 0 to n do
6:          for all unit productions Rj →ai do
7:            P[0, i, j] = μ(Rj → ai)
8:          end for
9:        end for
10:       for i = 1 to n do
11:         for j = 0 to n - i do
12:           for k = 0 to i do
13:             for all productions RA → RBRC do
14:               if P[k, j, B] > 0 and P[i-k-1, j+k+1;
C] > 0 then
15:                 P[i, j, A] = min(μ(RA → RBRC), P[k,
j, B], P[i-k-1, j+k+1, C])
```

```
16:             end if
17:           end for
18:         end for
19:       end for
20:     end for
21:     G(S) = P [n    1; 0; 0]
```

Grades of membership for the fuzzy production rules have to be adjusted after the route has been selected. The grammar rules that were used are reinforced by raising their grade of membership, while those that weren't are lowered. As epochs pass the rules most often used will have a higher grade of membership while the least used will eventually cease to be part of the grammar.

```
1:      P [n] is an array of production rules of
size n
2:      μ(P[i]) is the grade of membership of pro-
duction P[i]
3:      is the degree by which μ(P[i]) is lowered
or increased
4:      set α = 0.01
5:      for i = 0 to n do
6:        if Production P[i] was used during parsing
then
7:          Set μ(P[i]) = μ(P[i]) + (μ(P[i]) * α)
8:          Set μ̄(P[i]) = μ̄(P[i]) + (μ̄(P[i]) * α)
9:        else
10:         Set μ(P[i]) = μ(P[i]) - (μ(P[i]) * α)
11:         Set μ̄(P[i]) = μ̄(P[i]) + (μ̄(P[i]) * α)
12:       end if
13:     end for
```

Each ant is equipped with a method for deducing the grammar rules corresponding to a message it could not understand. This allows the ant to follow a pheromone trace of a different group, the advantage being that if a different group is more successful in finding food then the ant will eventually integrate itself into that group by adopting their language. The method is based on the CYK algorithm and is the same as described in [1] but the grade of membership for the new rules are given in an interval.

```
1:        S = a1a2...an is the message to parse of
length n
2:        The fuzzy grammar contains r non terminal
variables
3:        P[n, n, r] is a three dimensional matrix
with real values, each position stores both the lower
and upper grades of membership of a production rule
4:        for i = 0 to n do
5:          if the unit productions Rj → ai doesn't
exist then
6:            add μ(Rj → ai) = 0.01 to the fuzzy gram-
mar
7:            Set P[0, I, j] = 0.01
8:          end if
9:        end for
10:       for i = 1 to n do
11:         for j = 0 to n i do
12:           for k = 0 to i do
13:             if There doesn't exists a production RA
→ RBRC such that P[k, j, B] > 0   and P[i-k-1; j+k+1;
C] > 0 then
14:               add μ(RA → RBRC) = 0.01 to the fuzzy
grammar
15:               add μ̄ (RA →RBRC) = 0.0075 to the fuzzy
grammar
16:               Set P[i, j, A] = [0.0075; 0.01]
17:             end if
18:           end for
19:         end for
20:       end for
```

The last step of each epoch updates the pheromone intensity in all deposits, the following equation is used:

$$\tau(i) = (1-\rho)\tau(i) + \frac{Q}{1+f(j)} \tag{6}$$

Where $\tau(i)$ is the current pheromone intensity in deposit i, ρ is the forgetting factor, the constant Q is a value in the same order as $f(j)$ and $f(j)$ is the result of evaluating De Jong's function with the best solution found so far by ant j.

```
1:        ant[n] is an array of n ants
2:        deposits[m] is an array of pheromone depo-
sits of size m
3:        for i = 0 to n do
4:          for j = 0 to m do
5:            if ant[j] visited deposit[j] then
6:              Use equation 6 to set depo-
sit[j].intensity
7:              Set deposit[j].message =
ant[i].getMessage
8:            end if
9:          end for
10:       end for
```

4 Experiments and Results

4.1 Experiment 1

The first experiment consisted of two groups of ten ants each and for each group's grammar G, $\Delta(G) = 0$. This simulation ran for fifty epochs with no evolution, the results are given in table 1. The first group minimized the function before the twentieth epoch, while the second group couldn't explore the problem space because the dominant pheromone wasn't understood. This experiment illustrates how a group of ants can quickly reach a solution if there's no uncertainty in the language and it also shows how group that can't understand the dominant language is kept away from the solution.

Table 1. Two groups of ants over 50 epochs with no language evolution and $\Delta(G) = 0$

Epoch	Group 1	Group 2
5	7.152	17.704
10	1.1	17.704
15	0.288	17.704
20	0	17.704
25	0	17.704
30	0	17.704
35	0	17.704
40	0	17.704
45	0	17.704
50	0	17.704

4.2 Experiment 2

During the second experiment each group of ants had a grammar G with $\Delta(G) = 0.3$ and language evolution was applied. As is shown in 2, due to the level of uncertainty the first group managed to minimize the function on the fortieth epoch while the second group reached a lower solution than the first experiment but still didn't reach the global minimum. This simulation shows how uncertainty slows down the overall search.

Table 2. Two groups of ants over 50 epochs with language evolution and $\Delta(G) = 0.3$

Epoch	Group 1	Group 2
5	6.716	14.416
10	3.204	8.364
15	0.9	5.428
20	0.376	4.3
25	0.188	1.436
30	0.072	1.084
35	0.004	0.852
40	0	0.296
45	0	0.084
50	0	0.008

4.3 Experiment 3

In the third experiment the level of uncertainty is reduced to $\Delta(G) = 0.2$ in order to view the impact this would have on the results. Table 3 shows how it only took group one thirty epochs to reach a solution, but the level of uncertainty is still high enough that the second group couldn't reach it also.

Table 3. Two groups of ants over 50 epochs with language evolution and $\Delta(G) = 0.2$

Epoch	Group 1	Group 2
5	8.712	14.356
10	4.228	8.836
15	2.944	5.372
20	2.456	1.98
25	1.416	1.036
30	0	0.444
35	0	0.208
40	0	0.208
45	0	0.036
50	0	0.036

4.4 Experiment 4

For the final experiment Δ(G) was reduced even further to Δ(G) = 0.1. The results in table 4 show both groups of ants reaching the same solution in fewer epochs than in previous experiments, the level of uncertainty was low enough to allow this and both groups converged on compatible languages.

Table 4. Two groups of ants over 50 epochs with language evolution and *Δ(G) = 0.1*

Epoch	Group 1	Group 2
5	8.304	9.556
10	3.908	4.104
15	0.872	2.776
20	0.096	2.436
25	0.088	0.432
30	0	0.164
35	0	0
40	0	0
45	0	0
50	0	0

5 Conclusions

By formally defining Type-2 Fuzzy Grammars the work done in [1] and [19] was expanded in order to simulate the uncertainty experienced by an individual learning a new language.

In order to test this, ACO was modified in several ways, first each ant was equipped with a Type-2 Fuzzy Grammar that allows it to parse and create messages. Second, the pheromone was modified to carry a message, thus an ant must be able to parse the message with a low uncertainty in order to follow the trace. Different grammars where used as a way to distinguish between multiple groups of ants exploring the same problem space.

The experiments show that if two groups of ants attempt to solve the same problem and neither group is able to understand the other's language, only one will reach the solution; but if both groups are able to assimilate each other's language then both will converge on the solution. Also, uncertainty plays an important role in finding a solution, experiments show that a large enough uncertainty will slow down the search.

This leads to conclude that Type-2 Fuzzy Grammars are a viable tool in language evolution research.

References

1. Alvarado-Magaña, J.P., Rodríguez-Díaz, A., Castro, J.R., Castillo, O.: Simulation of Language Evolution using Fuzzy Grammars. In: 31st Annual NAFIPS Meeting (2012)
2. De Boer, B.: Emergent Vowel Systems in a Population of Agents. In: Proceedings of ECAL 1997 (1997)
3. Briscoe, T.: Language Acquisition: The Bioprogram Hypothesis and the Baldwin Effect, MS, Computer Laboratory, University of Cambridge (1997)
4. Castro, J.R., Castillo, O., Melin, P., Rodríguez-Díaz, A.: A hybrid learning algorithm for a class of interval type-2 fuzzy neural networks. Inf. Sci. 179(13), 2175–2193 (2009)
5. Castillo, O., Melin, P., Pedrycz, W.: Design of interval type-2 fuzzy models through optimal granularity allocation. Appl. Soft Comput. 11(8), 5590–5601 (2011)
6. Chiswick, B.R., Miller, P.W.: A Model of Destination-Language Acquisition: Application to Male Immigrants in Canada. Demography 38(3) (2001)
7. Chomsky, N.: Government and Binding. Foris, Dordrech (1981)
8. Cocke, J., Schwartz, J.T.: Programming Languages and Their Compilers. Courant Institute of Mathematical Sciences, New York (1970)
9. Deacon, T.: The Symbolic Species. Penguin, London (1997)
10. Dorigo, M., Di Caro, G.: The Ant Colony Optimization Meta-Heuristic. In: New Ideas in Optimization. McGraw-Hill (1999)
11. Dustmann, C., Fabbri, F.: Language Proficiency and Labour Market Performance of Immigrants in the UK. The Economic Journal 113(489) (2003)
12. Fernald, A., Simon, T.: Expanded intonation contours in mother's speech to newborns. Developmental Psychology 20, 104–113 (1984)
13. Hashimoto, T., Ikegami, T.: Emergence of net-grammar in communicating agents. BioSystems 38, 1–14 (1998)
14. Hutchins, E., Hazelhurst, B.: How to invent a lexicon. The development of shared symbols in interaction. Artificial societies: The computer simulation of social life. UCL Press, London
15. Jim, K.-C., Giles, C.L.: Talking Helps: Evolving Communicating Agents for the Predator-Prey Pursuit Problem. Artificial Life 6(3), 237–254 (2000)
16. Kasami, T.: An efficient recognition and syntax analysis algorithm for context-free languages., Technical Report AFCRL-65-758, Air Force Cambridge Research Laboratory, Bedford, Massachusetts (1965)
17. Kegl, J.: Conference Report: Linguistic Society of America Meeting. Signpost 7(1) (1994)
18. Kegl, J.: The Nicaraguan Sign Language Project: An Overview. Signpost 7(1) (1994)
19. Lee, E.T., Zadeh, L.A.: Note on fuzzy languages. Information Sciences 1, 421–434 (1969)
20. Mendel, J.M.: Uncertain Rule-Based Fuzzy Logic Systems: Introduction and New Directions. Prentice-Hall, Upper Saddle River (2001)
21. Perfors, A.: Simulated Evolution of Language. Journal of Artificial Societies and Social Simulation 5(2) (2000)
22. Pinker, S., Bloom, P.: Natural language and natural selection. Behavioural and Brain Sciences 13, 707–784 (1990)
23. Pinker, S.: The Language Instinct. William Morrow and Company (1994)

24. Pinker, S.: “Why the child holded the baby rabbits: A case study in language acquisition”, Language: An invitation to Cognitive Science, 2nd edn., pp. 107–133. MIT Press, Cambridge (1995)
25. Senghas, A.: The Development of Nicaraguan Sign Language via the Language Acquisition Process. In: BUCLD 19: Proceedings of the 19th Annual Boston University Conference on Language Development. Cascadilla Press, Boston (1995)
26. Shipley, E., Kuhn, I.: A constraint on comparisons: equally detailed alternatives. Journal of Experimental Child Psychology 35, 195–222 (1983)
27. Tomasello, M.: First words: A case study of early grammatical development. Cambridge University Press, Cambridge (1992)
28. Veglery, A.: Differential Social Integration among First Generation Greeks in New York. International Migration Review 22(4) (1988)
29. Younger, D.H.: Recognition and parsing of context-free languages in time n3. Information and Control 10(2), 372–375 (1967)
30. Zadeh, L.A.: The Concept of a Linguistic Variable and Its Application to Approximate Reasoning. Information Sciences 8, 199–249 (1975)

Experimental Study of a New Algorithm-Design-Framework Based on Cellular Computing

J. David Terán-Villanueva[1], Héctor Joaquín Fraire Huacuja[2], Juan Martín Carpio Valadez[1], Rodolfo A. Pazos R.[2], Héctor José Puga Soberanes[1], and José Antonio Martínez Flores[2]

[1] Instituto Tecnológico de León (ITL), Avenida Tecnológico s/No, C.P. 37290, León, Gto. Mexico
david_teran01@yahoo.com.mx, jmcarpio61@hotmail.com, pugahector@yahoo.com

[2] Instituto Tecnológico de Ciudad Madero (ITCM), Av. 1o. de Mayo s/No esq. Sor Juana Inés de la Cruz, C.P. 89440, Cd. Madero, Tam. Mexico
{automatas2002,r_pazos_r}@yahoo.com.mx, jose.mtz@itcm.edu.mx

Abstract. In this paper the linear ordering problem with cumulative costs (LOPCC) is approached. Currently, a tabu search and a GRASP with evolutionary path-relinking have been proposed to solve it. We propose a new pseudo-parallel strategy based on cellular computing that can be applied to the design of heuristic algorithms. In this paper the proposed strategy was applied on a scatter search algorithm to show the feasibility of our approach. A series of computational experiments of the designed cellular algorithm were carried out to analyze the performance reached with the proposed approach versus its monolithic counterpart. Additionally, several parameters used were analyzed to determine their effect on the performance of the designed algorithms. We consider that the new pseudo-parallel approach used in this work can be applied to design high performance heuristic algorithms.

1 Introduction

The linear ordering problem with cumulative costs is originated in wireless communications. In this context, the wireless devices have to communicate to a base station in order to be identified. To this end,, the Universal Mobile Telecommunication Standard (UMTS) adopted the code division multiple access technique where each device has a specific code. However, due to the simultaneous communication and radio propagation, a distortion is induced on the all the wireless devices. Then, each device produces distortion on the rest of the devices in different proportions. And so the need arises to keep the distortion as low as possible, while ensuring reception for each wireless device. There is a technique designed to keep

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 517–532.
springerlink.com

a low level distortion called successive interference cancellation (SIC). This technique detects one device at a time and then its interference is removed, so the rest of the devices would have less interference [2].

Then the problem arises of finding the order of detection for the devices that produces the minimal total interference, while keeping a desirable level of reception for each device. This problem is addressed as a joint power-control and receiver optimization (JOPCO), which is equivalent to the NP-hard linear ordering problem with cumulative costs (LOPCC) [1][2]. Formally LOPCC is defined as:

Given a complete digraph $G = (V, A)$ with no negative arc costs $c_{\pi_i \pi_j}$ and nodes with no negative costs d_{π_i} a, the problem is to find a permutation $\pi = (1, 2, \cdots, n-1, n)$ that minimizes:

$$LOPCC(\pi) = \sum_{i=1}^{n} \alpha_{\pi_i}$$

where

$$\alpha_{\pi i} = d_{\pi_i} + \sum_{j=i+1}^{n} \alpha_{\pi_i} + c_{\pi_i \pi_j} \text{ for } i = n, n-1, n-2, \cdots, 1$$

With respect to the wireless devices application, $c_{\pi_i \pi_j}$ represents the interference of device π_j on device π_i, and α_{π_i} represents the power of the signal emitted by device π_j.

In this work we propose a scatter search algorithm structured with a new cellular approach to solve LOPCC. This approach consists of the pseudo-parallel execution of a set of heuristic searches in the solution space. Each heuristic search evolves its own set of solutions, using the typical scatter search methods, and its execution is stopped once a stagnation condition is reached. Once all the heuristic searches have reached a stagnation state, an information interchange process is carried out and the pseudo-parallel execution of the heuristic searches continues. The main idea in the cellular processing approach is to split a sequential algorithm into several pseudo-parallel processing modules. The pseudo-parallel execution permits exploring different regions of the solutions space. Also the consistent verification of the stagnation conditions avoids wasting time in unnecessary tasks.

This paper is organized as follows. In Section 2 a detailed review of the related work is presented. The cellular processing approach and the proposed algorithm are described in Section 3. Finally, Sections 4 and 5 contain the experimental results and the conclusions of this work respectively.

2 Related Work

2.1 Linear Ordering Problem with Cumulative Costs

Bertacco in [2] presents for the first time the LOPCC problem and its context, the proof of its hardness and proposes two exact algorithms: a mixed-integer linear programming algorithm and an ad-hoc enumerative algorithm which uses branch

and bound techniques. This algorithm uses a queue to produce solutions (permutations) and thus uses less memory and time to produce the tree of solutions. An interesting aspect of that algorithm is that it does not wait to have a complete solution to evaluate it, instead it begins the evaluation at any part of the construction and, if the evaluation of the solution exceeds the best solution known so far, then it prunes the current branch and continues with the next.

Righini [10] proposes a modification to Bertacco's enumerative algorithm using a new lower bound parameter to reduce the search space. This new parameter yields a better efficiency than the original algorithm.

In [3] a tabu search algorithm (TS) is proposed. This algorithm includes four main phases: an initial construction, a local search that uses an indicator to detect critic elements to be reordered as long as they are not tabu elements, a local search that descends to a local optimum, and another local search process that allows worsening the current solution in order to add diversification to the search process. In [11] it is shown that the initial construction is the most relevant process in the tabu search algorithm proposed in [3]. The main drawback of this approach is that the resource consumption significantly grows with the size of the instances.

In [4] a GRASP algorithm hybridized with an evolutionary path-relinking is proposed. Evolutionary path-relinking is used to produce solutions that can be included in an elite-set (ES), and the algorithm keeps working as long as new solutions are included in ES. The standard experiment used to compare the performance of the solution algorithms to LOPCC is modified in this work. Currently, this algorithm is the state of the art to solve LOPCC.

In this paper a new scatter search algorithm structured with a new cellular approach is proposed to solve the linear ordering problem with cumulative costs (LOPCC). Also we show the results of a series of experiments, of three important variables that might improve the performance of our proposed algorithm.

3 Cellular Processing Scatter Search Algorithm

3.1 Scatter Search

The scatter search heuristic was first proposed by Fred Glover in [5]. This is a procedure that tries to evolve and improve a reference set through several methods and uses a small amount of randomness in order to find global optimal solutions.

Our motivation to use a scatter search algorithm is that, it is a poblational algorithm that has several customizable functions. Also its reference set *RefSet* is constituted of quality and diversity solutions, this property permits having better control over the intensification-diversification balance. The scatter search algorithm uses a reference set, which can be constructed using quality and diverse elements.

Also, instead of using a genetic algorithm, as an alternative we choose the scatter search approach, because of its multiple configurable methods, which permits fully customizing the algorithm.

The methods used in the scatter search algorithm implementation are described in the following sections.

3.1.1 Diversification Generation Method

A reactive greedy construction was implemented for the *diversification generation method* [9][7]. The general structure is the same as that of the basic greedy construction, but when producing the restricted candidate list (RCL) $RCL = Cost_{min} + \beta(Cost_{max} - Cost_{min}$, multiple values of β are used, where $0 \leq \beta \leq 1$. These values are selected according to statistical information of the performance of previous solutions obtained with each value. The general structure of the greedy construction is shown in Algorithm Algorithm 1.

Algorithm 1: Reactive greedy construction

```
          Output π
1         cost = 0
          insertElement(Random, π)
3         cost += calculateCostOfInsertion()
          While {notFinished(π)}
5           β = selectBeta()
            RCL = createCandidateList(β)
7           nextElement = selectRandomElement(RCL)
            insertElement(nextElement, π)
9           cost += calculateCostOfInsertion()
            cost += tryToImproveSolutionSoFar()
11          updateBeta()
            EndWhile
```

The selection and update of the β values are performed in lines 5 and 11 respectively; further detail can be found in [9]. The *insertElement* function in line 8, inserts the *nextElement* in the next position to be included in permutation π, it is important to remark that the permutation π is constructed in reverse order due to the cumulative costs. While the *tryToImproveSolutionSoFar()* is a partial local search that tries to find a better partial solution by trying to insert the new element in other position.

3.1.2 Improvement Methods

For the improvement methods in our cellular processing scatter search, three local searches were used: a percentage-of-critical-elements local search (PCLS), a stagnation local search (SLS) and an optimal local search (LOS). These local search algorithms work together to form two different configurations of composite local search, each one of them with a particular purpose [12].

The percentage-of-critical-elements local search (PCLS) builds a set C of critical elements of the current permutation ordered according to the α values considered to determine their objective values. The most critical element $c \in C$ is selected, inserted in a position that improves the objective value of the permutation and removed from C. This process continues until a certain percentage of critical elements are removed from C, which we set at 30%.

The stagnation local search (SLS) selects a random element of the current permutation in order to be inserted in a position that improves the objective value of the permutation. This process is carried out until a given number of iterations without improvement are reached. We define 10\% of the instance size as the limit of iterations without improvement.

The local optimal search (LOS) builds a set C of critical elements of the current permutation, based on the α values considered to determine their objective values. The most critical element $c \in C$ is selected, inserted in a position that improves the objective value of the permutation and removed from C. Once C is empty, it is rebuilt if any improvement was achieved during the search process, otherwise the local optimal search ends.

Improvement$_1$

The *Improvement$_1$* method is carried out after the *diversification generation* method and is applied to all the solutions in the pool of solutions. This improvement method is formed by a composite local search of PCLS and SLS. The purpose of this improvement method is to apply a fast and diverse improvement (PCLS and SLS respectively) in order to obtain diverse good solutions for the pool of solutions.

Improvement$_2$

The Improvement$_2$ is carried out after the *Solution Combination* method and the *interaction* method, and it is applied to all the solutions that result from both methods. *Improvement$_2$* is formed by a composite local search: a composition of PCLS, SLS and LOS. The purpose of this improvement method is to apply a fast, diverse and local optimal search (PCLS, SLS and LOS respectively) in order to obtain a variety of local optimal solutions.

3.1.3 RefSet Update$_1$ Method

It initially builds a set Q with the best quality solutions in the *pool.*A local optimal search (LOS) is applied to these solutions when they are incorporated to the reference set. This local search was not carried out during the creation of the *pool* of solutions in order to produce diverse solutions. Once the elements in Q are selected, *RefSet Update* continues building a set D of diverse solutions, including the elements of the *pool* with the highest average distance to set Q. The distance between permutations $q \in Q$ and $d \in D$ is defined as follows:

$$\sum_{i=1}^{n} ABS(Pos(i,q) - Pos(i,d))$$

where *Pos(i,p)* is the position of element *i* in permutation *p*.

3.1.4 Stagnation Condition

This condition detects if $Q \subset RefSet$ has been modified with new solutions, so if no new solution has been added to Q, then we consider $RefSet$ to be stagnated.

3.1.5 Sub-set Generation Method

This method chooses for combination all the pairs of solutions in Q plus all the pairs (q, d) such that $q \in Q$ and $d \in D$.

3.1.6 Combination Method

In order to describe how the permutations are combined, consider two permutations π_1 and π_2, an index Ind_r for the position of each element in π_r, and the set of differences $D = \{ d_k \mid d_k = \alpha_{Ind_{2_k}} - \alpha_{Ind_{1_k}} \text{ for } k = 1, \cdots, n\}$. Now the element $k^* = arg_{min}(D)$ is selected, meaning that $\alpha_{Ind_{2_{k^*}}}$ and $\alpha_{Ind_{1_{k^*}}}$ have the largest difference. Therefore, the best improvement will be reached if element k^* is placed in position $Ind_{2_{k^*}}$. A new solution π is built inserting the element placed in position $Ind_{1_{k^*}}$, of permutation π_1, in position $Ind_{2_{k^*}}$ and d_{k^*} is eliminated from \$D\$. This process continues until we have taken 15% from D and subsequently an *Improvement*$_2$ method is carried out.

3.1.7 RefSet Update$_2$ Method

It includes new solutions in Q if they have better quality than the worst solution in Q, if so then the solutions to be replaced in Q might be included in D. Also if the new solution is not better than the worst solution in , then it might be included into D if it has a higher average distance to set Q.

3.2 Cellular Processing Approach

Our cellular processing approach was motivated due to the excessive time consumption when a sequential scatter search algorithm is used to solve large scale instances of LOPCC. Therefore, our proposal tries to execute multiple scatter search procedures with small time consumption by reducing the size of each reference set. Also, by dividing the scatter search procedure in smaller processing cells, a greater diversity is achieved due to the pseudo-parallel exploration through the search space.

The cellular processing approach consists of simulating the parallel execution of a set of processing cells (PCells). Each PCell consists of a particular heuristic search of the solution space. In each iteration of the parallel simulation process, one step of all the PCells is carried out. The PCells can be implemented with the same or different heuristics. Each PCell evolves different sets of solutions, which

must be initialized before the execution of the PCells starts. The number of the execution steps and the size of the solution set to evolve; are two parameters that have a large impact on the cellular processing algorithm performance. In this work the PCells are built using the scatter search heuristic, and each one evolves its own reference set. The framework of the cellular processing approach is shown in Algorithm 2.

Algorithm 2: Cellular processing approach

```
InitializePCells()
Repeat
  Repeat
    For {i = 1 to numberOfPCells}
      If {IsNotStagnated(PCell_i)}
        PCell_i()
      EndIf
    EndFor
  Until {AllPCellAreStagnated()}
  PCellInteraction()
  Re-structureCells()
Until {StopCondition()}
```

Algorithms 3, 4, 5 and 6 show the detailed description of the main methods used: processing cell initializer, processing cell and cellular interaction. In the interaction process, once π^{**} is produced, if $Cost(\pi^{**}) < Min(Cost(\pi_1), Cost(\pi_2))$, then π^{**} will replace the worst solution in $\{\pi_1, \pi_2\}$.

Algorithm 3: PCell initializer

```
Output Pool
For {i=0 to poolSize}
  newSol = DiversificationGeneration()
  Pool += Improvement_1(newSol)
End for
```

Algorithm 4: PCell processing

```
Input i
PCellSol_i=SubSetGeneration(i)
NPCellS_i=SolCombination(PCellSol_i)
ImprovedPCellSol_i=Improvement_2(NPCellS_i)
RefSetUpdate(ImprovedPCellSol_i,i)
```

Algorithm 5: PCell interaction

```
Repeat
  For {i=0 to numberOfPCells}
    For {j=0 to numberOfPCells}
      If {i ≤ j}
        NewSol = Interaction(PCell_i,PCell_j)
        UpdateWorstCell(i,j,NewSol)
      EndIf
    EndFor
  EndFor
Until {NoBetterSolFound()}
```

Algorithm 6: Interaction

```
Input {π¹, π²}
Output π**
Let: Ind(π,k) be a function that returns
the position of element k in solution π
D = ∅
For {k = 1 \to n}
  If {Ind(π¹,k) ≠ Ind(π²,k)}
    d_k=α²_{π²Ind(π²,k)} - α¹_{π¹Ind(π¹,k)}
    D = D ∪ d_k
  EndIf
EndFor
For {i = 1 \to n*γ}  where 0 < γ ≤ 1
  d_{k*} = arg_min(D)
  π*= InsertMove(π¹,Ind(π¹,k*),Ind(π²,k*)
  D = D \ {d_{k*}}
EndFor
π** = Improvement_2(π*)
```

The cellular processing approach is used to improve the algorithmic performance of the heuristics integrated in the PCells. The search through the solution space is done by multiple PCells, which permits searching different regions of the solution space. On each iteration of the cellular processing algorithm, each PCell carries out one scatter search iteration at a time, while it is not stagnated.

The PCells are executed in each iteration on a fixed order from 1 to n, until every PCell stagnates. The individual stagnation technique has the advantage of low time consumption, because, once k PCells have stagnated, the rest of the process continues without wasting time on those k PCells.

Once all the PCells have reached a stagnation condition, an interaction process is carried out. The interaction between PCells is carried out by updating the reference set of each cell with new solutions. These new solutions are generated through the combination process among the best elements of each PCell. At the end of this interaction and if the stop condition is not reached, then a restructure of the reference set for each PCell is carried out. This *Re-structure* method is used to produce a new reference set of solutions for each processing cell. However, the new solutions are produced with the *Solution Combination* method of the scatter search, but in this case all the new solutions substitute the old ones except for the best solution of each processing cell.

The general structure of the cellular processing scatter search algorithm is shown in Figure 1.

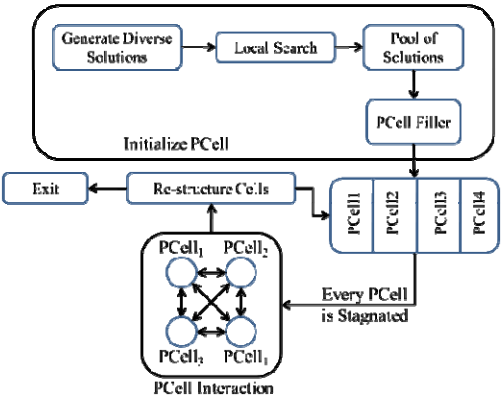


Fig. 1. Structure of the cellular processing scatter search

4 Experimental Results

In this section the experimental settings, the preliminary experimentation to configure the proposed algorithm and the performance evaluation of the cellular processing algorithm are presented.

4.1 Experimentation Settings

The algorithms were implemented in C, and two different sets of experimentations were carried out. One set of experiments was designed to study the impact of the different variables of the cellular processing proposal. The other set was designed for comparing the performance of the cellular processing scatter search (CPSS) algorithm with respect to a hypothetical algorithm that obtains all the best-known solutions.

For the first set of experimentations a computer with a Phenom X4 955 3.2GHz processor with 2GB of RAM was used. While for the comparative experiment versus the state-of-the-art algorithm, a computer with dual Xeon processors at 3.06 GHz and with 4 GB of RAM was used.

The instances used for the experimentations are:

- **UMTS**. These are instances from the group of telecommunications of the engineering school from the University of Padua, related with the order of detection for UMTS networks [2]. These instances consist of four sets of size 16, the characteristics for each set of instances are: synchronousand asynchronous, with and without scramble; and their optimal values are known.
- **Random**. Instances generated randomly with a uniform distribution proposed by Reinelt [8]. There are three sets of random instances, each one of size 35, 100 and 150 respectively; their optimal values are unknown.
- **LOLIB**. These instances come from input-output tables from the European economy [6] and are a well-known set of LOP instances. We use 48 instances of sizes 44 to 60, 30 of size 44, 4 of size 50, 11 of size 56 and 3 of size 60; their optimal values are unknown.

4.2 Study of the Impact of Different Variables for the Cellular Processing Proposal

A preliminary experimentation was conducted in order to identify the impact of a set of variables of the cellular processing proposal. In this preliminary experimentation a set of 43 LOLIB instances. The variables analyzed in these experiments were: *(a) the number of processing cells*, *(b) the processing cell size* (the running time of each processing cell) and *(c) the stagnation detection for the processing cells*.

The *number of processing cells (a)* test, tries to identify the impact of the number of processing cells in the proposed algorithm. This test is carried out in order to find out if there is any improvement attributable to the pseudo-parallel characteristic of our algorithm.

The *processing cell size (b)* test is designed to determine the impact of the processing cell size. The processing cell size is related with the number of combinations of Table 1. For example, a number of combinations equal to 60 (experiments from 1 to 5), means that in one processing cell execution, 60 combinations are produced, and therefore, this is the size of the processing cell.

The *stagnation detection for the processing cells (c)* test considers two main factors: the percentage of improvement and the amount of new solutions produced. The percentage of improvement tries to identify stagnation if no better solution has been produced with at least a certain percentage of improvement. And the amount of new solutions produced aims to detect stagnation when a certain amount of new better solutions has not been produced.

Table 1 shows the configurations used for the three variable studies in the preliminary experimentation. The first column indicates the number of experiment,

the second column shows the amount of processing cells used for the experiment, while the third and forth columns show the quantity of quality and diversity solutions in each processing cell, the fifth column contains the number of total combinations ($PCells * (Q * (Q - 1) + Q * D)$), the sixth one indicates the total number of solutions used ($PCells * Q + D$) and the seventh column shows the stagnation detection criterion used by each processing cell.

For the stagnation detection column, 1Q and 2Q mean finding at least one or two solution respectively, that replace a quality element in the processing cell. The values 1%, 2% and 3% Improvement means finding a solution that is at least 1%, 2% and 3% better than any quality solution. If these criteria are met then we state that the processing cell is not stagnated.

Table 1. Individual experiments for experiments (a), (b) and (c)

Experiment No.	Pcells	Q	D	Total of Combinations	Total of Solutions	Stagnation Detection
1	1	6	5	60	11	1Q
2	2	5	2	60	14	1Q
3	3	4	2	60	18	1Q
4	4	3	3	60	24	1Q
5	5	3	2	60	25	1Q
6	8	3	3	120	48	1Q
7	10	3	2	120	50	1Q
8	10	3	2	90	50	1Q
9	10	4	2	200	60	1Q
10	10	4	2	140	60	1Q
11	10	4	3	240	70	1Q
12	10	4	3	180	70	1Q
13	10	4	2	200	50	0.1% Improvement
14	10	4	2	200	50	0.01% Improvement
15	10	4	2	200	60	0.001% Improvement
16	10	4	2	200	60	2Q

a) Experiments from 1 to 7, b) Experiments from 7 to 12, c) Experiment 9 and from 13 to 16.

Table 2 shows the results obtained for the experiments 1 to 7 of Table 1, where we study the impact of *the number of processing cells (a)*. The first column shows the experiment number, the second one shows the average percentage error with

respect to the best-known solutions reported in [4], the third column shows the number of best-known solutions found, and the last one shows the average time for instance solution in CPU seconds.

In this table we can see that experiments 5, 6 and 7 are the ones with the best average error. Experiment 7 has the largest number of best-known solutions found (35). And the experiments with the best times are 3 and 5. However, it is important to remember that experiments from 1 to 5 are experiments with 60 combinations, and in this group experiment 1 (one processing cell) obtains 34 best-known solutions, while experiment 5 finds 32 best-known solutions in almost half the time used in experiment 1. Furthermore, experiment 7 obtains a better average error, 35 best-known solutions and uses less time than experiment 1, even though it produces 120 combinations in comparison with the 60 combinations of experiment 1. So in this experiment we can see the improved results from using a cellular processing approach instead of a monolithic approach.

Table 2. Results for the number of processing cells; (a) experiment

Experiment No.	Avg. Err.	# of Best-Known Solutions	CPU. Sec.
1	0.656	23	29.17
2	0.656	21	15.451
3	1.1	21	14.743
4	0.654	22	15.413
5	0.633	22	14.921
6	0.632	23	25.034
7	0.634	23	23.172

Table 3 shows the results obtained for experiments 7 to 12 of Table 1, where we study the impact of *the processing cell size (b)*. The first column shows the experiment number, the second one shows the average percentage error with respect to the best-known solutions reported in [4], the third column shows the number of best-known solutions found, and the last one shows the average time for instance solution in CPU seconds.

Here we can see that there is an average reduction of time of 1.5 seconds comparing experiment 7 vs 8, 9 vs 10 and 11 vs 12. The experiments are compared in that way because each pair has the same number of solutions and a different number of combinations. Also the best average error is found with experiment 11, which is the experiment with the largest number of combinations (240). However experiment 12 is the one that finds a larger number of best-known solutions (36). However, this behavior seems to be a fluke, because if we compare all the previous pairs of experiments (7,8) (9,10), they tend to reduce the number of best-known solutions found, from 35 to 31. According to these results there is a small time saving with respect to the 4 lost best-known solutions for two of the three comparisons.

Table 3. Results for the processing cell size; (b) experiment

Experiment No.	Avg. Err.	# of Best-Known Solutions	CPU. Sec.
7	0.634	23	23.172
8	0.638	21	22.428
9	0.632	23	25.262
10	0.657	22	23.553
11	0.043	23	28.434
12	0.632	24	26.276

Table 4 shows the results obtained for the experiment 9 and 13 to 16 of Table 1, where we study the impact of *the stagnation detection for the processing cells (c).* The experiments selected for this study contain the same amount of quality and diversity solutions as well as the number of combinations, the only variable is their stagnation detection. Here we can see that there is not really a tendency for experiments 13, 14 and 15, where we expected there should be an increasing average error, a decreasing number of best-known solutions and a decreasing CPU seconds. But the only tendency observed is the decreasing number of best-known solutions. However, all of them use about 10 CPU seconds less than experiment 9, which is the base experiment. For experiment 16 there is a small reduction in time and the number of best-known solutions is reduced by 2 in comparison with experiment number 9.

According to these results, if we use the configuration of experiment 13, we can decrease the CPU seconds in about 12 seconds and expect a low impact on the average error and in the number of best-known solutions found. So this is a good alternative in order to reduce the time consumed by this algorithm. However, the percentage of improvement for detecting stagnation has to be tuned for a specific set of instances, due to the range of the objective function values.

Table 4. Results for the stagnation detection for the processing cells; (c) experiment

Experiment No.	Avg. Err.	# of Best-Known Solutions	CPU. Sec.
13	0.632	23	25.272
14	0.632	23	25.294
15	0.632	23	25.279
16	0.634	22	24.522

4.3 Cellular Processing Scatter Search Performance

Table 5 shows the comparative average performance of the cellular processing scatter search (CPSS) algorithm versus a hypothetical algorithm (HA) capable of obtaining all the best-known solutions reported in [4]. This table contains the average percentage error, the number of best-known solutions found, the number of new best-known solutions produced, and the average time for instance solution in CPU seconds. The time comparison was performed against the EvPR algorithm, which is the best performance algorithm reported in [4]. We make our comparisons versus the HA algorithm, because the results in [4] do not include the objective values obtained with EvPR for each instance.

Here we can see that for the UMTS instances our algorithm was able to find 100 optimal solutions out of 100. Also the time spent on these instances was almost halved for the CPSS algorithm.

For the LOLIB instances the CPSS algorithm has a lower average error than the hypothetical algorithm and finds 43 best-known solutions. Also 16 of them are new best-known solutions. However, it uses more than one second than EvPR.

Table 5. Comparative average performance of CPSS versus HA algorithms

Algorithm	Avg. Err.	# Best-Known or Optimal Solutions	New Best-Known Solutions	CPU Sec.
UMTS				
CPSS	0	100	-	0.863
HA	0	100	-	1.62 (EvPR)
LOLIB				
CPSS	0.01	43	16	33.41
HA	382057.38	33	-	32.34 (EvPR)
Rnd 35				
CPSS	0.37	22	0	4.66
HA	0	25	-	3.75 (EvPR)
Rnd 100				
CPSS	1.78	10	10	431.95
HA	1.49	15	-	351.38 (EvPR)
Rnd 150				
CPSS	4.75	11	11	1628.75
HA	3.26	14	-	1127.24 (EvPR)

Table 6. Wilcoxon test to compare the performance of the CPSS algorithm versus the HA algorithm

Instances	Algorithm	Sum of ranks	n	Reference Value (Significance 10%)
LOLIB	CPSS	99	17	101
	HA	37		
Rnd 35	CPSS	0	3	–
	HA	6		
Rnd 100	CPSS	132	25	225
	HA	193		
Rnd 150	CPSS	121	25	225
	HA	204		

The Random instances of size 35 is the only scenario where CPSS is not able to find new best-known solutions and it only finds 22 best-known solutions.

For the Random instances of size 100 and 150, CPSS finds 10 and 11 new best-known solutions respectively. But in these scenarios our proposed algorithm uses more time than EvPR, 80 for the Random 100 instances and 501 for the Random 150 instances.

Table 6 shows the Wilcoxon tests results obtained when the CPSS algorithm and the hypothetical algorithm (HA) are compared. Here we can see that all the tests are statistically equivalent, because in all the cases the sum of ranks is never the same or greater than the reference value. This tendency indicates the algorithms are statistically equivalent.

5 Conclusion and Future Work

In this paper the problem of minimizing power consumption on wireless communications systems was approached. It is equivalent to the NP-hard linear ordering problem with cumulative costs, whose formulation is used in this work. In the literature several heuristics are proposed to solve it. In this paper a cellular processing scatter search algorithm for the linear ordering problem with cumulative costs is introduced. The cellular processing approach consists of simulating the parallel execution of a set of processing cells (PCells). In each iteration of the parallel simulation process, one step of all the PCells is carried out. Each PCell is built using a specific heuristic strategy.

A series of experiments were carried out in order to identify the impact of: the number of processing cells, the processing cell size and a stagnation detection criterion for the processing cells. These experiments show that the cellular processing approach obtains better results than the non cellular processing

implementation, a decreasing performance associated with a small-sized processing cell, and a relative good time saving if a percentage of improvement is used as stagnation criteria for the processing cells.

Also it is shown that the proposed algorithm is a good alternative to solve LOPCC instances, because it finds 37 new best-known solutions and the Wilcoxon test shows that CPSS is statistically equivalent to the hypothetical state-of-the-art algorithm. We are currently working on the implementation of a heterogeneous processing cell algorithm.

References

1. Benvenuto, N., Carnevale, G., Tomasin, S.: Optimum power control and ordering in sic receivers for uplink cdma systems. In: IEEE-ICC 2005 (2005)
2. Bertacco, L., Brunetta, L., Fischetti, M.: The linear ordering problem with cumulative costs. Eur. J. Oper. Res. 189(3), 1345–1357 (2008)
3. Duarte, A., Laguna, M., Marti, R.: Tabu search for the linear ordering problem with cumulative costs. Computational Optimization and Applications 48, 697–715 (2011)
4. Duarte, A., Marti, R., Alvarez, A., Angel Bello, F.: Metaheuristics for the linear ordering problem with cumulative costs. European Journal of Operational Research 216(2), 270–277 (2012)
5. Glover, F.: Heuristics for integer programming using surrogate constraints. Decis. Sci. 8, 156–166 (1977)
6. Laguna, M., Marti, R., Campos, V.: Intensification and diversification with elite tabu search solutions for the linear ordering problem. Computers & Operations Research 26(12), 1217–1230 (1999), doi:10.1016/s0305-0548(98)00104-x
7. Prais, M., Ribeiro, C.: Parameter variation in grasp procedures. Investigacion Operativa 9, 1–20 (2000)
8. Reinelt, G.: The linear ordering problem: Algorithms and applications. Mathematical Social Sciences 14(2), 199–200 (1985)
9. Resende, M., Riberio, C.: Greedy Randomized Adaptive Search Procedures: Advances, Hybridizations, and Applications. In: Handbook of Metaheurictics, vol. 146, pp. 283–319. Springer (2010)
10. Righini, G.: A branch-and-bound algorithm for the linear ordering problem with cumulative costs. European Journal of Operational Research 186, 965–971 (2008)
11. Teran-Villanueva, J., Fraire-Huacuja, H., Duarte, A., Pazos-Rangel, R., Carpio Valadez, J.M., Puga-Soberanes, H.J.: Improving Iterated Local Search Solution for the Linear Ordering Problem with Cumulative Costs (LOPCC). In: Setchi, R., Jordanov, I., Howlett, R.J., Jain, L.C. (eds.) KES 2010, Part II. LNCS, vol. 6277, Springer, Heidelberg (2010)
12. Teran-Villanueva, J., Pazos-Rangel, R., Martinez, J.A., Lopez-Loces, M.C., Zamarron-Escobar, D., Pineda, A.: Hybrid grasp with composite local search and path-relinking for the linear ordering problem with cumulative costs. International Journal of Combinatorial Optimization Problems and Informatics 3(1), 21–30 (2012)

Restaurant Recommendations Based on a Domain Model and Fuzzy Rules

Xochilt Ramírez-García and Mario García-Valdez

Tijuana Institute of Technology, Tijuana Mexico
xochilt.ramirez@gmail.com, mario@tectijuana.edu.mx

Abstract. This research proposes a hybrid recommender system for restaurants that uses fuzzy inference systems together with collaborative filtering and content-based techniques, considering the expert's experience, the ratings given by similar users and restaurant model. Content-based technique seeks to alleviate the cold-start problem, which commonly arises in collaborative filtering. The goal is to help each user to find interesting restaurants in the city. To evaluate the recommender system a data set of 50 users and 60 restaurants was tested. Was used RMSE for obtain the accuracy in the recommendations.

Keywords: Content-Based, collaborative filtering, fuzzy logic, recommender systems.

1 Introduction

Before the Internet consumers had limited access to information related to products and services. Traditionally, a way of recommending any product between consumers was from person to person. The Web has facilitated the spread of this information to users over the past years but this information has become so abundant that far from helping users has become a problem, huge amounts of information make overwhelmed users.

Internet brought a new way to sell products through e-commerce. One of the main objectives of e-commerce is to meet the needs and expectations of its customers, but in larger databases the search capability of users is reduced.

This need give rise to the emergence of information technologies, such as recommender systems that seek to reduce this problem through the information filtering. Companies had adopted this technology to bring more personalized interaction and to increase the customer's satisfaction.

Burke classifies recommendation techniques as: 1) Knowledge-Based, 2) Content-Based, 3) Collaborative filtering, 4) Utility-Based and 5) Demographic. In recommender systems research a recent tendency is to combine recommendation techniques to achieve peak performance. All of the known recommendation techniques have strengths and weaknesses, and many researchers have chosen to combine techniques in different ways [5].

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 533–546.
springerlink.com

Some applications of hybrid recommender systems are used in Fab [3] and REJA [12]; Collaborative filtering is combined with other techniques usually [2]. In Fab recommender system a hybrid method is used, collaborative filtering and the content-based techniques are combined. In collaborative filtering rather than compute the similarity of the items, we compute the similarity of the users. Typically, for each user a set of nearest neighbor users is found with those past ratings there is the strongest correlation. Collaborative filtering does no analyses of the items at all; in fact, all that is known about an item is unique identifier. The recommendation for user is made solely on the basis of similarities to other users. The content-based to recommendation the information retrieval community is used, Text documents are recommended based on a comparison between their content and a user profile. Data structures for both of these are created using features extracted from the text documents.

In REJA author proposes a hybrid recommender system that uses the collaborative filtering and the knowledge-based techniques. On one hand, knowledge-based technique use linguistic information and makes recommendations for registered and temporary users, these recommendations are based on user information obtained explicitly. In the other hand, collaborative filtering uses the user ratings to measure the similarity with other users and create groups with similar preferences, uses this information for recommendations.

In this work, the architecture proposed in [14] is adapted to a service oriented domain, in particular restaurant recommendation.

Section 2 explains concepts related to recommender systems, Section 3 explains the methodology used by the hybrid system and the changes proposed to improve performance in restaurants domain, section 4 explains the results of the system evaluation and in section 5 presents the conclusions.

2 Recommender Systems

Recommender systems suggest items to purchase or examine based on user's preferences and interests. An early description of recommender systems was *"A system where people provide recommendations as inputs, which the system then aggregates and directs to appropriate recipients [15]."*

Recommender Systems are software tools and techniques providing suggestions for items to be of use to a user. The suggestions relate to various decision-making processes, such as what items to buy, what music to listen to, or what online news to read. This suggestion is based on the user preferences stored in the user profile. Recommender Systems obtain user information through data collection techniques: implicit or explicit [8].

2.1 Hybrid Approaches

Many recommender applications are actually hybrids; however, little theoretical work has focused on how to hybridize algorithms and in which situations one can

expect to benefit of the hybrid methods. An example is the Netflix Prize competition, in which hundreds of students and researchers teamed up to improve a collaborative movie recommender by hybridizing hundreds of different collaborative filtering techniques and approaches to improve the overall accuracy. The Burke's work [5] is a well-known survey of the design of different hybrid recommendation algorithms; Burke proposes taxonomy of recommendation algorithms. Collaborative filtering and content-based and knowledge-based recommender systems are the three base approaches.

Another characterizing dimension is the system's hybridization design is the method used to combine two or more techniques. Recommendation components can work in parallel before combining their results, or two or more single recommender systems may be connected in a pipelining architecture in which the output of one recommender serves as input for the next one[15].

3 Methodology

This research is based in the methodology proposed in [14]. In adapting the system to the restaurants domain, memory-based technique (based on learning profile) is replaced by content-based technique (based on restaurant profile) and the variables are modified in the Fuzzy Inference Systems also, the details of these changes are explained below.

3.1 Restaurant Model

An effective online recommender system must be based upon an understanding of consumer preferences and successfully mapping potential products onto the consumer's preferences [1]. Following from Pan and Fesenmaier [13], it is argued that this can be achieved through the understanding of how consumers describe in their own language a product, a place, and the experience when consuming the product or the place.

Traditionally, choosing a restaurant has been considered as rational behavior whereby a number of attributes contribute to the overall utility of a restaurant. For example, food type, food and service quality, image, and atmosphere of the restaurant, and availability of information about a restaurant, play an important role at different stages in consumer's choice-making [3],[11]. While food quality and food type have long been perceived as the most important variables for consumers' restaurant selection, situational and contextual factors have been found to be important [11]. Due to this in Kivela [11] identified four distinct types of restaurants: 1) fine dining/gourmet, 2) theme/atmosphere, 3) family/popular, and 4) convenience/fast-food; and Auty [3] identifies four types of dining-out occasions: 1) namely celebration, 2) social occasion, 3) convenience/quick meal, and 4) business meal.

In [12] REJA recommender system was developed that uses a model with four attributes: 1) service quality, 2) quantity, 3) cuisine and 4) restaurant type, these attributes are used to filtering items of the database.

This research modeled the main features of a restaurant; the idea was based on the features reviewed in [16], and takes the characteristics of the model used in REJA. In restaurant model proposed are included other relevant attributes that the user considers when deciding which restaurants to visit, also attributes have been categorized into groups.

The restaurant model is a vector with the following attributes: 1) price range, that contains the labels "cheaper", "cheap", "more or less", "expensive", "very expensive"; 2) services, divided in nine groups with the labels "parking", "payment", "kind of food", "lounges", "banquet", "atmosphere", "extra services", "language", and "others", which in turn have their own classification; and 3) cuisine type, which includes labels such as "mexican", "chinese", "japanese", "italian", "spanish", "brazilian", and many others. The restaurant model is depicted in figure 1.

price					services																		cuisine					
					parking		payment		food kind		lounges		banquet		atmosphere		ext. services		language		others								
cheaper	cheap	more or less	expensive	v. expensive	free	valet park	credit car	cash	breakfast	buffet	private	events	banquet1	banquet2	family	friends	promotions	Big Screen	English	French	holydays	Mexican	Japanese	Chinese	Brazilian	Spanish	Italian
0	0	1	0	0	1	0	1	1	1	0	0	1	1	0	1	0	1	0	1	0	1	0	1	1	0	0	0

Fig. 1. Some possible values in restaurant model representation

Figure 1 shows some possible values of the model content restaurant recommendation system modeling features with binary vectors (zeros and ones) where *one* means that the restaurant has the property that corresponds to the value. An adequate representation of the restaurant model, is shown in figure 2, in this way is displayed to the user.

Name: *Carl's Jr.*
Description: Is a hamburger restaurant, but there are also salads and desserts. The service is efficient and fast.
Address: Tecnológico Avenue, Mesa de Otay.
Url: http://www.carlsjr.com/
Need reservation: False.
Price: Cheap.
Phone: 6249343.
Hours: 10 hrs- 22 hrs.
Slug name: Carlitos
Chain: Carl Karcher Enterprises Inc.
Lat: 0.00000
Alt: 0.00000
Public date: July 26, 2011, 2:14 p.m.
Services: Fast-food, t.v., promotions, parking, birthday, party childrens, credit card, cash, security.
Cuisines: American, breakfast, salads, desserts.
Atmosphere: Family, childrens, friends.

Fig. 2. The restaurant model

The information contained in the restaurant model is useful for Content-Based, the recommendation process is detailed in the next section.

3.2 Content-Based

This paper proposes the implementation of the content-based technique, for a functional hybrid method such as [4], which allows implement a model with information about the item characteristics.

Normally this method is based in characteristics vector, which specifies keywords for each item that the user gave a high rating.

The profile is a structured representation of user interests, adopted to recommend interesting items to the user. The recommendation process consists in matching up the attributes of the user profile against the attributes of a content object. The result is a correlation value that represents the interest level of the user in that item; this value determines the items that will be recommended.

In this research the profile representation of the items with binary vectors was done, each item contains the characteristics that identify it in terms of zeros and ones; these values are compared one by one to obtain the similarity to other items. The user profile is obtained from the ratings matrix where the content-based identifies the items with high rating for the comparison, make the prediction and recommend items.

To obtain the similarity between restaurants, the content-based uses the Cosines distance (called Cosine Similarity) that calculates a value between 0 and 1 to determine the similarity between the restaurants, where *zero* represents no similar and *one* represents very similar. The formula for calculating the Cosine Similarity is as follows:

$$Cos(\vec{I_x}, \vec{I_y}) = \frac{\sum_{i=1}^{n} I_{i,x} I_{i,y}}{\sqrt{\sum_{i=1}^{n} I_{i,x}^2} \sqrt{\sum_{i=1}^{n} I_{i,y}^2}} \quad (1)$$

Where I_x represents the item profile with high rating and I_y represents the item profile with which it is compared, n represents the number of characteristics contained in the vector, $I_{i,y}$ and $I_{i,x}$ represent the characteristic which corresponds to the i position into the vectors x and y.

Content-Based will choose (based on threshold) the items in a list to recommend. The process is depicted in figure 3.

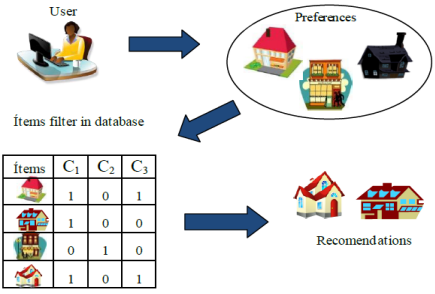


Fig. 3. Content-Based recommendation process, this technique uses user preferences to filter information of the database

3.3 User Profile

In this work the user's profile is derived from the ratings matrix. Let $U=[u_1,u_2,...u_n]$ the set of all users and $I=[i_1,iu_2,...i_n]$ the set of all items, if R represent the ratings matrix, an element $R_{u,i}$ represents a user's rating $u \in U$ in a item $i \in I$. The unknown ratings are denoted as $\varnothing$. The matrix R can be decomposed into rows vectors, the row vector is denoted as $\rightharpoonup r_u = [R_{u,1}... R_{u,|I|}]$ for every $u \in U$. Therefore, each row vector represents the ratings of a particular user on items. Also denote a set of items rated by a certain user u is denoted as $I_u = i \in I \mid \forall i: R_{u,i} \neq \varnothing$. This set of items rated represents the user preferences where for each domain element $R_{u,i} \in [1,5]$ represents the intensity of the user interest for the item.

The user profile is used in collaborative filtering and is detailed in the following section.

3.4 Collaborative Filtering

The basic idea of collaborative filtering is that if users shared the same interests in the past – if they viewed or bought the same books, for instance – they will also have similar tastes in the future, in other words, collaborative recommendation approach exploits information about past behavior or the opinions of an existing user community to predicting which items the current user of the system will most probably like or be interested in [10]. The technique can be broken down into the following steps:

1. Establish how similar other users are to the current user *u*.
2. Use this similarity to weight the ratings of these users for the recommended item *i* in a weighted sum.
3. Apply additional filters or weights.

In collaborative filtering the user profile (mentioned in previous section) is represented as a ratings vector that the user gives for each item. The K-Nearest Neighbor method (K-NN) is used to identify the correlation degree (similarity) of the users with similar tastes to the current user. To obtain the correlation between the neighboring and the current user, K-NN uses Pearson correlation which is the most commonly used [4]. Then, it shows the formula for the Pearson Correlation.

$$P(U_x,U_y) = \frac{\sum_{i=1}^{n}(U_{xi}-\overline{U})^2(U_{yi}-\overline{U})^2}{\sqrt{\sum_{i=1}^{n}(U_{xi}-\overline{U})^2\sum_{i=1}^{n}(U_{yi}-\overline{U})^2}} \quad (2)$$

Where $P(U_x,U_y)$ is the correlation coefficient with range between -1.0 and 1.0, U_x represents the current user and U_y represents other user (from community user), these users U_x and U_y are compared to obtain the correlation. Users with higher positive correlation are considered more similar.

Collaborative filtering obtains predictions using the weighted average (is the product of the user correlation and the user rating to the item) and compare the ratings of neighbors with the current user to generate a recommendations list (Top-N) based on a threshold of 0.75. The collaborative filtering process is depicted in figure 4. The collaborative filtering recommendations are inputs to the Fuzzy Inference System that is explained in the next section.

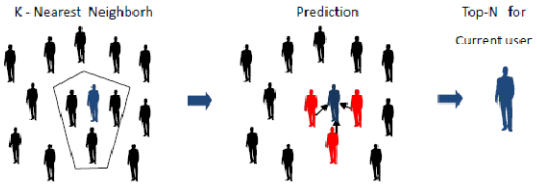


Fig. 4. Collaborative filtering process

3.5 Fuzzy Logic

Fuzzy logic is a methodology that provides a simple way to obtain conclusions from data entry vague, ambiguous, imprecise, noisy or incomplete. Is based on the traditional process of how a person makes decisions based on information with features mentioned. Fuzzy logic is a computational intelligence technique allows using information with a high degree of inaccuracy; this is the difference with the conventional logic that only uses concrete and accurately information [17].

Fuzzy logic is a multivalued logic that allows intermediate values to define evaluations between yes/no, true/false, white/black, hot/cold, small/large, near/far, few/many, etc. In the traditional set theory the element belongs or does not belong to a set, in a fuzzy set is not defined its borders and the membership degree gives a value between 0 and 1. The concept of *membership degree* is subjective and dependent on the domain [7].

In this work the importance of fuzzy logic into recommender system is modeling fuzzy variables. In recommender system was implemented two fuzzy inference systems that represent: 1) the expert recommendation, and 2) the final recommendation. The first generates recommendations when the recommendation techniques (collaborative filtering and content-based) cannot generate due to the cold start problem mainly. The second evaluates the predictions for each recommendation technique (including the expert) and assigns a weight to obtain the final recommendation. The following explains details the operation of both fuzzy inference systems.

To illustrate the use of fuzzy variables, the price variable is explained (figure 5), where the fuzzy sets are represented by membership functions labeled "low" and "high", the linguistic values are granulated into a domain of 0 to 5.

Another variable is used to measure how many items have been rated by the current user, if a user has rated few items, predictions cannot be as accurate as if the user rated more items. In this case the variable domain is 0 to 15 or more and contains three membership functions labeled as "insufficient", "minimum" and "sufficient".

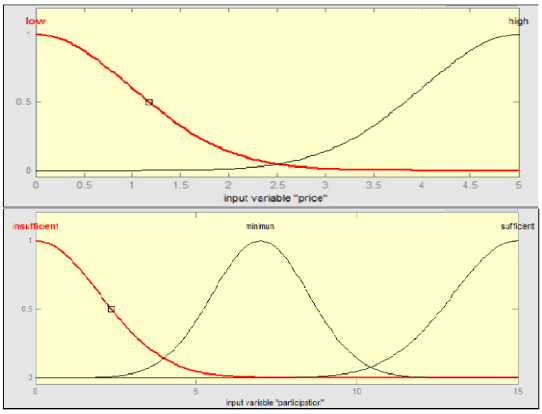


Fig. 5. The figure shows the membership function of price and participation variables

Expert Recommendation

The Instructor's recommendation in [14] is equivalent to the expert recommendation where the fuzzy variables are changed to adapt the domain. The proposed FIS in this research (figure 6), represents a user who has experience and knowledge about restaurants. The expert user specifies their predictions based in input variables: 1) rating, which corresponds to the ratings average of the other users; 2) price, which has a restaurant; and 3) votes, which is the amount of users who evaluated the restaurant. The FIS considers these factors very important to users who visit a place. This information is extracted from the user profile and restaurant profile, the system uses this information for prediction.

The FIS uses five inference rules that are designed according to the above factors. The input variables (are used in the rules) determine the recommendation activation, each input variable contains labels as "low" and "high" that also correspond to memberships functions of Gaussian type. For the output variable "recommendation" the labels "low", "medium", and "high" are used, that also correspond to membership functions of Gaussian type. The rules are:

1. If rating is high and price is low then recommendation is high.
2. If rating is high and votes is sufficient then recommendation is high.
3. If rating is high and votes is insufficient then recommendation is medium.
4. If rating is low and price is high then recommendation is low.
5. If rating is low and votes is insufficient then recommendation is low.

To recommend this FIS takes in account the suggestions of the community users (ratings) and expert knowledge. The FIS to provide a recommendation when the user participations (votes) are not sufficient and the recommendation techniques can not generate predictions.

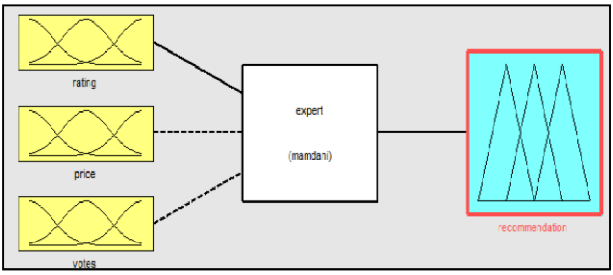


Fig. 6. Fuzzy Inference System for expert prediction

Final Recommendation

In [9], the FIS for making the final weighted recommendation is described, the inputs variables are provided by the recommendation techniques implemented in the system, the outputs variables represent the weights assigned to these recommendations.

In this work an input variable representing the Content-Based recommendation was added, the input variables in the FIS (figure 7) are:

1) *userSimilarity*, is the correlation average between other users and the current user, if the average correlation is high the system will assign a greater weight;
2) *restaurantSimilarity*, is the average similarity between restaurants, if the average similarity is high the system will assign a greater weight;
3) *participation*, is the participation (votes) of the current user, the system provides better recommendations with many number of votes.

The outputs variables are: 1) *expert*, 2) *restaurantProfile*, and 3) *correlation*, corresponding to the recommendation techniques implemented in the system, these variables play the same role as in [9]. The expert recommendation will activate when the recommendation techniques cannot make predictions, this usually happens when the current user has no ratings (or not enough) to activate the collaborative filtering, and when the current user does not have a high rating to activate the content-based.

The FIS gives weights to the input values, these weights are used to predictions and to obtain the final recommendation.

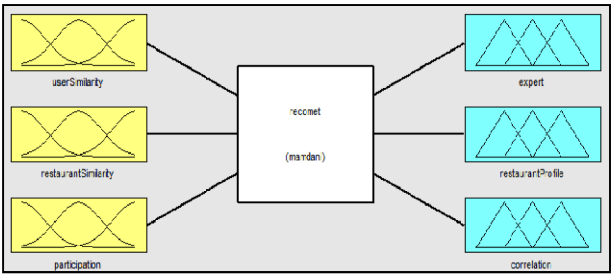


Fig. 7. Fuzzy Inference System for final recommendation

In rule one, expert recommendation is activated, here, the other techniques cannot provide recommendations. Determines that if the user similarity is low, the restaurants similarity is low and the user participations are low then the recommendation of the expert will have more weight.

1. `If userSimilarity is low and restSimilarity is low and participation is insufficient then expert is high and restProfile is low and correlation is low.`

In the rules two and three, the recommendation of collaborative filtering is activated. The difference between the rules lies in the modification of the participation variable that is expressed as "sufficient" and "minimum", respectively. In these rules the collaborative filtering recommendation will have more weight.

2. `If userSimilarity is low and restSimilarity is low and participation is sufficient then expert is low and restProfile is low and correlation is high.`
3. `If userSimilarity is low and restSimilarity is low and participation is minimum then expert is low and restProfile is low and correlation is high.`

In the rules four, five and six, the content-based recommendation is activated. The difference is the participation variable that is modified for each case to express "insufficient", "minimum" and "sufficient", then, content-based recommendation will have more weight.

4. `If userSimilarity is low and restSimilarity is high and participation is insuficient then expert is low and restProfile is high and correlation is low.`
5. `If userSimilarity is low and restSimilarity is high and participation is minimum then expert is low and restProfile is high and correlation is low.`
6. `If userSimilarity is low and restSimilarity is high and participation is sufficient then expert is low and restProfile is high and correlation is low.`

The inference rules are static and the outputs values are obtained through the centroid method.

3.6 Architecture

The final recommendation process proposed in [9] used the weighted average, was used this same method for this domain. The architecture proposed in this research with the techniques and methods explained above it shown in the figure 8.

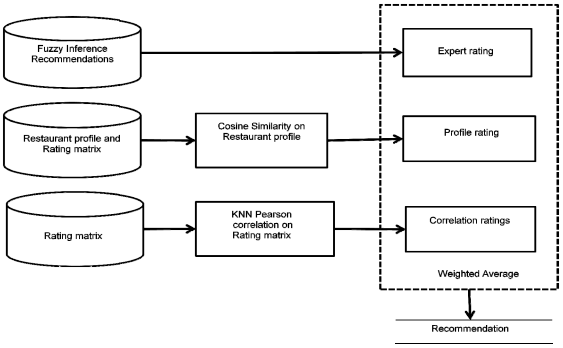


Fig. 8. System architecture, recommendations are generated in parallel

4 Results and Discussion

Database

The database is a collection obtained from information posted in restaurant's Web sites; the focus was on restaurants in the vicinity of Tijuana. Information was also gathered from the personal experience of users who have visited the restaurant and provide valuable information about the current condition of the restaurants. The database contains 60 restaurants available for evaluation by 50 users and a total of 250 ratings stored; the average is 7 ratings per user.

Experimental procedure

To evaluate the recommender system two experiments were made, both measuring recommendation accuracy using RMSE:

1) As a first experiment the accuracy of the predictions made by single recommendation technique was measured.
2) As a second experiment the accuracy of the final recommendation made with weighted average and arithmetic mean, was compared.

In first experiment, the users profiles are represented by the ratings matrix, RMSE calculates the error between a predictions vector (values proposed by the system)

$v_1 = [x_{1,1}, x_{1,2},\ldots x_{1,n}]$ and a ratings vector (or user profile) $v_2 = [x_{2,1}, x_{2,2},\ldots x_{2,m}]$, the system compare these vectors and RMSE is calculated. In (3) n represent the vector size.

$$RMSE(v_1, v_2) = \sqrt{\frac{\sum_{i=1}^{n}(x_{1,i} - x_{2,i})^2}{n}} \tag{3}$$

Five tests for predictions using two distance measures were made: Pearson correlation and Euclidean distance to measure the accuracy and determine which is more efficient with the data base, varies the number of ratings according to the number of users, the data behavior and averages are shown in Table 1.

Table 1. Comparison of results of the error using the Pearson correlation and Euclidean distance, the results show an average error of 0.86 in a ratings scale of 1 to 5 this average represent 17% of error in accuracy

Users	Ratings	Average ratings	Average predictions with Pearson correlation.	Average predictions with Euclidean distance	RMSE with Pearson correlation	RMSE with Euclidian distance
10	45	3.79	3.17	3.18	0.87	0.87
20	89	3.74	3.15	3.17	0.91	0.87
30	156	3.87	3.38	3.40	0.77	0.87
40	191	3.71	3.29	3.30	0.82	0.83
50	250	3.60	3.14	3.20	0.83	0.83

Second experiment was done to verify the recommendations accuracy of the weighted average, with the method proposed. The aim research is to verify that using the weighted average can show better results than using the arithmetic mean in the accurate of the recommendation. Was used 30 predictions randomly taken from user community, the recommendation was made using the weighted average and the arithmetic mean. The results are data with similar patterns of error. The average test showed a prediction of 4.81 for the weighted average and 4.77 for the arithmetic mean. Table 2, shows the results in the predictions.

Table 2. Comparison results with weighted average and arithmetic mean

Prediction	W.Avg.	A.Mean	Prediction	W.Avg.	A.Mean
1	4.92	4.86	16	4.82	4.71
2	4.89	4.86	17	4.66	4.58
3	4.51	4.50	18	4.88	4.85
4	4.90	4.86	19	4.86	4.85
5	4.92	4.86	20	4.87	4.85
6	4.89	4.84	21	4.89	4.85
7	4.87	4.87	22	4.50	4.38
8	4.91	4.86	23	4.86	4.86
9	4.84	4.80	24	4.85	4.85
10	4.87	4.84	25	4.54	4.61
11	4.88	4.86	26	4.77	4.61
12	4.57	4.35	27	4.83	4.71
13	4.85	4.85	28	4.89	4.85
14	4.86	4.86	29	4.85	4.85
15	4.87	4.80	30	4.85	4.85

5 Conclusions and Future Work

The architecture [14] was adapted easily to the restaurants domain, the adaptation allowed to add new techniques and adding a user profile with the linguistic description of their tastes and preferences. Seen from another perspective, a rating says more than just a number, even the rating can have implicit factors as the quality food, decor, comfort, customer service, etc.., this information can be extracted and specified in the user profile to obtain better recommendations.

Was proposing the implementation of a Geo-referencing system that allow collect contextual information (of the user) implicitly, such as physical location or the distance between the restaurant and physical location. Also aims to collect explicit information about the user's current conditions, variables such as money, transportation or friends, may be relevant when user make the decision to go to a restaurant.

The experimental tests performed on the prototype determine that the distances used are effective for the purpose of the investigation, however, with Pearson correlation did had a better performance into the set of test data. Certainly, the recommendation techniques may behave differently with each data set, perhaps to a data set of films (for instance MovieLens) collaborative filtering has more accuracy, but in a data set of restaurants may not be as accurate. In future work can evaluate other important factors (for instance quality, diversity, and utility) to determine the system performance in other terms.

Finally, the result of the second experiment shows that there was only a small difference between the two averages, although the architecture is specified the use of weighted average, the arithmetic mean may be used for the same purpose without affecting system performance.

References

[1] Adomavicius, G., Tuzhilin, A.: Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. IEEE Transactions on Knowledge and Data Engineering 17(6), 734–749 (2005)

[2] Ahn, H.: A New Similarity Measure for Collaborative Filtering To Al-leviate the New User Cold Starting Problem. Information Sciences, 37–51 (2008)

[3] Auty, S.G.: Consumer choice and segmentation in the restaurant industry. Service Industries Journal 12(3), 324–339 (1992)

[4] Balabanovic, M., Shoham, Y.: Fab: Content-Based, Collaborative Recommendation. Communications of the ACM 40(3), 66–72 (1997)

[5] Burke, R.: Hybrid Recommender Systems: Survey and Experiments. Departament of Information Systems and Decision Sciences, 1–29 (2002)

[6] Claypool, M., Gokhale, A., Miranda, T., Murnikov, P., Netes, D., Sartin, M.: Combining Content-Based and Collaborative Filters in an Online Newspaper. Computer Science Department, 1–8 (1999)

[7] D' Negri, C., De Vito, E.: Introduction to approximate reasoning: fuzzy logic, vol. 4, pp. 126–136 (2006) (in Spanish)

[8] Galán Nieto, S.M.: Collaborative filtering and Recommender Systems. Communications Networks Intelligence (2007) (in Spanish)
[9] García-Valdez, M., Alanis, A., Parra, B.: Fuzzy Inference for Learning Object Recommendation. In: Proceedings of WCCI, Barcelona, Spain (2010)
[10] Jannach, D., Zanker, M., Felfernig, A., Friedrich, G.: Recommender Systems, an introduction. Cambrige University Press (2011)
[11] Kivela, J.: Restaurant marketing: selection and segmentation in Hong Kong. International Journal of Contemporary Hospitality Management 9(3), 116–123 (1997)
[12] Martínez, L., Rodríguez, R.M., Espinilla, M.: REJA: a georeferenced hybrid recommender system for restaurants. In: IEEE/WIC/ACM Conference on Web Intelligence and Intelligent Agent Technology, pp. 187–190 (2008)
[13] Pan, B., Fesenmaier, D.R.: Online information search: vacation planning process. Annals of Tourism Research 33(3), 809–832 (2006)
[14] Parra, B., García-Valdez, M.: A Hybrid Recommender System Architecture for Learning Objects. In: Castillo, O., Pedrycz, W., Kacprzyk, J. (eds.) Evolutionary Design of Intelligent Systems in Modeling, Simulation and Control. SCI, vol. 257, pp. 205–211. Springer, Heidelberg (2009)
[15] Resnick, P., Varian, H.: Recommender Systems. Communications of the ACM 40(3), 56–58 (1997)
[16] Xiang, Z., Kim, S.E., Hu, C., Fesenmaier, D.R.: Language representation of restaurants: Implications for developing on-line recommender systems. Hospitality Management 26, 1005–1018 (2007)
[17] Zadeh, L.A.: Knowledge Representation in Fuzzy Logic. IEEE Transactions on Knowledge and Data Engineering 1(1), 89–99 (1989)

User Modeling for Interactive Evolutionary Computation Applications Using Fuzzy Logic

J.C. Romero and Mario García-Valdez

Division of Graduate Studies and Research, Tijuana Institute of Technology,
Tijuana, México
jcromerohdz@gmail.com, mgarcia@tectijuana.edu.mx.

Abstract. Interactive evolutionary computation (IEC) is a branch of evolutionary computation where users are involved in the evolution process. In IEC systems the user generally evaluates subjective information of the population in large quantities. One of the problems in the IEC systems is not having friendly interfaces for the evaluation of mass information and this causes the user lose interest. These systems have quickly migrated to the Web by the large number of users that can be found on a voluntary basis. For these applications we can find users with different characteristics, for example, users with different level of knowledge about the application domain, different participation interest or experience in use of Web-Based IEC applications. In this paper we propose a user modeling for IEC to help tailor the user interface depending on the characteristics, preferences, interests, etc. of the user using fuzzy logic.

Keywords: User modeling, interactive evolutionary computation, IEC, Fuzzy logic.

1 Introduction

Interactive evolutionary computation (IEC) is a branch of evolutionary computation where users participate evaluating individuals in a population given by an application. It is based on subjective human evaluation [12].Users participate in such systems depending on their interest of the application domain and because of this several of these systems are Web-based. Literature tells us that in such systems there is a problem that has to do with the user interface, sometimes these user interfaces are not friendly for the user to interact with and this causes them to lose interest [12]. Because of this we proposed to use user-modeling techniques to personalize the user interface for a given application domain on interactive evolutionary computation. The proposed User model will provide us with the preference and needs according to the interface currently used by the user. With this information we can use fuzzy ruled based system to infer in what type of user is in a given time, then knowing the type of user we then can perform the interface adaptation.

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 547–557.
springerlink.com

2 Interactive Evolutionary Computation

This technology is branch of evolutionary computation. Based on subjective human evaluation. Basically, this technique requires that the objective function is replaced by a person (user) [12]. In Fig 1 we show a general IEC system.

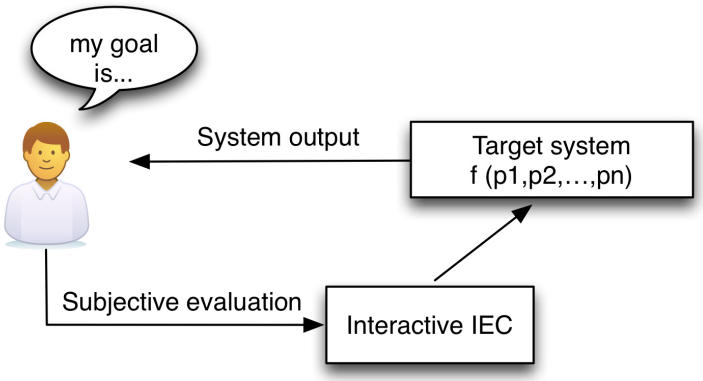


Fig. 1. General Interactive Evolutionary Computation system based on subjective evaluation. [12]

In this general IEC system we note that the user replaced the fitness function at the moment he or she interacts with the system. Logically the user needs a goal to know what is the evaluation he or she needs to perform. Finally the users receive an output and the process and start over again.

These techniques have been used in several areas of application, in particular:

- Music and sound.
- Digital Art.
- Design and editing documents
- Processing acoustic signals.
- Industrial design.
- Data mining and acquire knowledge.
- Face recognition.
- Robotics and control.

2.1 Problems and Limitations

In IEC there are some problems and limitations whereby the user is not comfortable to interact with these kinds of systems, Takagi identifies [12]:

- User fatigue: 20 to 30 iterations
- User memory: limits the maximum number of individuals per generation.
- User selection: Simple user interfaces.
- Stop condition: User dependent.
- Fluctuations in decisions: Changes in preferences.

This is why the user loses interest to participate in this kind of systems, in these paper we will focus in one of the problems, the user selection, we want to increase their participation in these applications by presenting a simple and usable interface considering their characteristics, preferences, interest, etc..

3 User Modeling

There is a reality that the World Wide Web is growing in an exponential way, which means the presence of more users in Web sites, Web systems are very varied, and we have users of any kind. These users follow different goals, for example make a reservation of a room in a five stars resort or just consulting their bank account or checking their status in Facebook. This variation of users represents a diversity of individuals, individuals with different abilities, interest, preferences, forms of learning, and knowledge, because of this they need different presentation of the information that they use to interact with the different variety of applications.

When we are dealing with users in systems that we want to personalize, we need to know some personal information about the users. This information is a collection of their needs, characteristics, feelings, etcetera to adapt something that we want in a system. This information is needed to represent knowledge about the user and all this is called user modeling (UM) [11].

A user modeling can be a simple profile system that represents some knowledge about them or can be a very complex representation of their characteristics, needs, and interest to understand a specific individual.

The main goal of a user modeling is to represent real world aspects of the users in an automatically autonomous form. There are several user models techniques, these UM techniques are the most frequently used:

- Rule based models.
- Ontologies based models.
- Stereotype based models.
- Hybrid models.

In next sections a brief description of these technique are presented.

3.1 Rule-Based Models

This type of user model is based on the experience of an expert in a given context, this means that rules can be made by that knowledge of the expert has, for example, an expert can make rules to represent that knowledge to a user model to infer in the personalization of a given application. In this way we can represent the user model [11].

3.2 Ontology-Based Models

Ontologies-based models are forms of representing knowledge starting by the idea of a well form semantic, "An ontology is a formal specification of shared specification"[7]. For example the semantic Web is based on ontologies that is an important part of this approach.

We can understand better an ontologies-based UM by the system architecture that Liana Razmerita [7] presented in her work, in Figure 2 we are present the mentioned architecture.

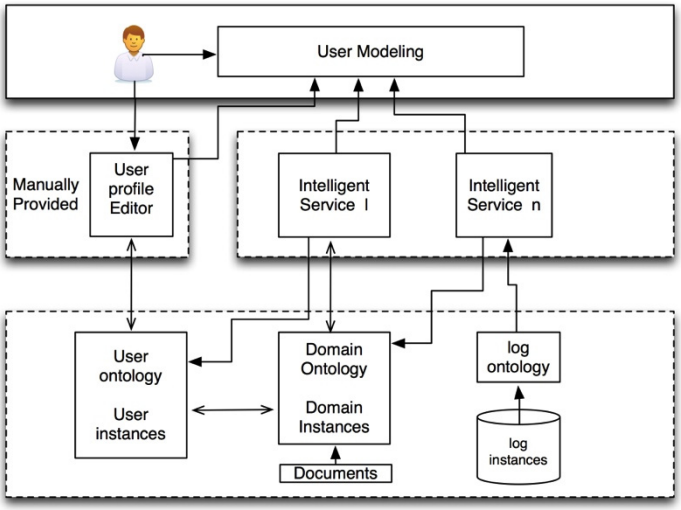


Fig. 2. An ontology-based user modeling system

Here in this architecture (Fig. 2) the ontology part is represented in three important aspects:

1. The User ontology structure represents distinct characteristics in their relationships.
2. The Domain ontology is defined by the context of the specific application and their relationships.
3. The log ontology represents the semantic of the user interaction with the system.

Monitoring the user interaction with the system generates the log data storage.

The intelligent services have two principal aspects in the system:

1. To maintain and update based of the data usage through the application by the number of heuristics.
2. To adapt service based on the needs of the user like skills, preferences, and knowledge.

3.3 Stereotype Based Models

This kind of user models is based on demographic statistics. This technique simply uses collections of facet-value combinations that describe groups of system users. The user is classified into common stereotypes. Using this technique the system can infer assumptions about the user if there might be no data about that particular context, because there is fact that demographic studies have shown that other users have the same characteristics given by the stereotype.

In figure 3 we present a Stereotype Hierarchy in GRUNDY [9].

This stereotype hierarchy representation adjusts the user model by an explicit representation of knowledge. The advantage of stereotype-based user model is with a small evidence about the user a system should infer a very decent model information.

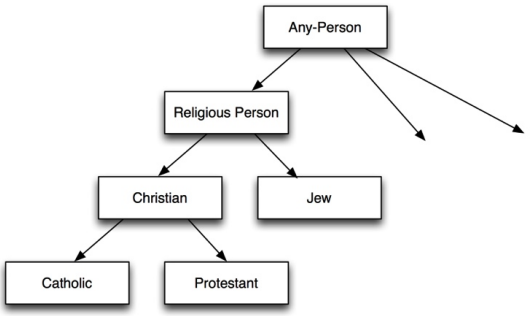


Fig. 3. Stereotype Hierarchy in GRUNDY[9]

The advantage of stereotype-based user model is with a small evidence about the user a system should infer an appropriate model information.

3.4 Hybrid Based Models

This kind of user models is just the combination of one or more techniques that we explained above. We can use this technique when we compare that the technique we used is better in particular cases so we produce the UM in a kind a switch state. For example if in the context of e-learning we have good results with rule-based technique and not too good in e-commerce we need to adjust in a particular context which technique we want to used. That is why we combine user-modeling techniques to gain better results [6].

4 Uncertanity in User Modeling

The current activities on the web have an increasing demand of preference discovery, on these days the user preferences information has become in a major tool for several areas as Marketing, eLearning and too many others. For example it is very common to find several applications to provide suggestions for some products based on recent purchases.

For that reason user preference provides multiple applications as an option for recommender systems in order to solve problems of personalized recommendations.

It's a quite of a challenge to deal with the description of user knowledge in order to recognize user's specific needs because these data involves imprecision and uncertain data.

Different from human interaction where we learn by explicit or implicit methods, it is necessary to design a user model to detect user preferences, profiles, choices, likes and dislikes [2].

In relation to this subject it is important to improve preference modeling for recommender systems to provide value added to the current web activities to succeed in an increasingly competitive world.

When a user interacts with systems, provides a high-grade uncertainty information, this is the main reason of the user modeling systems existence in order to control uncertainly data with certain degree [2].

These systems need to collect all the user information about their profiles, interaction history, choices and basic data with decision criteria.

In previous works we can find that the more information a user model has, the better the content and presentation will be personalized.

It is known that there exist different methods to manage uncertainty for user modeling, and is important to mention three general approaches:

- Rules with certainty factors
- Fuzzy Logic
- Bayes probability Networks

Fuzzy Logic goes more than a 0-1, truth or false ranges, it is based on many-valued logic and has the ability to handle partially answers with ambiguity. The general methodology of reasoning in fuzzy logic is by the IF…THEN rules [13].

Fuzzy Logic defines a framework in which the inherent ambiguity of real information can be captured, modeled and used to reason with uncertainty [13].

A traditional FL inference system is divided into three steps:

1. Fuzzification.
2. Fuzzy Inference.
3. Defuzzification.

Typically FL has been used in UM to implement applications that are based on a recommendation task. In these applications FL provides the ability of mixing different user preferences and profiles that are satisfied to a certain degree [4].

5 Proposed Method

Our proposed is to attract the users to participate in given IEC application and to improve their overall experience with the system. With the UM our expectations are user participation increases. We are going to explain in more detail with the architecture that we proposed in figure 4.

Is worth mention that we are focusing in the Web-based IEC application, in other words we want some how to adapt the interface of the IEC application to the user that is interacting with.

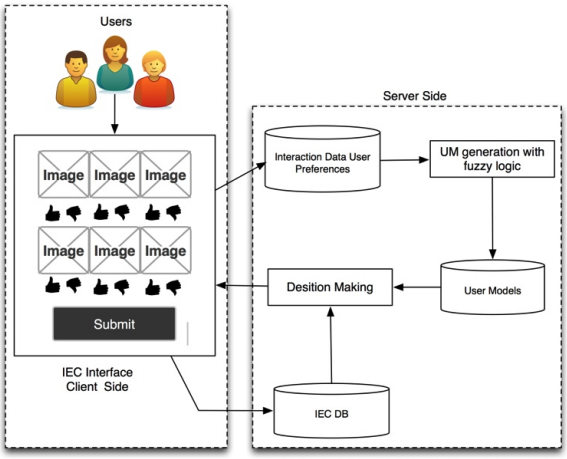


Fig. 4. Architecture for IEC using Fuzzy Logic

In Web-Based IEC applications we found that there are different types of users. For this reason we define three types of user for Web-Based IEC applications these are:

- Specialized users.
- Ordinary users.
- Anonymous users.

5.1 Specialized Users

These kinds of users are users that know all the functionality of the IEC application where they going to participate, in other words they know all the objectives of the IEC application and all the aspects of the goals that the particular application is going to, because of this they need all the functionality elements in the interface to interact with the IEC application.

5.2 Ordinary Users

These types of users are users that know certain knowledge about the goals and functionality of the IEC application where they going to participate, so they do not need all the functionality and elements of the interface to interact with, but these types of users are more attracted by the domain of the IEC application.

5.3 Anonymous Users

These classes of users are users that they feel some sort of empathy with the domain of IEC application but these classes of user have less degree of participation in the IEC application, for this reason the elements and functionality of the IEC application interface have to be very simple.

As you can see in the above architecture there is a user interface that represents the interactivity with the IEC application, that interface is the one we want to adapt to the different types of users previously defined.

5.4 Architecture for IEC Using Fuzzy Logic

Continuing with explanation of the UM architecture for IEC using fuzzy logic there is a data store mechanism that represent the preferences, interests of the users by the iterations in the IEC application. This mechanism is a monitoring of the IEC users in accordance to their different type of users.

Thanks to that mechanism we now have to store the preferences that we were provided by different types of users, now in the architecture there is a fuzzy UM generation process, where we propose to apply a fuzzy inference system (FIS) to perform the process of knowing what type of user is interacting with the IEC system. It is worth mentioning that there are several ways to accomplish this fuzzy inference system, we propose to develop according to the time the user spends using the IEC system and the difficulty of the task presented to the user. Under these conditions we can have an acceptable output in the fuzzy inference system, which represent the type of user. In Figure 5 we present is fuzzy inference system.

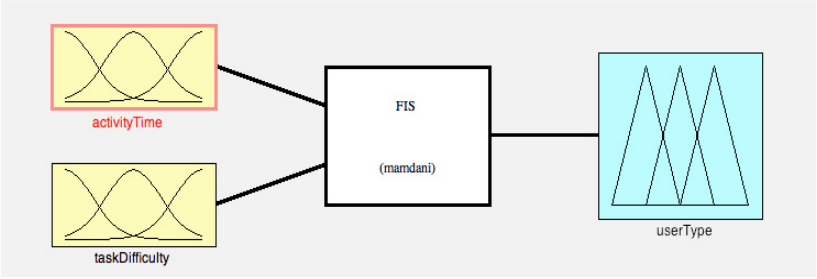


Fig. 5. IEC user interface Fuzzy Inference System

In this FIS we propose to use triangular membership functions in the input and also to the output. We also propose to use Mamdani type for simplicity. Where the ranges of activity could go from 0 to 30 that would represent the time of user activity in the IEC system, also the range of tasks (evaluations) in the sense of difficulty that could go from 0-100.In Fig 6 we present the inputs and the output that might be.

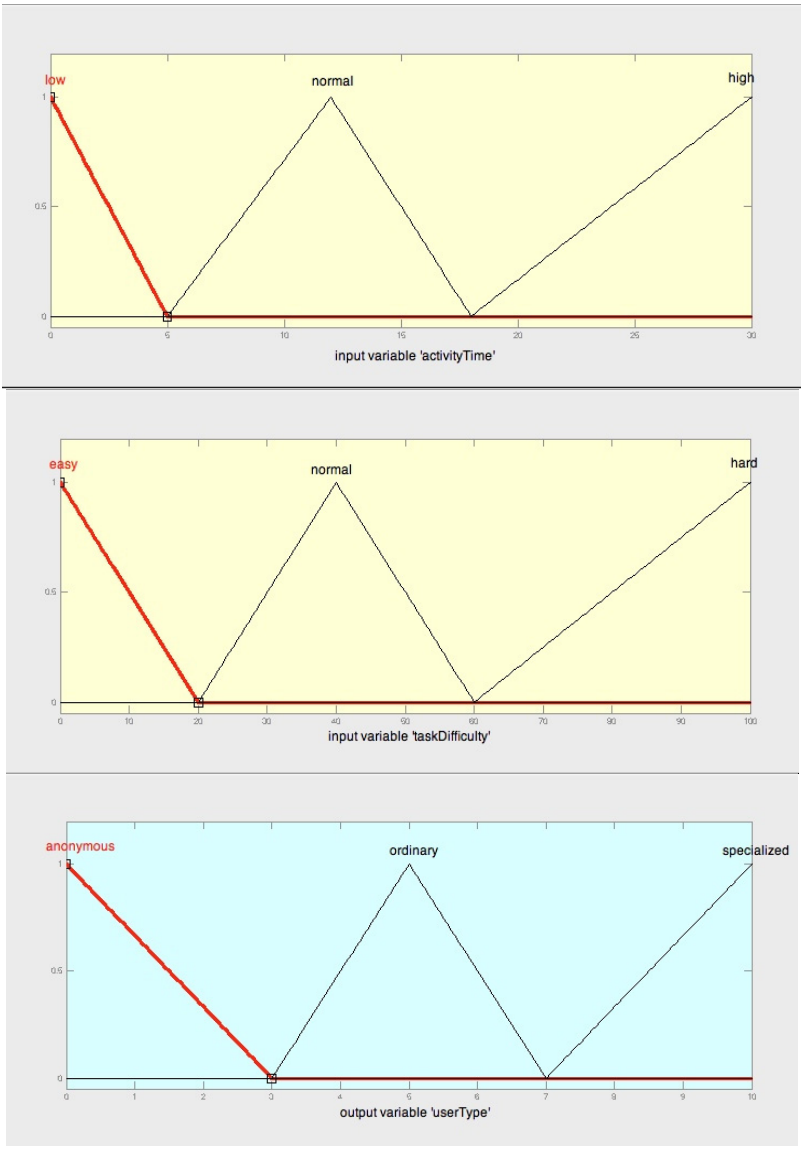


Fig. 6. Membership functions for the inputs and the output

Some of the fuzzy rules could be the following:

1. *if(activityTime is low) and (taskDifficulty is easy) then (userType is anonymous).*
2. *if (activityTime is normal) and (taskDifficulty is normal) then (userType is ordinary).*
3. *if (activityTime is high) and (taskDifficulty is hard) then (userType is specialized).*

Ones we have this rules we can proceed to store the inferences we obtain by FIS, then proceed to the process of the decision-making. In the decision-making we make the decisions based on the results of the FIS give, for example if we have an anonymous user, now we perform personalization for the interface on the IEC system, similarly when the user is ordinary we need to perform personalization on the interface elements, finally when the user is a specialized we will perform the personalization on the interface elements for IEC system.

6 Conclusion

We this method that we proposed we expect to increment the level of participation that the users have on interactive evolutionary computation applications. That means we reduce the non-friendly user interface that this kind of applications have.

Future work following this method is to tested in a real interactive evolutionary computation application. First phase by given the user a normal IEC application with a predetermined user interface and see what is the level of participation of the users. Then a second phase is to put this method on the IEC application and see if the level of participation of the users increases.

References

[1] Brusilovsky, P., Kobsa, A., Nejdl, W. (eds.): Adaptive Web 2007. LNCS, vol. 4321. Springer, Heidelberg (2007)

[2] Frias-Martinez, E., Magoulas, G., Chen, S., Macredie, R.: Recent Soft Computing Approaches to User Modeling in Adaptive Hypermedia. Department of Information Systems & Computing Brunel University, Uxbridge, Middlesex. United Kingdom. Rich, E. User Modeling via Stereo-types. Cognitive Science: A Multidisciplinary Journal 3(4), 329–354 (1979b)

[3] Jang, J.-S.R., Sun, C.-T., Mitzutain, E.: Neuro-Fuzzy and Soft Computing A Computational Approach to Learning and Machine Intelligence (1997) ISBN 0132610663

[4] Kavcic, A.: Fuzzy User Modeling for Adaptation in Educational Hyper-media. In: IEEE 2004 Faculty of Computer and Information Science, 1000. Man, and Cybernetics—Part C: Applications and Reviews, 34(4). University of Ljubljana, Ljubljana (November 2004)

[5] Kowaliw, T., McCormack, J., Dorin, A.: An Iteractive Electronic Art System Based on Artifitial Ecosystemics. Institut Systémes Complexes, Paris, France (2005)

[6] Nguyen, H., Santos Jr., E., Smith, N., Chuett, A.S.: Hybrid User Model for Information Retrieval. National Geospatial Intelligence Agency Grant No. HM158-04-1-2027 and UWW Grant for Undergraduate Research, American Association for Artificial Intelligence (2006), http://www.aaai.org

[7] Razmerita, L., Angehrn, A., Maedche, A.: Ontology-based Modeling for Knowledge Management System. INSEAD, CALT-Centre of Advanced Learning Technologies, 77300, Fontaine bleau, France (2003)

[8] Rich, E.: Building and Exploiting User Models. Unpublished PhD thesis. Carnegie Mellon University, Pittsburgh, PA (1979a)

[9] Rich, E.: User Modeling via Stereotypes. Cognitive Science: A Multidisciplinary Journal 3(4), 329–354 (1979b)

[10] Secretan, J., Beato, N., D'Ambrosio, D.B., Rodriguez, A., Camp-bell, A., Folsom-Kovarik, J.T., Stanley, K.O.: Pic breeder: A Case Study in Collaborative Evolutionary Exploration of Design Space. Department of Electrical Engineering and Computer Science, University of Central Florida. Evolutionary Computation Journal, MIT Press (2011)

[11] Sosnovsky, S.: Ontological Technologies for User Modeling. State-of-the-Art Paper Submitted to the Information Science PhD. Committee of the School of Information Sciences, University of Pittsburgh as Part of Requirements for the Comprehensive Examinations (November 29, 2007)

[12] Takagi, H.: Interactive Evolutionary Computation: Fusion of the capabili-ties of EC Optimization and Human Evaluation. IEEE (2001)

[13] Zadeh, L.A.: Fuzzy sets. Information Control 8, 338–353 (1965)

[14] Zenebe, A., Zhou, L., Norcio, A.F.: User preferences discovery using fuzzy logic. Fuzzy Sets and Systems 161, 3044–3067 (2010)

Personalization of Learning Object Sequencing and Deployment in Intelligent Learning Environments

Francisco Arce and Mario García-Valdez

Tijuana Institute of Technology, Tijuana México
francisco.j.arce@gmail.com, Mario@tectijuana.edu.mx

Abstract. The update and emergence of new technologies changed the traditional methods for learning in artificial environments (physical or virtual), as interactive tables, more powerful smartphones, tablet PCs, cameras that perceive depth (kinect). In this work we propose a new approach and the personalization of learning objects that we will call environmental learning object and deploy them on an intelligent learning environment in which we will use a single extension of the simple sequencing standard.

1 Introduction

A learning environment is defined as a "place" or "space" where the process of knowledge acquisition occurs. Learning environments are those areas where conditions are created for the individual to appropriate new knowledge, new experiences, new elements that generate processes of analysis, reflection and appropriation. Some of these systems use learning resources called learning objects. For the exchange of learning objects between systems standardization initiatives have been developed and there are some implementations and repositories that manage the content using these standards. Learning objects usually have the following characteristics are self-contained, each learning object can be used independently, are reusable, a learning object can be used in different contexts, can be added, learning objects can be grouped into collections and are labeled with metadata. Each learning object has associated certain information that describes it. This facilitates reuse by automatic means. In this work we propose a new approach and the personalization of learning objects that we will call environmental learning object. Environment Learning Objects (ELO) can contain a configuration of educational content presented by multiple heterogeneous devices, with metadata describing educational as well as technical data needed for their selection and adaptation. We propose extending current learning object standards to add IE capabilities. By tagging through the metadata we can make a selection of learning objects to create and manage different contexts when applied to a learning environment where each learning object is selected and routed to a device that

O. Castillo et al. (Eds.): Recent Advances on Hybrid Intelligent Systems, SCI 451, pp. 559–567.
springerlink.com

operates its utility like sound file on a speaker of web page un a monitor. Below we will see theory regarding issues of interest of this work.

An **intelligent learning environment** is a new kind of intelligent educational system which combines the features of traditional Intelligent Tutoring Systems (ITS) and learning environments. An intelligent learning environment (ILE) includes special component to support student-driven learning, the environment module. The term environment is used to refer to that part of the system specifying or supporting the activities that the student does and the methods available to the student to do those activities [8]. Some recent ITS and ILE include also a special component called manual which provides an access to structured instructional material. The student can work with the manual via help requests or via special browsing tools exploring the instructional material on her own. An integrated ILE which includes the environment and the manual components in addition to regular tutoring component can support learning both procedural and declarative knowledge and provide both system-controlled and student-driven styles of learning.

An **intelligent tutoring system** (ITS) is a computer system that aims to provide immediate and customized instruction or feedback to its learners during a task, without intervention from humans. ITSs have maintained the common goal of enabling learners to acquire information in a meaningful and effective manner by employing tools from a range of different technologies to direct the task at hand. There are numerous examples of ITSs being used in both education and professional settings since the mid-1920s. As a result, there is a strong relationship between ITSs, cognitive learning theories and instructional design. As with any mechanism for learning, ITSs have experienced its share of successes and limitations with continuous research investigating the best approach to addressing the dialogues and affective responses to learning.

Research in ITS, organizes the "problem" in (1) knowledge about a domain, (2) knowledge about the learner, and (3) pedagogy (knowledge of teaching strategies). The major components of a typical ITS are an expert (or domain) model, student model and tutoring model. The expert model should be able to solve the problems the tutoring module submits to the students. The tutor module controls the interaction with the student, based on its teaching knowledge and comparisons between the student model and the domain knowledge. The student model reflects what the machine can infer about the student's cognitive state.

Learning environments are typically constructivist in nature, engaging learners in "sense-making" or reasoning about extensive resource sets. Learning environments typically include four components: an enabling context, resources, a set of tools, and scaffolds [8]. Authentic or realistic contexts are provided to motivate learners, and typically take the form of complex, full-scale problems representative of real-world tasks. To help students understand their complex problems, extensive resources can be provided. A truly open-ended learning environment would involve students in independent research to find and select their own relevant resources (e.g., in the campus library, on the internet). In some learning environments, however, selected resource sets are provided to learners. A full set of tools should be provided to help learners to process information, manipulate data, and discuss the data. Scaffolds should also be present to bolster student problem solving as needed. These can take many forms from tools to teachers to student peers.

A **Learning object** is "a collection of content items, practice items, and assessment items that are combined based on a single learning objective" [2]. The author of the definition is Wayne Hodgins when he created a working group in 1994 though the concept was first described by Gerard in 1967 [16]. Learning objects go by many names, including content objects, educational objects, information objects, intelligent objects, knowledge bits, knowledge objects, learning components, media objects, reusable curriculum components, nuggets, reusable information objects, reusable learning objects, testable reusable units of cognition, training components, and units of learning. The Institute of Electrical and Electronics Engineers (IEEE) defines a learning object as "any entity, digital or non-digital, that may be used for learning, education or training", Chiappe defined Learning Objects as: "A digital self-contained and reusable entity, with a clear educational purpose, with at least three internal and editable components: content, learning activities and elements of context. The learning objects must have an external structure of information to facilitate their identification, storage and retrieval: the metadata."[8].

Learning Objects and Metadata is based on the object-oriented paradigm; learning objects are typically defined as components of instruction material, which can be reused in multiple contexts. Instruction designers can create and maintain these components. Learning objects can range from complex simulations, to videos, images, quizzes, or simple text. Learning objects are the basic elements of current Learning Management Systems (LMS) and are the focus of standardization initiatives whose goal is defining open technical standards and their characteristic metadata [4]. The most important initiatives are the Advanced Distributed Learning Initiative (ADL-SCORM) [15], the Instructional Management System Project (IMS) [1], the Alliance of Remote Instructional Authoring Distribution Networks of Europe (ARIADNE) [10], and the IEEE Learning Technology Standards Committee [11]. The main objective of these open standards is to enable the interoperability of learning objects between different LMSs and Learning Objects Repositories (LORS). Basic metadata schema specifications for learning objects include:

Learning Object Metadata (LOM). Based on the Dublin Core metadata [12] this specification defines a set of meta- data elements that can be used to describe learning resources. LOM includes educational, relation, technical, and classification elements [8].

Content Aggregation Model (CAM). CAM defines a package for the aggregation, distribution, management, and deployment of learning objects. Defines an organization element which contains information about one particular, passive organization of the material, the organization for now is limited to a tree structure [9].

Learner Information (LI). A collection of information about a learner or a producer of learning content, the elements are based upon accessibilities; activities; affiliations; competencies; goals; identifications; interests; qualifications, certifications and licenses; relationship; security keys; and transcripts [9].

Sequence and Navigation (SN). SN defines a method for representing the intended behavior of an authored learning experience such that any Learning Technology system (LTS) can sequence discrete learning activities in a consistent way. Provides a rule based sequencing of behaviors [9].

These standards have been the basis for various research projects in eLearning [15] and also extensions to support adaptability have been proposed [17], [18]. Certain limitations of these specification initiatives have been noticed mainly regarding their weak support for the instructional design of the educational resources and pedagogy [15].

The **IMS Simple Sequencing** Specification defines a method for representing the intended behavior of an authored learning experience such that any learning technology system (LTS) can sequence discrete learning activities in a consistent way. The specification defines the required behaviors and functionality that conforming systems must implement. It incorporates rules that describe the branching or flow of instruction through content according to the outcomes of a learner's interactions with content. The Specification was released to the public in March 2003." [8].

IMS SS finds application in a number of areas of e-learning. Principal among these are:

- Creating multiple paths through one set of learning activities.
- Creating patterns of formative assessment.
- Creating summative assessments.
- Creating Decision Trees.
- Adding Context Sensitive Network Services to Content.
- Creating Step Through Tutorials.

IIMS SS works by a computer system reading instructions and determining which activity to present next to the learner. The instructions are known as sequencing information. The process of determining the next activity is called sequencing behavior. Sequencing information is broken down into two categories: the definition information model and the tracking information model. Definition information is added by an author or a tool to an XML file. Currently the only XML file recognized for this purpose is the imsmanifest.xml file used in IMS CP. The result is one or more activity trees integrated with a set of sequencing instructions.

This paper is ordered as follows we will see the background in section on the section 3 we will see the work in progress of the doctoral thesis and the conclusions on the section 4.

3 Proposed Approach

In this work we propose the implementation of an intelligent learning environment in which we will use an extension of the simple sequencing standard and a new kind of learning object.

First we propose the Environmental Learning object, a standard learning object is defined in [2]. The teacher prepares teaching materials using content from various sources; select different pieces of information that subsequently assembled to form the course or class to teach. Learning objects are based on this methodology, seeing the didactic content as a component that is designed to combine with others and to be used in different contexts; learning objects usually have the following characteristics:

- They are self-contained. Each learning object can be used independently.
- Are reusable. A learning object can be used in different contexts, for multiple purposes.
- Can be added. Learning objects can be grouped into collections, after being presented with a traditional course structure.
- Are labeled with metadata. Each learning object has associated certain information that describes it. This facilitates reuse by automatic means.

By tagging through the metadata we can make a selection of learning objects to create and manage different contexts when applied to a learning environment as we can note in Figure 1.

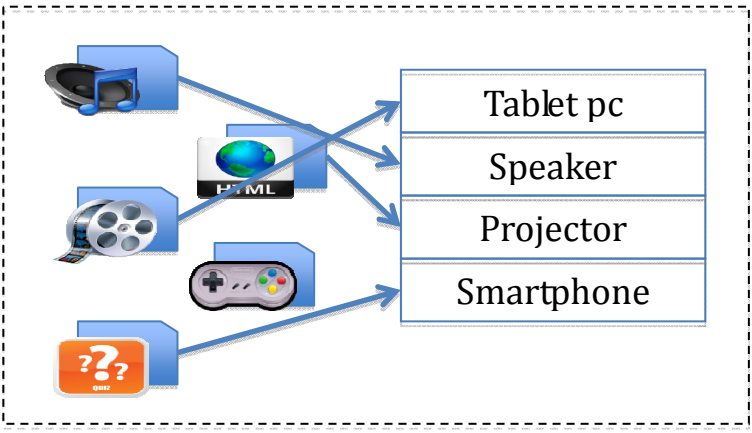


Fig. 1. Example of Environmental learning object. here only the learning object "game" is not being selected may be for several reasons, one of them is there is no device for use or is not within the context of the activity

Where each learning object is selected and routed to a device that operates its utility like a sound file that can't be used on a monitor maybe on a Smartphone or a tablet pc, these are the types of problems that can occur when using these learning objects. Also this ELO will give contextual information for feedback. Then the learning object is attached to the sequence as we can see on figure 2.

The main components of the Simple Sequencing standard are the Learning Activities and the Activity Tree. A Learning Activity is defined as a pedagogically neutral unit of instruction, knowledge, evaluation, etc. Learning Activities can have nested sub-activities arbitrary depth.

There is an implicit hierarchy of containers in the tree. Depending on the application concept labels can be applied to learning activities. Only leaf nodes can be associated with Resources Activity (the equivalent of Learning Objects).

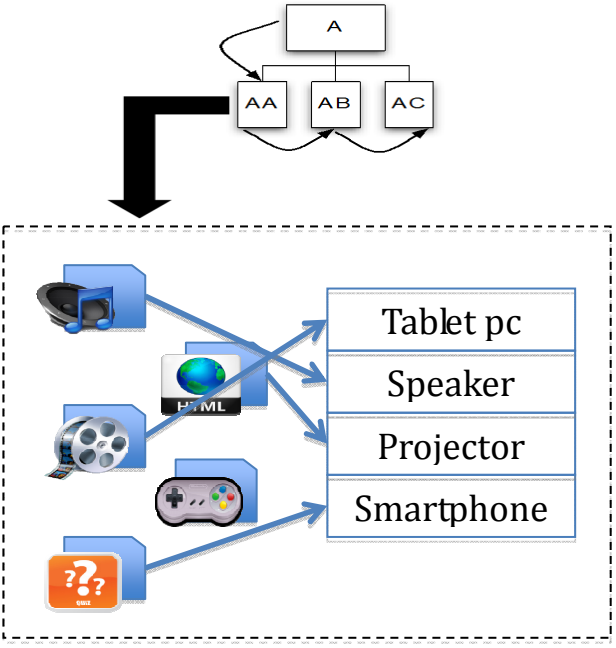


Fig. 2. Configuration of the activity. In this figure shows how the learning objects are utilized in its respective device

An example of the configuration obtained by sequencing can be observed in Figure 3. The tree is traversed as follows from the root there is the General activity (can be any subject in particular) with 2 nodes in it, we see that has the attribute "forwardOnly" on true, this means they have to be traveled sequentially by users, the first activity is a pre-evaluation, which is associated with a learning object in this case a test that contains a rule that makes the activity is carried out or otherwise will not advance to the next activity, as specified in the Pre_condition rule.

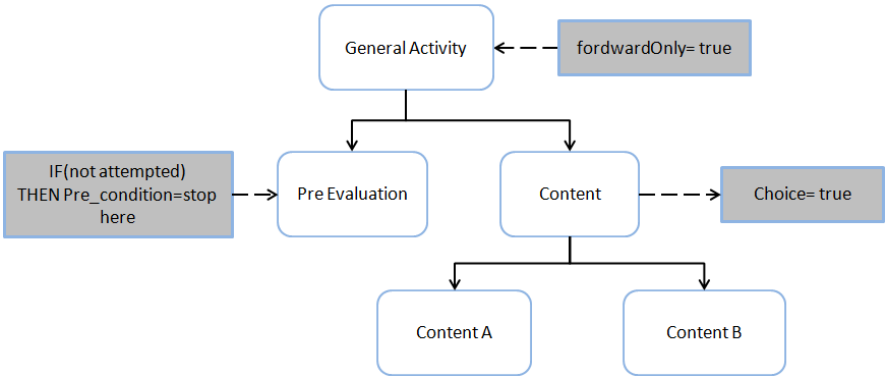


Fig. 3. Basic example of an Activity Tree with their rules and the configuration of the navigation in the tree

The activity content contains two objects which the user choose one, in this case Rollup Rule "Choice" on true states that the activity be satisfied when you meet any of the activities in the container. Although possibly both content learning objects A and B Content review the same issue, you may do so with different media, making them more appropriate to different learning styles. Although the student is free to choose any of the two actions depending on your learning style Instructor may recommend one or another, such Content an activity is recommended for students with theoretical learning style.

Our approach will be similar to the general structure but with some aggregations on the sequence, one of which is the proposed object environmental learning which will be taken by each of the activities where the difference will be that the activity can be composed of one or more environmental learning objects. The rules on every can also become fuzzy as we see in Figure 3 where the level of learning style is determined by fuzzy rules. We can also see how the environmental learning objects are taken of the repository and merging with the sequence in the figure 4.

Then the learning environment will be used by a group of users and devices will have basic input and output as monitors, projectors, tablet PCs, mobile devices and RFID sensors. The teacher will generate a sequence with a theme related to the test users as a programming Class.

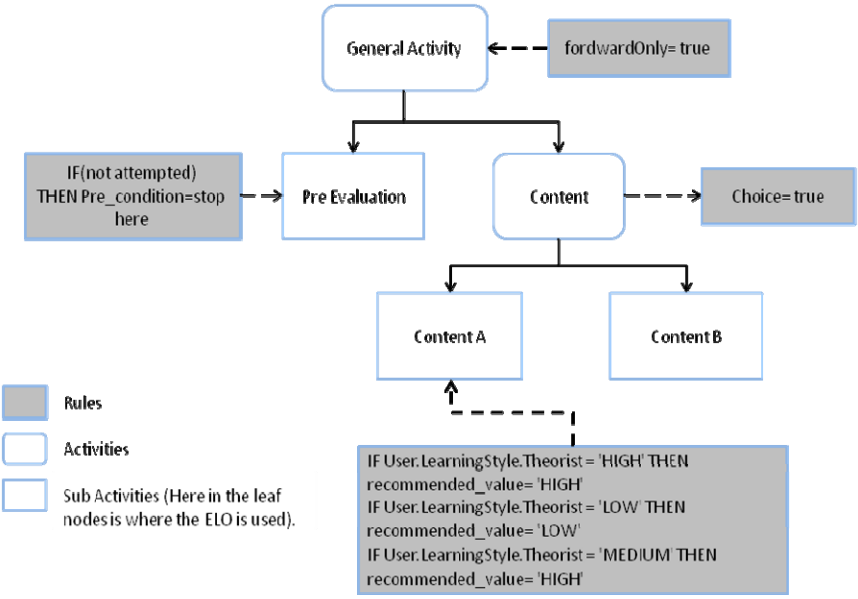


Fig. 4. The proposed methodology with learning objects and the fuzzy rules

4 Conclusions

This paper presents the proposed approach to environmental learning object and a brief overview of this learning object, how is made and how is used in the environment by the sequencing. As future work we expect to implement these learning objects in an environment with real devices and use fuzzy logic to automate some tasks and rules of the sequencing that will start manuals.

References

[1] Dattolo, A., Vicenzo, L.: A Fuzzy approach to Adaptive Hypermedia. In: Proceedings. 4th Int. Conf. on K.B. Intelligent Engineering Systems and Allied Technologies, vol. 2, pp. 700–703 (2000)

[2] Cisco Systems, Reusable information object strategy, http://www.cisco.com/warp/public/779/ibs/solutions/learning/whitepapers/elcisco_rio.pdf (reviewed in May 2012)

[3] Castillo, G., Breda, A.M.: Adaptive Bayes for a Student Modeling Prediction Task Based on Learning Styles. In: Brusilovsky, P., Corbett, A.T., de Rosis, F. (eds.) UM 2003. LNCS(LNAI), vol. 2702, pp. 328–332. Springer, Heidelberg (2003)

[4] Redeker, G.: An Educational Taxonomy for Learning Objects. In: IEEE International Conference on Advanced Learning Technologies, ICALT 2003 (2003)

[5] IMS Global Learning Consortium, I (2003a). Ims simple sequencing best practiceand implementation guide version 1.0 _nal speci_cation. Rep. tec., IMS Global Learning Consortium, Inc. (reviewed in March 2012)

[6] Psotka, J., Mutter, S.A.: Intelligent Tutoring Systems: Lessons Learned. Lawrence Erlbaum Associates (1988) ISBN 0-8058-0192-8

[7] Papanikolau, K., Grigoriadou, M., Kornilakis, H., Magoulas, G.: Personalizing the Interaction in a Web-based Educational Hypermedia System: the case of IN-SPIRE. User Modeling and User-Adapted Interaction 13, 213–267 (2001)

[8] Learning Technology Standards Committee (2002) (PDF), Draft Standard for Learning Object Metadata. IEEE Standard 1484.12.1, Institute of Electrical and Electronics Engineers, New York (reviewed in May 2012)

[9] Hannafin, M., Land, S., Oliver, K.: Open learning environments: Foundations, methods, and models. In: Reigeluth, C. (ed.) Instructional Design Theories and Models, pp. 115–140. Lawrence Erlbaum Associates, Mahwah (1999)

[10] Santally, M., Senteni, A.: A Learning Object Approach to Personalized Web-based Instruction. European Journal of Open Distance and Learning (2005)

[11] Polsani, P.: Use and abuse of reusable learning objects, Learning Technology Center. University of Arizona, USA (2003)

[12] Brusilovsky, P.: Student model centered architecture for intelligent learning environments. International Center for Scientific and Technical Information Kuusinen str. 21b, Moscow 125252, Russia plb@plb.icsti.su

[13] Stathacopoulou, R., Grigoriadou, M., Magoulas, G., Mitropoulo, D.: A Neuro-fuzzy Approach in Student Modeling. In: Brusilovsky, P., Corbett, A.T., de Rosis, F. (eds.) UM 2003. LNCS(LNAI), vol. 2702, pp. 345–368. Springer, Heidelberg (2003)

[14] Beck, R.J.: What Are Learning Objects?, Learning Objects, Center for International Education, University of Wisconsin-Milwaukee (retrieved April 29, 2004)

[15] Denaux, R., Dimitrova, V., Aroyo, L.: Interactive Ontology-Based User Modeling for Personalized Learning Content Management. Computer Science Department, Eindhoven Univ. of Technology, The Netherlands (2004)
[16] Gerard, R.W.: Shaping the Mind: Computers In Education. In: Atkinson, R.C., Wilson, H.A. (eds.) Computer Assisted Instruction: A Book of Readings. Academic Press, New York (1969)
[17] Fischer, S.: Course and Exercise Sequencing Using Metadata in Adaptive Hypermedia Learning Systems. ACM:Journal of Educational Resources in Computing 1 (2001)
[18] García-Barrios, V.M., Modritscher, F., Gutl, C.: Personalisation versus Adaptation? A User-centered Model Approach and its Applications. In: KNOW 2005, pp. 120–127 (2005)

Author Index