Pesquisa Operacional (2013) 33(3): 443-465

Printed version ISSN 0101-7438 / Online version ISSN 1678-5142
www.scielo.br/pope

# A COMPARATIVE ANALYSIS OF THREE METAHEURISTIC METHODS APPLIED TO FUZZY COGNITIVE MAPS LEARNING

Bruno A. Angélico[1*], Márcio Mendonça[1], Taufik Abrão[2] and Lúcia Valéria R. de Arruda[3]

Received February 14, 2012 / Accepted June 22, 2013

**ABSTRACT.** This work analyses the performance of three different population-based metaheuristic approaches applied to Fuzzy cognitive maps (FCM) learning in qualitative control of processes. Fuzzy cognitive maps permit to include the previous specialist knowledge in the control rule. Particularly, Particle Swarm Optimization (PSO), Genetic Algorithm (GA) and an Ant Colony Optimization (ACO) are considered for obtaining appropriate weight matrices for learning the FCM. A statistical convergence analysis within 10000 simulations of each algorithm is presented. In order to validate the proposed approach, two industrial control process problems previously described in the literature are considered in this work.

**Keywords**: Fuzzy Cognitive Maps, Particle Swarm Optimization, Genetic Algorithm, process control, statistical convergence analysis.

## 1 INTRODUCTION

Fuzzy Cognitive Maps are signed and directed graph, in which the involved variables are fuzzy numbers. They were initially proposed by Kosko [1, 2, 3], as an extension of cognitive maps proposed by Axelrod [4]. The graph nodes represent linguistic concepts modeled by fuzzy sets and they are linked with other concepts through fuzzy connections. Each of these connections has a numerical value (weight) taken from a fuzzy set, which models the relationship strength among concepts. In addition, FCM can be considered a type of cognitive neuro-fuzzy network, which is initially developed by the acquisition of expert knowledge, but can be trained by several supervised and/or unsupervised techniques as reviewed in [5].

There are in the literature several models based on fuzzy cognitive maps developed and adapted to a large range of applications, such as political decision [6], medical decision [7, 8], industrial process control [9, 10], artificial life [11], social systems [12], corporative decision, policy

*Corresponding author
[1]Federal University of Technology – Paraná (UTFPR), Cornélio Procópio, Brazil. E-mails: bangelico@utfpr.edu.br; mendonca@utfpr.edu.br
[2]State University of Londrina – Paraná (UEL), Londrina, Brazil. E-mail: taufik@uel.br
[3]Federal University of Technology – Paraná (UTFPR), Curitiba, Brazil. E-mail: lvrarruda@utfpr.edu.br

analysis [13], among others. A good survey of recent applications is given by Papageorgiou e Salmeron in [14].

A FCM construction can be done in the following manner:

- Identification of concepts and its interconnections determining the nature (positive, negative or null) of the causal relationships between concepts.
- Initial data acquisition by the expert opinions and/or by an equation analysis when the mathematical system model is known.
- Submitting data from expert opinions to a fuzzy system which output represents the FCM weights.
- Weight adaptation and optimization of the initially proposed FCM, adjusting its response to the desired output.
- Validation of the adjusted FCM.

However, when the experts are not able to express the causal relationships or they substantially diverge in opinion about it, computer methods for learning FCMs may be necessary.

Particularly, this paper focuses on the FCM learning using three different population based metaheuristics: particle swarm optimization (PSO), genetic algorithm (GA) and a continuous ant colony optimization version based on the $ACO_{\mathbb{R}}$ proposed by [15]. A statistical convergence analysis of these three algorithms is proposed. Two process control problems described in [9] and [10] are considered in this work.

The rest of the paper has the following organization: Section 2 briefly describes the FCM modeling and the control processes. Section 3 considers the PSO, GA and ACO approaching for FCM learning, while Section 4 shows the simulation results. Lastly, Section 5 points out the main conclusions.

## 2 FCM MODELING IN CONTROL PROCESSES

In FCMs, concepts (nodes) are utilized to represent different aspects and behavior of the system. The system dynamics are simulated by the interaction of concepts. The concept $C_i$, $i = 1, 2, \ldots, N$ is characterized by a value $A_i \in [0, 1]$.

Concepts are interconnected concerning the underlying causal relationships amongst factors, characteristics, and components that constitute the system. Each interconnection between two concepts, $C_i$ and $C_j$, has a weight, $W_{i,j}$, which is numerically represented to the strength of the causal relationships between $C_i$ and $C_j$. The sign of $W_{i,j}$ indicates whether the concept $C_i$ causes the concept $C_j$ or vice versa. Hence,

$$\begin{cases} W_{i,j} > 0, & \text{positive causality} \\ W_{i,j} < 0, & \text{negative causality} \\ W_{i,j} = 0, & \text{no relation} \end{cases}$$

The number of concepts and the initial weights of the FCM are determined by human knowledge and experience. The numerical values, $A_i$, of each concept is a transformation of the fuzzy values assigned by the experts. The FCM converges to a steady state (limit cycle) according to the scheme proposed in [3]:

$$A_i\,(k+1) = f\left(A_i\,(k) + \sum_{\substack{j=1 \\ j\neq i}}^{N} W_{ji} A_j\,(k)\right), \tag{1}$$

where $k$ is the interaction index and $f(\cdot)$ is the sigmoid function

$$f(x) = \frac{1}{1+e^{-\lambda x}}, \tag{2}$$

that guarantees the values $A_i \in [0,\ 1]$. $\lambda > 0$ is a parameter that determines its steepness in the area around zero. In this work, $\lambda = 1$.

Applications of FCM are found in many fields of science, such as artificial life [11], social systems [12], modeling and decision make in corporative environments, in rapid access highways [13], in medical field [8], among others. In this paper, two FCM models for process control problems, described in [9] and [10], are considered.

### 2.1 First Process Control (PROC1)

A simple chemical process frequently considered in literature [9, 16, 17], is initially selected for illustrating the need of FCM learning.

Figure 1 represents the process (PROC1) consisting of one tank and three valves that controls the liquid level in the tank. Valves $V_1$ and $V_2$ fill the tank with different liquids. A chemical reaction takes place into the tank producing a new liquid that leaves the recipient by valve $V_3$. A sensor (gauger) measures the specific gravity of the resultant mixture.

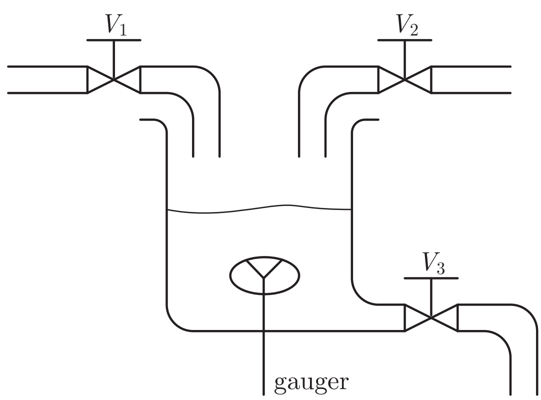



**Figure 1** – PROC1: a chemical process control problem described in [9].

When the value of the specific gravity, $G$, is in the range $[G_{\min}, G_{\max}]$, the desired liquid has been produced. The height of the liquid inside, $H$, must lie in the range $[H_{\min}, H_{\max}]$. The controller has to keep $G$ and $H$ within their bounds, *i.e.*,

$$H_{\min} \leq H \leq H_{\max}; \tag{3}$$

$$G_{\min} \leq G \leq G_{\max}. \tag{4}$$

The group of experts defined a list of five concepts, $C_i$, $i = 1, 2, \ldots, 5$, related to the main physical quantities of the process, [9].

- Concept $C_1$: volume of liquid inside the tank (depends on $V_1$, $V_2$, and, $V_3$);
- Concept $C_2$: state of $V_1$ (closed, open or partially open);
- Concept $C_3$: state of $V_2$ (closed, open or partially open);
- Concept $C_4$: state of $V_3$ (closed, open or partially open);
- Concept $C_5$: specific gravity of the produced mixture.

For this process, the fuzzy cognitive map in Figure 2 can be abstracted [9].

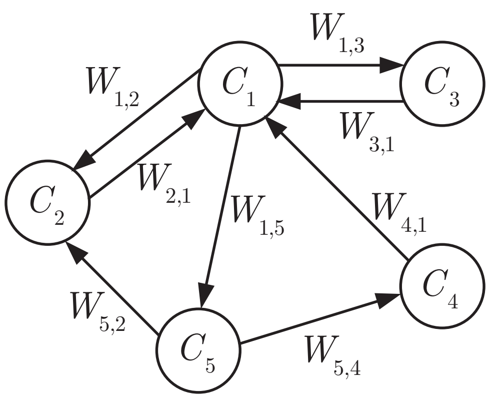



**Figure 2** – Fuzzy Cognitive Map proposed in [9] for the chemical process control problem.

The experts also had a consensus regarding the range of the weights between concepts, as presented in Equations (5) to (12).

$$-0.50 \leq W_{1,2} \leq -0.30; \tag{5}$$

$$-0.40 \leq W_{1,3} \leq -0.20; \tag{6}$$

$$0.20 \leq W_{1,5} \leq 0.40; \tag{7}$$

$$0.30 \leq W_{2,1} \leq 0.40; \tag{8}$$

$$0.40 \leq W_{3,1} \leq 0.50; \tag{9}$$

$$-1.0 \leq W_{4,1} \leq -0.80; \tag{10}$$

$$0.50 \leq W_{5,2} \leq 0.70; \tag{11}$$

$$0.20 \leq W_{5,4} \leq 0.40. \tag{12}$$

For this problem the following weight matrix is obtained:

$$\mathbf{W} = \begin{bmatrix} 0 & W_{1,2} & W_{1,3} & 0 & W_{1,5} \\ W_{2,1} & 0 & 0 & 0 & 0 \\ W_{3,1} & 0 & 0 & 0 & 0 \\ W_{4,1} & 0 & 0 & 0 & 0 \\ 0 & W_{5,2} & 0 & W_{5,4} & 0 \end{bmatrix}. \tag{13}$$

According to [9], all the experts agreed on range of values for $W_{2,1}$, $W_{3,1}$, and $W_{4,1}$, and most of them agreed on the same range for $W_{1,2}$ and $W_{1,3}$. However, regarding the weights $W_{1,5}$, $W_{5,2}$, and $W_{5,4}$, their opinions varied significantly.

Finally, the group of experts determined that the values output concepts, $C_1$ and $C_5$, which are crucial for the system operation, must lie, respectively, in the following regions:

$$0.68 \leq A_1 \leq 0.70; \tag{14}$$

$$0.78 \leq A_5 \leq 0.85. \tag{15}$$

## 2.2 Second Process Control (PROC2)

In [10] a system consisting of two identical tanks with an input and an output valve each one, with the output valve of the first tank being the input valve of the second (PROC2), as illustrated in Figure 3.

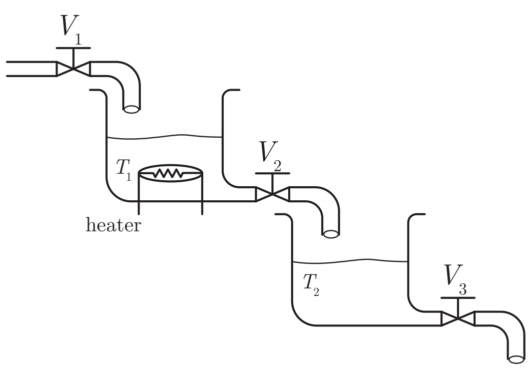



**Figure 3** – PROC2: a chemical process control problem described in [10].

The objective is to control the volume of liquid within the limits determined by the height $H_{\min}$ and $H_{\max}$ and the temperature of the liquid in both tanks within the limits $T_{\min}$ and $T_{\max}$, such that

$$T^1_{\min} \leq T^1 \leq T^1_{\max}; \tag{16}$$

$$T^2_{\min} \leq T^2 \leq T^2_{\max}; \tag{17}$$

$$H^1_{\min} \leq H^1 \leq H^1_{\max}; \tag{18}$$

$$H^2_{\min} \leq H^2 \leq H^2_{\max}. \tag{19}$$

The temperature of the liquid in tank 1 is increased by a heater. A thermostat continuously senses the temperature in tank 1, turning the heater on or off. There is also a temperature sensor in tank 2. When $T_2$ decreases, the valve $V_2$ is open and hot liquid comes into tank 2.

Based on this process, a FCM is constructed with eight concepts:

- Concept $C_1$: volume of liquid inside the tank 1 (depends on $V_1$ and $V_2$);
- Concept $C_2$: volume of liquid inside the tank 2 (depends on $V_1$ and $V_2$);
- Concept $C_3$: state of $V_1$ (closed, open or partially open);
- Concept $C_4$: state of $V_2$ (closed, open or partially open);
- Concept $C_5$: state of $V_3$ (closed, open or partially open);
- Concept $C_6$: Temperature of the liquid in tank 1;
- Concept $C_7$: Temperature of the liquid in tank 2;
- Concept $C_8$: Operation of the heater.

According to [10], the fuzzy cognitive map in Figure 4 can be constructed.

It is assumed in this paper only causal constraints in the weights between concepts, where weights $W_{4,1}$ and $W_{5,2}$ are $\in (-1, 0]$ and the others have positive causality. The weight matrix for PROC2 is given by

$$\mathbf{W} = \begin{bmatrix} 0 & 0 & W_{1,3} & W_{1,4} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & W_{2,4} & W_{2,5} & 0 & 0 & 0 \\ W_{3,1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ W_{4,1} & W_{4,2} & 0 & 0 & 0 & 0 & W_{4,7} & 0 \\ 0 & W_{5,2} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & W_{6,3} & 0 & 0 & 0 & 0 & W_{6,8} \\ 0 & 0 & 0 & W_{7,4} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & W_{8,6} & 0 & 0 \end{bmatrix}. \tag{20}$$

Finally, it is assumed in this paper that the values output concepts, $C_1$, $C_2$, $C_6$ and $C_7$, which are crucial for the system operation, must lie, respectively, in the following regions:

$$0.64 \leq A_1 \leq 0.69; \tag{21}$$

$$0.48 \leq A_2 \leq 0.52; \tag{22}$$

$$0.63 \le A_6 \le 0.67; \tag{23}$$

$$0.63 \le A_7 \le 0.67. \tag{24}$$

Two significant weaknesses of FCMs are its critical dependence on the experts opinions and its potential convergence to undesired states. In order to handle these impairments, learning procedures can be incorporated, increasing the efficiency of FCMs and avoiding convergence to undesired steady states [18].

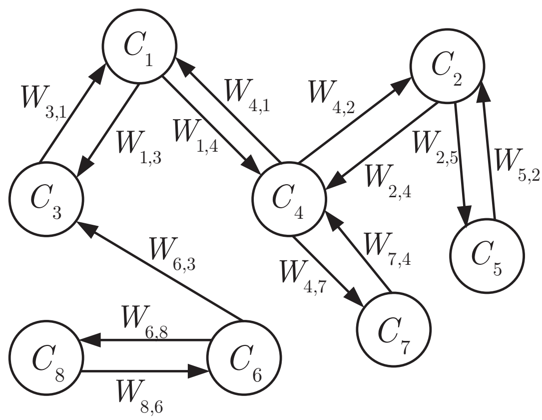



**Figure 4** – Fuzzy Cognitive Map proposed in [10] for the chemical process control problem.

## 3 METAHEURISTIC FCM LEARNING

In [5] a very interesting survey on FCM learning is provided. The FCM weights optimization (FCM learning) can be classified into three different methods.

1. Hebbian learning based algorithm;
2. Metaheuristic optimization techniques, including genetic algorithm, particle swarm optimization, differential evolution, ant colony optimization (these four are population based algorithms), simulated annealing, etc;
3. Hybrid approaches.

In the Hebbian based methodologies, the FCM weights are iteratively adapted based on a law which depends on the concepts behavior [11], [19], requiring the experts' knowledge for initial weight values. Dickerson and Kosko in [11] proposed the Differential Hebbian Learning (DHL) algorithm, which is a classic example . On the other hand, metaheuristic techniques tries to find a proper **W** matrix by minimizing a cost function based on the error among the desired values of the output concepts and the current output concepts' values (27). The experts' knowledge is not totally necessary, except for the causality constraints, due to the physical restrictions[1].

[1] For instance, a valve cannot be negatively open.

These techniques are optimization tools and generally are computationally complex. Examples of hybrid approaching considering Hebbian learning and Metaheuristic optimization techniques can be found in [20], [21].

According to [22], metaheuristic search methods represent methodologies that can be used in designing underlying heuristics to solve specific optimization problems. PSO, GA and ACO are inside a specific class named population-based metaheuristics.

There are several works in the literature dealing with metaheuristic optimization learning. Most of them are population-based algorithms. For instance, in [9] the PSO algorithm with constriction factor is adopted; in [23] it is presented a FCM learning based on a Tabu Search (TS) and GA combination; in [24] a variation of GA named RCGA (real codec-G.A.) is proposed; in [25] a comparison between GA and Simulated Annealing (SA) is done; in [16] the authors presented a GA based algorithm named Extended Great Deluge Algorithm.

The purpose of the learning is to determine the values of the FCM weights that will produce a desired behavior of the system, which are characterized by the $M$ output concept values that lie within desired bounds determined by the experts. Hence, the main goal is to obtain a connection matrix

$$\mathbf{W} = \left[W_{i,j}\right], \quad i, j = 1,\ 2,\ \ldots,\ N, \tag{25}$$

that leads the FCM to a steady state with output concept values within the specified region. Note that, with this notation, and defining $\mathbf{A} = [A_1 \ \cdots \ A_N]^\top$, and $\underline{\mathbf{W}} = \mathbf{W}^\top + \mathbf{I}$, with $\{\cdot\}^\top$ meaning transposition and $\mathbf{I}$ identity matrix, Equation (1) can be compactly written as

$$\mathbf{A}(k+1) = f\left(\underline{\mathbf{W}} \cdot \mathbf{A}(k)\right). \tag{26}$$

After the updating procedure in (26), the following cost function is considered for obtaining the optimum matrix $\mathbf{W}$ [9]:

$$\begin{aligned} F(\mathbf{W}) \quad = \quad & \sum_{i=1}^{M} H\left(\min\left(A^i_{\text{out}}\right) - A^i_{\text{out}}\right)\left|\min\left(A^i_{\text{out}}\right) - A^i_{\text{out}}\right| + \\ & \sum_{i=1}^{M} H\left(A^i_{\text{out}} - \max\left(A^i_{\text{out}}\right)\right)\left|\max\left(A^i_{\text{out}}\right) - A^i_{\text{out}}\right|, \end{aligned} \tag{27}$$

where $H(\cdot)$ is the Heaviside function, and $A^i_{\text{out}}$, $i = 1,\ \ldots,\ M$, represents the value of the $i$th output concept. Figure 5 shows a flowchart of the FCM learning by using metaheuristic search methods.

### 3.1 Particle Swarm Optimization

The PSO principle is based on the movement of a population (swarm) of individuals (particles) randomly distributed in the search space, each one with its own position and velocity. The

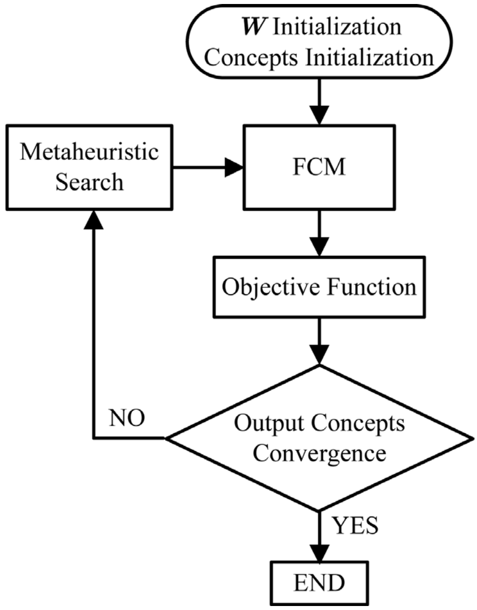



**Figure 5** – Flowchart Fuzzy Cognitive Map learning using metaheuristic search methods.

position of a particle is modified by the application of velocity in order to reach a better performance [26, 27]. In PSO, each particle is treated as a point in a $\mathcal{W}$-dimensional space[2] and represents a vector-candidate. The $i$th particle position at instant $t$ is represented as

$$\mathbf{x}_i(t) = \left[ \; x_{i,1}(t) \quad x_{i,2}(t) \quad \cdots \quad x_{i,\mathcal{W}}(t) \; \right]. \tag{28}$$

In this paper, each $x_{i,1}(t)$ represents one of the $W_{i,j}$ in the $t$th iteration. Each particle retains a memory of the best position it ever encountered. The best position among all particles until the $t$th iteration (best global position) is represented by $\mathbf{x}_g^{best}$, while the best position of the $i$th particle is represented as $\mathbf{x}_i^{best}$. As proposed in [28], the particles are manipulated according to the following equations:

$$\mathbf{v}_i(t+1) = \omega \cdot \mathbf{v}_i(t) + \phi_1 \cdot \mathbf{U}_i^1 \left(\mathbf{x}_g^{\text{best}}(t) - \mathbf{x}_i(t)\right) + \phi_2 \cdot \mathbf{U}_i^2 \left(\mathbf{x}_i^{\text{best}}(t) - \mathbf{x}_i(t)\right) \tag{29}$$

$$\mathbf{x}_i(t+1) = \mathbf{x}_i(t) + \mathbf{v}_i(t+1), \tag{30}$$

where $\phi_1$ and $\phi_2$ are two positive constants representing the individual and global acceleration coefficients, respectively, $\mathbf{U}_i^2$ and $\mathbf{U}_i^2$ are diagonal matrices whose elements are random variables uniformly distributed (u.d.) in the interval [0, 1], and $\omega$ is the inertial weight that plays the role of balancing the global search (higher $\omega$) and the local search (smaller $\omega$).

A typical value for $\phi_1$ and $\phi_2$ is $\phi_1 = \phi_2 = 2$ [27]. Regarding the inertia weight, experimental results suggest that it is preferable to initialize $\omega$ to a large value, and gradually decrease it.

[2] $\mathcal{W}$ is the number of FCM connections (relationships).

The population $\mathcal{P}$ size is kept constant in all iterations. In order to obtain further diversification for the search universe, a factor $V_{\max}$ is added to the PSO model, which is responsible for limiting the velocity in the range $[\pm V_{\max}]$, allowing the algorithm to escape from a possible local solution.

Regarding the FCM, one $i$th vector-candidate, $\mathbf{x}_i$, is represented by a vector formed by $\mathcal{W}$ FCM weights. It is important to point out that after each particle update, restrictions must be imposed on $W_{i,j}$ according to the experts opinion, before the cost function evaluation. Algorithm 1 in Appendix describes the implemented PSO.

### 3.2 Genetic Algorithm

Genetic Algorithm is an optimization and search technique based on selection mechanism and natural evolution, following Darwin's theory of species' evolution, which explains the history of life through the action of physical processes and genetic operators in populations or species. GA allows a population composed of many individuals to evolve under specified selection rules to a state that maximizes the "fitness" (maximizes or minimizes a cost function). Such an algorithm became popular through the work of John Holland in the early 1970s, and particularly his book Adaptation in Natural and Artificial Systems (1975). The algorithm can be implemented in a binary form or in a continuous (real-valued) form. This paper considers the latter case.

Initially, a set of $\mathcal{P}$ chromosomes (individuals) is randomly (uniformly distributed) defined, where each chromosome, $\mathbf{x}$ consists of a vector of variables to be optimized, which, in this case, is formed by FCM weights, respecting the constraints. Each variable is represented by a continuous floating-point number. The $\mathcal{P}$ chromosomes are evaluated through a cost function.

$T$ strongest chromosomes are selected for mating, generating the mating pool, using the roulette wheel method, where the probability of choosing a given chromosome is proportional to its fitness value. In this paper, each pairing generates two offspring with crossover. The weakest $T$ chromosomes are changed by the $T$ offspring from $T/2$ pairing. The crossover procedure is similar to the one presented in [29]. It begins by randomly selecting a variable in the first pair of parents to be the crossover point

$$\alpha = \lceil u \cdot \mathcal{W} \rceil , \tag{31}$$

where $u$ is a random variable (r.v) u.d. in the interval $[0,\ 1]$, and $\lceil \cdot \rceil$ is the upper integer operator. A pair of parents is defined as

$$\begin{aligned} \text{dad}_1 &= \begin{bmatrix} x_{d,1} & x_{d,2} & \cdots & x_{d,\alpha} & \cdots & x_{d,\mathcal{W}} \end{bmatrix} \\ \text{mom}_1 &= \begin{bmatrix} x_{m,1} & x_{m,2} & \cdots & x_{m,\alpha} & \cdots & x_{m,\mathcal{W}} \end{bmatrix}. \end{aligned} \tag{32}$$

Then the selected variables are combined to form new variables that will appear in the offspring

$$\begin{aligned} x_{o,1} &= x_{m,\alpha} - \beta\big[x_{m,\alpha} - x_{d,\alpha}\big]; \\ x_{o,2} &= x_{d,\alpha} + \beta\big[x_{m,\alpha} - x_{d,\alpha}\big]. \end{aligned} \tag{33}$$

where $\beta$ is also a r.v. u.d. in [0, 1]. Finally,

$$\begin{aligned}\text{offspring}_1 &= \left[x^d_{j,1}\ x^d_{j,2}\ x^d_{j,\alpha-1}\ \cdots\ x^{\text{o}}_{j,1}\ x^m_{j,\alpha+1} \cdots\ x^m_{j,\mathcal{W}}\right]\\ \text{offspring}_2 &= \left[x^m_{j,1}\ x^m_{j,2}\ x^m_{j,\alpha-1}\ \cdots\ x^{\text{o}}_{j,2}\ x^d_{j,\alpha+1} \cdots\ x^d_{j,\mathcal{W}}\right].\end{aligned} \tag{34}$$

In order to allow escaping from possible local minima, a mutation operation is introduced in the resultant population, except for the strongest one (elitism). It is assumed in this work a Gaussian mutation. If the probability of mutations is given by $P_m$, there will be $N_m = \lceil P_m \cdot (\mathcal{P}-1) \cdot \mathcal{W} \rceil$ mutations uniformly chosen among $(\mathcal{P}-1)\cdot\mathcal{W}$ variables. If $x_{i,w}$ is chosen, with $w = 1, 2, \ldots, \mathcal{W}$, than, after Gaussian mutation, it is substituted by

$$x'_{i,w} = x_{i,w} + \mathcal{N}\left(0, \sigma_m^2\right), \tag{35}$$

where $\mathcal{N}\left(0, \sigma_m^2\right)$ represents a normal r.v. with zero mean and variance $\sigma_m^2$.

After mutation, restrictions must be imposed on $W_{i,j}$ according to the experts opinion, before the cost function evaluation. Algorithm 2 in Appendix describes the implemented GA.

### 3.3 Ant Colony Optimization

The Ant Colony Optimization (ACO) metaheuristic was developed to solve optimization problems based on the behavior of ants [30]. These individuals are inserted into a highly structured society, which performs tasks sometimes very complex, for example, the search for food. During the search for food, ants release a substance called pheromone, which helps in locating the shortest paths from their nest through the food source [30]. The ACO came from this idea, being originally used for solving combinatorial optimization problems.

The algorithm implemented in this work is a continuous version of ACO based on the $\text{ACO}_{\mathbb{R}}$ proposed in [15], where the memory represented by pheromone deposition is substituted by a solutions file.

Initially, a solutions file is created with $K$ candidate solutions, randomly generated. The number of variables of each candidate solution is equal to $\mathcal{W}$. Figure 6 illustrates the configuration of the solutions file.

The solutions file is sorted according to the quality of each solution evaluated through the cost function (fitness), given by (27). Next, for each vector of the solutions file ($l = 1, \ldots, K$), a weight $\omega_l$ is calculated according to the Gaussian function with unitary mean and standard deviation $\sigma = qK$, such that:

$$\omega_l = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(l-u)^2}{2\sigma^2}} = \frac{1}{qK\sqrt{2\pi}} e^{-\frac{(l-1)^2}{2q^2K^2}} \tag{36}$$

where $q$ is an input parameter of the algorithm. If $q$ is small, the best solutions evaluated will preferably be chosen. On the other hand, if $q$ is high, the choice becomes more uniform [15].

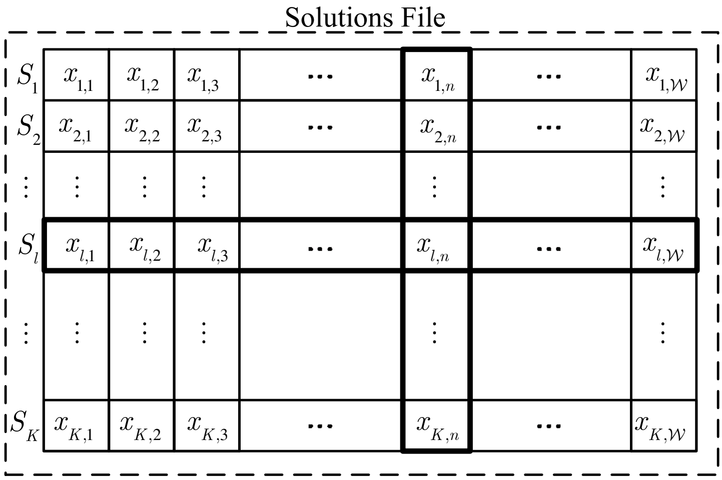



**Figure 6** – ACO solutions file.

A colony with $M$ ants is adopted, which means that $M$ of the $K$ elements are taken from the solution file to update the position. The roulette wheel method was adopted for choosing the ants, based on a cumulative probability vector with probabilities given by:

$$p_l = \frac{\omega_l}{\sum_{r=1}^{K} \omega_r} \tag{37}$$

The next step consists of finding the standard deviation corresponding to each element of a solution vector. Assuming a position (with $l = 1, \ldots, K$, $n = 1, \ldots, \mathcal{W}$) of the solution vector, the standard deviation of this element is calculated as:

$$\sigma_{l,n} = \frac{\zeta}{K-1} \sum_{\substack{j=1 \\ j \neq l}}^{K} \left| x_{j,n} - x_{l,n} \right| \tag{38}$$

where $\zeta > 0$, which is also an input parameter, has an effect similar to the evaporation rate of the pheromone in the classical ACO. The convergence speed of the algorithm increases with $\zeta$ [15]. Thereupon, a Gaussian random variable with zero mean and standard deviation $\sigma_l^i$ is added to a given element of a chosen ant, $x_{l,n}$, so that new directions are generated for each one of the ants. Algorithm 3 in Appendix describes the implemented ACO.

## 4 SIMULATION RESULTS

All simulations were taken considering $10^4$ trials for PROC1 and PROC2. The input parameters of the metaheuristic methods were empirically adjusted. For PSO, the first choices were $\phi_1 = \phi_2 = 2$, $\omega = 1$, $V_{\max} = 0$ and $\mathbf{v}_i(0) = 0$. For GA, $T = 2 \cdot round(\mathcal{P}/4)$, $P_m = 0.1$ and $\sigma_m = 0.2$ were initially chosen. For ACO, $q = 0.1$ and $\zeta = 0.85$ were initially tested. However, after the first tests, these parameters were adjusted as shown in Table 1.

### 4.1 Process PROC1

The adopted values for the input parameters of PSO, GA and ACO are summarized on Table 1. Four different scenarios for the process PROC1 were analyzed. The main performance results for each scenario are described in the next subsections.

**Table 1** – PSO, GA and ACO input parameters values for PROC1.

| PSO | |
|---|---|
| Population | $\mathcal{P} = 10, 20$ |
| Acceleration Coefficients | $\phi_1 = 2, \phi_2 = 0.2$ |
| Inertial Weight | $\omega = 1.2$ |
| Initial Velocity | $\mathbf{v}_i(0) = 0.1$ |
| Maximum Velocity | $V_{\max} = 2$ |
| **GA** | |
| Population | $\mathcal{P} = 10$ |
| Mating Pool | $T = 2 \cdot round(\mathcal{P}/4)$ |
| Rate of Mutation | $P_m = 0.2$ |
| Mutation Std. Deviation | $\sigma_m = 0.3$ |
| **ACO** | |
| Number of Ants | $M = 10, M = 20$ |
| Size of the Solutions File | $K = M + 5$ |
| Parameter $q$ | $q = 0.5$ |
| Parameter $\zeta$ | $\zeta = 1.5$ |

#### 4.1.1 Scenario 1

This scenario considers all the constraints on the FCM weights shown in Equations (5) to (12). As mentioned in [9] and also verified here, there is no solution in this case.

#### 4.1.2 Scenario 2

In this scenario, the constraints on the FCM weights $W_{1,5}$, $W_{5,2}$, and $W_{5,4}$ have been relaxed, since the experts' opinions have varied significantly. In this case the values of these weights were allowed to assume any value in the interval [0, 1] in order to keep the causality of relationships. Tables 2, 3 and 4 present the obtained simulation results for PSO, GA and ACO, respectively.

As can be observed, in the Scenario 2, GA has a better performance than PSO and ACO, achieving convergence without failure with $\mathcal{P} = M = 10$. Both PSO and ACO have presented few errors when $\mathcal{P} = M = 10$, being PSO somewhat better than ACO, and no convergence errors in $10^4$ independent experiments when $\mathcal{P} = M = 20$. Figures 7 and 8 present the mean FCM concepts convergence in the Scenario 2 under $10^4$ trials. Considering the best case simulated for each algorithm (GA with $\mathcal{P} = 10$, PSO with $\mathcal{P} = 20$ and ACO with $M = 20$), PSO presented the fastest average convergence, being ACO with $M = 20$ the second fastest.

**Table 2** – PSO simulation results for Scenario 2, PROC1.

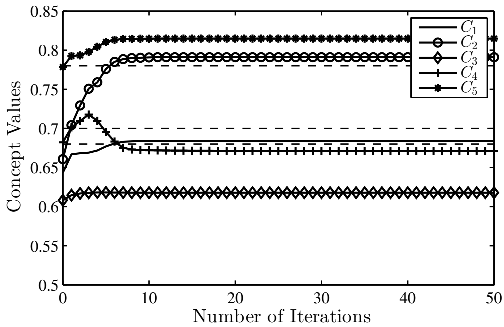
| PSO with $\mathcal{P} = 10$ | | | | | |
|---|---|---|---|---|---|
| $A_{\max}$ | 0.6882 | 0.8058 | 0.6203 | 0.8389 | 0.8186 |
| $A_{\min}$ | 0.6477 | 0.7297 | 0.5749 | 0.6590 | 0.7800 |
| $A_{average}$ | 0.6840 | 0.7911 | 0.6178 | 0.6713 | 0.8148 |
| Number of Failures: 49 | | | | | |
| Probability of Success: 0.9951 | | | | | |
| PSO with $\mathcal{P} = 20$ | | | | | |
| $A_{\max}$ | 0.6882 | 0.8051 | 0.6183 | 0.6967 | 0.8186 |
| $A_{\min}$ | 0.6800 | 0.7297 | 0.5750 | 0.6590 | 0.7801 |
| $A_{average}$ | 0.6838 | 0.7926 | 0.6178 | 0.6730 | 0.8139 |
| Number of Failures: 0 | | | | | |
| Probability of Success: 1.0 | | | | | |

**Table 3** – GA simulation results for Scenario 2, PROC1.

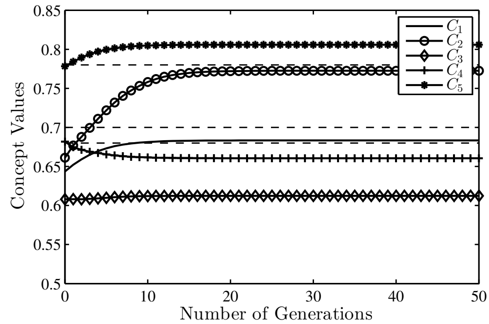
| GA with $\mathcal{P} = 10$ | | | | | |
|---|---|---|---|---|---|
| $A_{\max}$ | 0.6882 | 0.8051 | 0.6183 | 0.6964 | 0.8186 |
| $A_{\min}$ | 0.6800 | 0.7297 | 0.5749 | 0.6590 | 0.7800 |
| $A_{average}$ | 0.6833 | 0.7726 | 0.6123 | 0.6605 | 0.8059 |
| Number of Failures: 0 | | | | | |
| Probability of Success: 1.0 | | | | | |

**Table 4** – ACO simulation results for Scenario 2, PROC1.

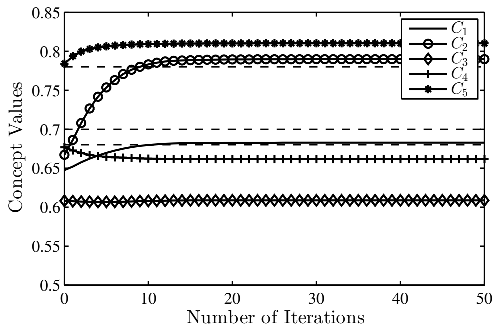
| ACO with $M = 10$ | | | | | |
|---|---|---|---|---|---|
| $A_{\max}$ | 0.6882 | 0.8054 | 0.6192 | 0.6960 | 0.8186 |
| $A_{\min}$ | 0.6659 | 0.6050 | 0.5749 | 0.6590 | 0.7800 |
| $A_{average}$ | 0.6828 | 0.7898 | 0.6088 | 0.6616 | 0.8102 |
| Number of Failures: 61 | | | | | |
| Probability of Success: 0.9939 | | | | | |
| ACO with $M = 20$ | | | | | |
| $A_{\max}$ | 0.6882 | 0.8051 | 0.6183 | 0.6965 | 0.8186 |
| $A_{\min}$ | 0.6800 | 0.7299 | 0.5749 | 0.6590 | 0.7800 |
| $A_{average}$ | 0.6830 | 0.7893 | 0.6088 | 0.6611 | 0.8102 |
| Number of Failures: 0 | | | | | |
| Probability of Success: 1.0 | | | | | |




(a)

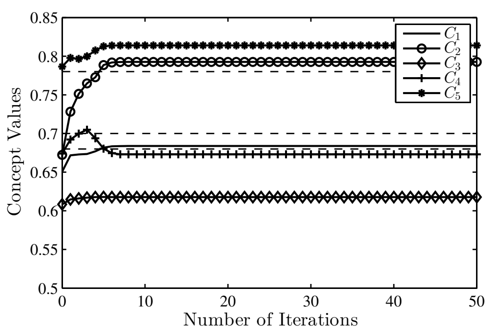



(b)

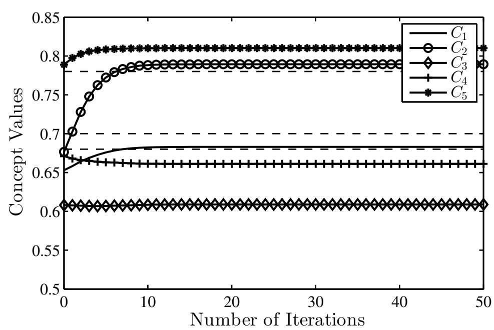



(c)

**Figure 7** – Mean convergence for (a) PSO, (b) GA and (c) ACO in PROC1, Scenario 2 with $\mathcal{P} = M = 10$.




(a)




(b)

**Figure 8** – Mean convergence for (a) PSO and (b) ACO in PROC1, Scenario 2 with $\mathcal{P} = M = 20$.

### 4.1.3 Scenario 3

In this Scenario, all the weights constrains were relaxed, but the causalities were kept, *i.e.*, the value of the weights were fixed in the interval [0, 1] or in the interval [−1, 0), according to the causality determined by the experts. Table 5 presents the obtained results. As can be seen, $\mathcal{P} = M = 10$ was enough for achieving 100% of convergence in all schemes. Figure 9 presents the mean convergence of the concepts in $10^4$ independent experiments. In this scenario, the mean convergence speeds were very similar in all cases.

**Table 5** – GA, PSO and ACO simulation results for Scenario 3, PROC1.

| | | | | | |
|---|---|---|---|---|---|
| PSO with $\mathcal{P} = 10$ | | | | | |
| $A_{\max}$ | 0.7000 | 0.8404 | 0.6590 | 0.8404 | 0.8206 |
| $A_{\min}$ | 0.6800 | 0.4339 | 0.4339 | 0.6590 | 0.7800 |
| $A_{average}$ | 0.6907 | 0.6886 | 0.6112 | 0.7333 | 0.8080 |
| Number of Failures: 0 | | | | | |
| Probability of Success: 1.0 | | | | | |
| GA with $\mathcal{P} = 10$ | | | | | |
| $A_{\max}$ | 0.7000 | 0.8404 | 0.6590 | 0.8404 | 0.8206 |
| $A_{\min}$ | 0.6800 | 0.4339 | 0.4339 | 0.6590 | 0.7800 |
| $A_{average}$ | 0.6906 | 0.6496 | 0.5914 | 0.7139 | 0.8027 |
| Number of Failures: 0 | | | | | |
| Probability of Success: 1.0 | | | | | |
| ACO with $M = 10$ | | | | | |
| $A_{\max}$ | 0.7000 | 0.8404 | 0.6590 | 0.8404 | 0.8206 |
| $A_{\min}$ | 0.6800 | 0.4339 | 0.4339 | 0.6590 | 0.7800 |
| $A_{average}$ | 0.6901 | 0.6502 | 0.5821 | 0.7202 | 0.8095 |
| Number of Failures: 0 | | | | | |
| Probability of Success: 1.0 | | | | | |

#### 4.1.4 Scenario 4

In this situation, the causality and the strength of the causality were totally relaxed. The algorithms were able to determine proper weights for the connection matrix with $\mathcal{P} = 10$. Table 6 presents the obtained statistical results, while Figure 10 presents the mean convergence of the concepts in $10^4$ independent experiments. As in Scenario 3, the three schemes presented similar mean convergence speed.

### 4.2 Process PROC2

For this process the value of the weights were fixed in the interval $[0,\ 1]$ or in the interval $[-1,\ 0)$, according to the causality determined by the experts. The adopted input simulation parameters for PSO, GA and ACO algorithms were the same considered in PROC1, except for the number of ants in ACO that was enough for $M = 10$.

Table 7 summarizes the simulation results, while Figure 11 presents the mean convergence of the concepts value considering $10^4$ independent experiments. As can be seen, $\mathcal{P} = 10$ was enough for achieving convergence rate of 100% in GA and in ACO. With $\mathcal{P} = 10$, PSO presented

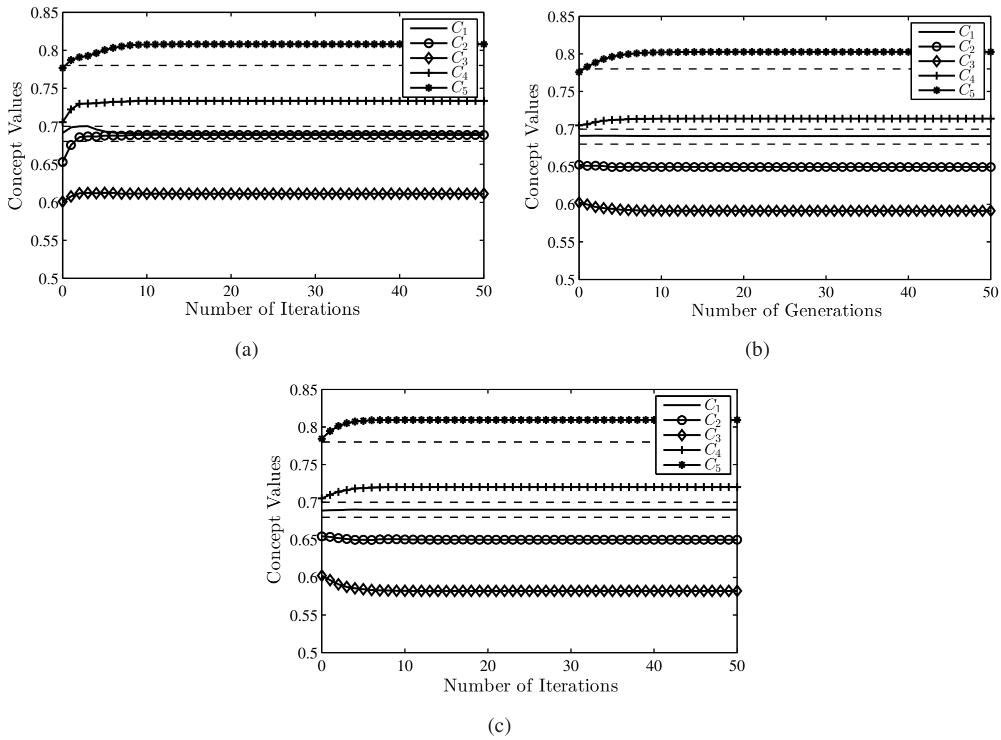



**Figure 9** – Mean convergence for Scenario 3 in PROC1 considering (a) PSO, (b) GA and (c) ACO with $\mathcal{P} = M = 10$.

**Table 6** – GA, PSO and ACO simulation results for Scenario 4, PROC1.

| | | | | | |
|---|---|---|---|---|---|
| PSO with $\mathcal{P} = 10$ | | | | | |
| $A_{\max}$ | 0.7000 | 0.9199 | 0.8206 | 0.8404 | 0.8206 |
| $A_{\min}$ | 0.6800 | 0.2129 | 0.4339 | 0.3952 | 0.7800 |
| $A_{average}$ | 0.6901 | 0.6908 | 0.6767 | 0.6777 | 0.8101 |
| Number of Failures: 0 | | | | | |
| Probability of Success: 1.0 | | | | | |
| GA with $\mathcal{P} = 10$ | | | | | |
| $A_{\max}$ | 0.7000 | 0.9192 | 0.8206 | 0.8404 | 0.8206 |
| $A_{\min}$ | 0.6800 | 0.2130 | 0.4339 | 0.3952 | 0.7800 |
| $A_{average}$ | 0.6905 | 0.6154 | 0.6371 | 0.6306 | 0.8030 |
| Number of Failures: 0 | | | | | |
| Probability of Success: 1.0 | | | | | |
| ACO with $M = 10$ | | | | | |
| $A_{\max}$ | 0.7000 | 0.9199 | 0.8206 | 0.8404 | 0.8207 |
| $A_{\min}$ | 0.6800 | 0.2129 | 0.4338 | 0.3952 | 0.7800 |
| $A_{average}$ | 0.6902 | 0.5880 | 0.6215 | 0.6120 | 0.8113 |
| Number of Failures: 0 | | | | | |
| Probability of Success: 1.0 | | | | | |

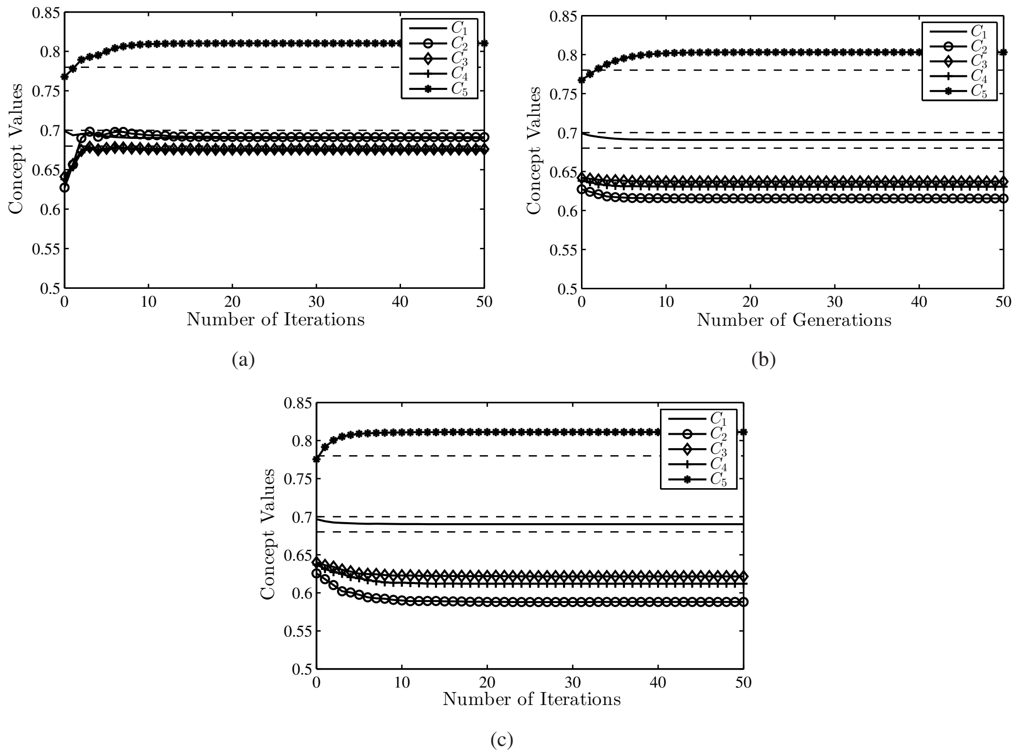



**Figure 10** – Mean convergence for Scenario 4 in PROC1 considering (a) PSO, (b) GA and (c) ACO with $\mathcal{P} = M = 10$.

30 failures and probability of success equal to 0.9970. However, with $\mathcal{P} = 20$, PSO was able to achieve 100% of success. For this process, PSO has a faster average convergence than GA and ACO.

Table 8 shows the mean execution time in 50 iterations for each algorithm in all considered configuration. A computer with CPU Intel Core i7-2675QM @ 2.20GHz and 6GB of RAM was considered. As can be seen, the way the algorithms were implemented, PSO has a execution time shorter than GA and ACO, being ACO the slowest scheme.

## 5 CONCLUSIONS

Fuzzy cognitive maps have being considered with great interest in a large range of applications. Learning methods may be necessary for obtaining proper values for the weight matrix and reducing the dependence on the experts' knowledge.

Within this context, this paper presented a comparison of three heuristic search approaches, PSO, GA and ACO, applied to FCM weight optimization in two processes control. In all the considered cases, GA presented a performance in terms of probability of success better or equal

**Table 7** – GA, PSO and ACO simulation results for PROC2.

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| PSO with $\mathcal{P} = 10$ | | | | | | | | |
| $A_{\max}$ | 0.7040 | 0.6590 | 0.9029 | 0.9407 | 0.8134 | 0.7234 | 0.6700 | 0.8153 |
| $A_{\min}$ | 0.6400 | 0.4130 | 0.6590 | 0.6590 | 0.6590 | 0.6590 | 0.6590 | 0.6590 |
| $A_{average}$ | 0.6592 | 0.5006 | 0.7079 | 0.7221 | 0.6809 | 0.6593 | 0.6592 | 0.6850 |
| Number of Failures: 30 | | | | | | | | |
| Probability of Success: 0.9970 | | | | | | | | |
| PSO with $\mathcal{P} = 20$ | | | | | | | | |
| $A_{\max}$ | 0.6800 | 0.5200 | 0.9040 | 0.9411 | 0.7869 | 0.6700 | 0.6700 | 0.8154 |
| $A_{\min}$ | 0.6400 | 0.4800 | 0.6590 | 0.6590 | 0.6590 | 0.6590 | 0.6590 | 0.6590 |
| $A_{average}$ | 0.6594 | 0.5002 | 0.7116 | 0.7267 | 0.6813 | 0.6594 | 0.6593 | 0.6865 |
| Number of Failures: 0 | | | | | | | | |
| Probability of Success: 1.0 | | | | | | | | |
| GA with $\mathcal{P} = 10$ | | | | | | | | |
| $A_{\max}$ | 0.6800 | 0.5200 | 0.9038 | 0.9409 | 0.7870 | 0.6700 | 0.6700 | 0.8153 |
| $A_{\min}$ | 0.6400 | 0.4800 | 0.6590 | 0.6590 | 0.6590 | 0.6590 | 0.6590 | 0.6590 |
| $A_{average}$ | 0.6596 | 0.5014 | 0.7500 | 0.7717 | 0.7055 | 0.6596 | 0.6596 | 0.7082 |
| Number of Failures: 0 | | | | | | | | |
| Probability of Success: 1.0 | | | | | | | | |
| ACO with $M = 10$ | | | | | | | | |
| $A_{\max}$ | 0.6800 | 0.5200 | 0.9049 | 0.9415 | 0.7870 | 0.6700 | 0.6700 | 0.8154 |
| $A_{\min}$ | 0.6400 | 0.4800 | 0.6590 | 0.6590 | 0.6590 | 0.6590 | 0.6590 | 0.6590 |
| $A_{average}$ | 0.6595 | 0.4997 | 0.7674 | 0.7938 | 0.7045 | 0.6604 | 0.6604 | 0.7170 |
| Number of Failures: 0 | | | | | | | | |
| Probability of Success: 1.0 | | | | | | | | |

**Table 8** – Execution time.

| Process | Algorithm | Population | Time (ms) |
|---|---|---|---|
| PROC. 1 | PSO | $\mathcal{P} = 10$ | 16.6 |
| PROC. 1 | PSO | $\mathcal{P} = 20$ | 30.4 |
| PROC. 1 | GA | $\mathcal{P} = 10$ | 21.6 |
| PROC. 1 | ACO | $M = 10$, $K = 15$ | 45.4 |
| PROC. 1 | ACO | $M = 20$, $K = 25$ | 94.9 |
| PROC. 2 | PSO | $\mathcal{P} = 10$ | 17.2 |
| PROC. 2 | PSO | $\mathcal{P} = 20$ | 33.0 |
| PROC. 2 | GA | $\mathcal{P} = 10$ | 22.9 |
| PROC. 2 | ACO | $M = 10$, $K = 15$ | 52.8 |

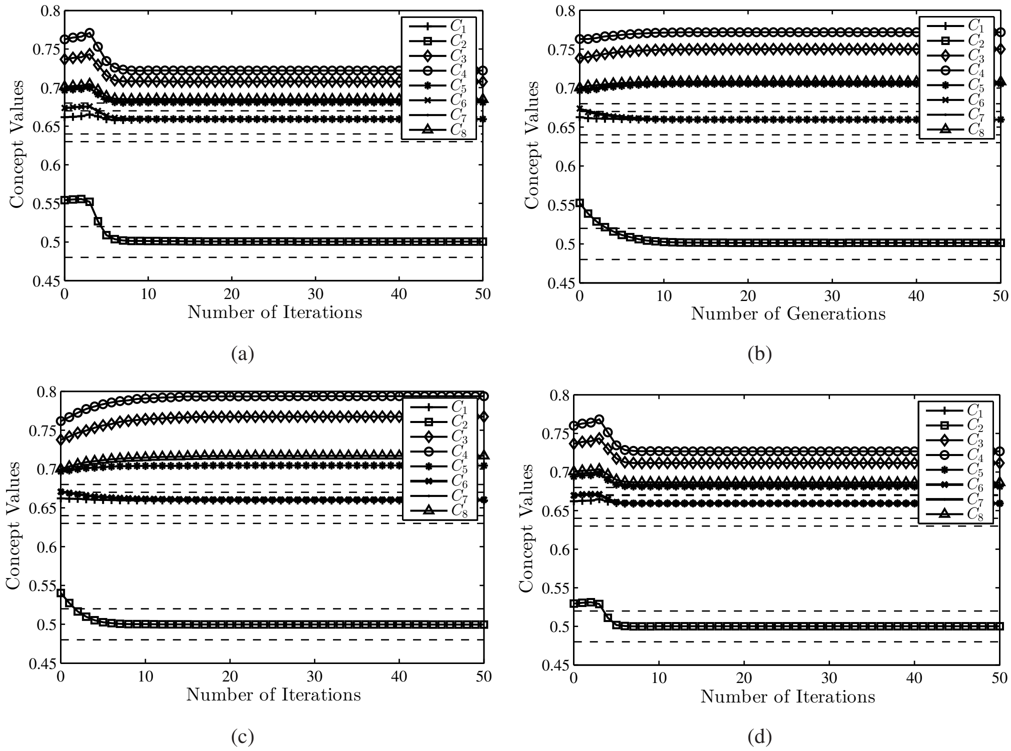



**Figure 11** – Mean convergence in PROC2 for (a) PSO with $\mathcal{P} = 10$, (b) GA with $\mathcal{P} = 10$, (c) ACO with $M = 10$ and (d) PSO with $\mathcal{P} = 20$.

to the other two schemes, being ACO the second best technique in terms of probability of success in the two considered processes (somewhat better than PSO).

Specially in scenario 2 of PROC 1, when there are several weight constraints, GA achieved 100% of success in $10^4$ independent experiments with a population of 10 chromosomes. PSO and ACO needed $\mathcal{P} = 20$ particles and $M = 20$ ants in the population in order to reach 100% of success. In PROC2, both GA and ACO needed $\mathcal{P} = M = 10$, while PSO presented only 30 failures in $10^4$ independent experiments when $\mathcal{P} = 10$, and no errors when $\mathcal{P} = 20$.

In terms of execution time in 50 iterations and considering the input parameters adopted, ACO presented the slowest time, being PSO the fastest algorithm. It is important to emphasize that no code optimization techniques have been considered while implementing PSO, GA and ACO.

As future work, other optimization heuristics can be considered to have a broader comparison, as Simulated Annealing (SA), Tabu Search, among others, as well as techniques based on Hebian learning. Other more complex applications of FCM in different areas of research can also be considered.

## REFERENCES


[1] KOSKO B. 1986. Fuzzy cognitive maps. *International Journal of Man-Machine Studies*, **24**: 65–75.

[2] KOSKO B. 1992. Neural networks and fuzzy systems: a dynamical systems approach to machine intelligence. Prentice Hall, Upper Saddle River, NJ.

[3] KOSKO B. 1997. Fuzzy Engineering. Prentice-Hall, Inc., Upper Saddle River, NJ.

[4] AXELROD RM. 1976. Structure of Decision: The Cognitive Maps of Political Elites. Princeton University Press, Princeton, NJ.

[5] PAPAGEORGIOU EI. 2012. Learning algorithms for fuzzy cognitive maps – a review study. *IEEE Transactions on Systems, Man, and Cybernetics* – Part C, **42**(2): 150–163.

[6] ANDREOU AS, MATEOU NH & ZOMBANAKIS GA. 2005. Soft computing for crisis management and political decision making: the use of genetically evolved fuzzy cognitive maps. *Soft Computing*, **9**: 194–210.

[7] STYLIOS CD, GEORGOPOULOS VC, MALANDRAKI GA & CHOULIARA S. 2008. Fuzzy cognitive map architectures for medical decision support systems. *Applied Soft Computing*, **8**: 1243–1251.

[8] PAPAGEORGIOU EI, STYLIOS CD & GROUMPOS PP. 2003. An integrated two-level hierarchical system for decision making in radiation therapy based on fuzzy cognitive maps. *IEEE Transactions on Biomedical Engineering*, **50**(12): 1326–1339.

[9] PAPAGEORGIOU EI, PARSOPOULOS KE, STYLIOS CS, GROUMPOS PP & VRAHATIS MN. 2005. Fuzzy cognitive maps learning using particle swarm optimization. *Journal of Intelligent Information Systems*, **25**: 95–121.

[10] STYLIOS CD, GEORGOPOULOS VC & GROUMPOS PP. 1997. The use of fuzzy cognitive maps in modeling systems. In: Proceeding of 5$^{th}$ IEEEMediterranean Conference on Control and Systems, Paphos.

[11] DICKERSON JA & KOSKO B. 1993. Virtual worlds as fuzzy cognitive maps. In: *IEEE Virtual Reality Annual International Symposium*, **3**: 471–477.

[12] TABER R. 1991. Knowledge Processing With Fuzzy Cognitive Maps. *Expert Systems With Applications*, **1**(2): 83–87.

[13] PERUSICH K. 1996. Fuzzy cognitive maps for policy analysis. In: International Symposium on Technology and Society Technical Expertise and Public Decisions, pages 369–373.

[14] PAPAGEORGIOU EI & SALMERON JL. 2013. A review of fuzzy cognitive maps research during the last decade. *IEEE Transactions on Fuzzy Systems*, **21**(1): 66–79.

[15] SOCHA K & DORIGO M. 2008. Ant colony optimization for continuous domains. *European Journal of Operational Research*, **185**(3): 1155–1173.

[16] BAYKASOGLU A, DURMUSOGLU ZDU & VAHIT K. 2011. Training fuzzy cognitive maps via extended great deluge algorithm with applications. *Computers in Industry*, **62**(2): 187–195.

[17] GLYKAS M. 2010. Fuzzy Cognitive Maps: Advances in Theory, Methodologies, Tools and Applications. 1$^{st}$ edition, Springer Publishing Company, Incorporated.

[18] PARSOPOULOS KE & GROUMPOS PP. 2003. A first study of fuzzy cognitive maps learning using particle swarm optimization. In: IEEE 2003 Congress on Evolutionary Computation, pages 1440–1447.

[19] STYLIOS CD, PAPAGEORGIOU EI & GROUMPOS PP. 2004. Active hebbian learning algorithm to train fuzzy cognitive maps. *International Journal of Approximate Reasoning*, **37**(3): 219–249.

[20] PAPAGEORGIOU EI & GROUMPOS PP. 2005. A new hybrid learning algorithm for fuzzy cognitive maps learning. *Applied Soft Computing*, **5**: 409–431.

[21] ZHU Y & ZHANG W. 2008. An integrated framework for learning fuzzy cognitive map using rcga and nhl algorithm. In: Int. Conf. Wireless Commun., Netw. Mobile Comput., Dalian, China.

[22] TALBI E-G. 2009. Metaheuristics – From Design to Implementation. 1st edition, Wiley, Hoboken, NJ, USA.

[23] ALIZADEH S, GHAZANFARI M, JAFARI M & HOOSHMAND S. 2007. Learning fcm by tabu search. *International Journal of Computer Science*, **3**: 142–149.

[24] STACH W, KURGAN LA, PEDRYCZ W & REFORMAT M. 2005. Genetic learning of fuzzy cognitive maps. *Fuzzy Sets and Systems*, **153**(3): 371–401.

[25] GHAZANFARI M, ALIZADEH S, FATHIAN M & KOULOURIOTIS DE. 2007. Comparing simulated annealing and genetic algorithm in learning fcm. *Applied Mathematics and Computation*, **192**(1): 56–68.

[26] KENNEDY J & EBERHART R. 1995. Particle swarm optimization. In: IEEE International Conference on Neural Networks, pages 1942–1948.

[27] KENNEDY J & EBERHART RC. 2001. Swarm Intelligence. 1st edition, Morgan Kaufmann.

[28] SHI Y & EBERHART RC. 1998. A modified particle swarm optimizer. In: IEEE International Conference on Evolutionary Computation, pages 69–73.

[29] HAUPT RL & HAUPT SE. 2004. Practical Genetic Algorithms. 2nd edition, John Wiley & Sons, Hoboken, NJ, USA.

[30] DORIGO M & ST¨UTZLE T. 2004. Ant Colony Optimization. 1st edition, Bradford Book, Hoboken, NJ, USA.


## APPENDIX

**Algorithm 1** PSO FCM LEARNING

**Input**: $\mathcal{P}$, $\mathcal{G}$, $\omega$, $V_{\max}$, $\mathbf{v}_i(0)$, $\phi_1$, $\phi_2$, $V_{\max}$, $\mathbf{W}_{\min}$, $\mathbf{W}_{\max}$, $\mathbf{A}_{\min}$, $\mathbf{A}_{\max}$
**Output**: $\mathbf{W}_{opt}$
Initialize first population (randomly generated) of weights $\in [\mathbf{W}_{\min}, \mathbf{W}_{\max}]$;
Initial Evaluation through cost function (Equation (27));
Find best global and best individual positions (initialy, $\mathbf{x}_g^{best} = \mathbf{x}_i^{best}$);
**while** ($t \leq G$ or output concepts convergence):
    a. update velocity $\mathbf{v}_i(t+1)$, $i = 1, \ldots, \mathcal{P}$, through (29);
    b. velocity limitation by $V_{\max}$;
    c. update position $\mathbf{x}_i(t+1)$, $i = 1, \ldots, \mathcal{P}$, through (30);
    d. $\mathbf{x}$ (*i.e.*, $\mathbf{W}$) limitation $\in [\mathbf{W}_{\min}; \mathbf{W}_{\max}]$;
    e. Evaluation through cost function (Eq. (27));
    f. Find best global ($\mathbf{x}_g^{best}$) and best individual positions ($\mathbf{x}_i^{best}$);
**end**

$\mathbf{W}_{\min}, \mathbf{W}_{\max}$: matrices of weights (particle positions) constraints;
$\mathbf{A}_{\min}, \mathbf{A}_{\max}$: matrices of concepts constraints;
$\mathbf{W}_{opt}$: optimum weight matrix.

**Algorithm 2** GA FCM LEARNING

**Input**: $\mathcal{P}$, $\mathcal{G}$, $T$, $P_m$, $\sigma_m$, $\mathbf{W}_{\min}$, $\mathbf{W}_{\max}$, $\mathbf{A}_{\min}$, $\mathbf{A}_{\max}$
**Output**: $\mathbf{W}_{opt}$
Initialize first population (randomly generated) of weights $\in [\mathbf{W}_{\min}, \mathbf{W}_{\max}]$;
Initial Evaluation through cost function (Eq. (27));
**while** ($t \leq G$ or output concepts convergence):
  a. Selection of $T$ individuals: roulette wheel for selecting chromosomes;
  b. Pairing and crossover (Eq. (31) to (34));
  c. Gaussian Mutation (Eq. 35);
  d. $\mathbf{x}$ (*i.e.*, $\mathbf{W}$) limitation $\in [\mathbf{W}_{\min}; \mathbf{W}_{\max}]$;
  d. Evaluation through cost function (Eq. (27));
  e. Sort $\mathbf{W}$ and update best result;
**end**

$\mathbf{W}_{\min}$, $\mathbf{W}_{\max}$: matrices of weights (particle positions) constraints;
$\mathbf{A}_{\min}$, $\mathbf{A}_{\max}$: matrices of concepts constraints;
$\mathbf{W}_{opt}$: optimum weight matrix.

**Algorithm 3** ACO FCM LEARNING

**Input**: $M$, $K$, $q$, $\zeta$, $\mathcal{G}$, $\mathbf{W}_{\min}$, $\mathbf{W}_{\max}$, $\mathbf{A}_{\min}$, $\mathbf{A}_{\max}$
**Output**: $\mathbf{W}_{opt}$
Initialize solutions file (randomly generated) of weights $\in [\mathbf{W}_{\min}, \mathbf{W}_{\max}]$;
Initial Evaluation through cost function (Eq. (27));
Sort solutions file according to the fitness;

Calculate $\omega_l$ (Eq. (36)) and $p_l$ (Eq. (37));
**while** ($t \leq G$ or output concepts convergence):
  a. Selection of $M$ ants: roulette wheel according to $p_l$;
  b. Calculate $\sigma_{l,n}$ (Eq. (38));
  c. Update ants position ($x_{l,n}$) adding a $\mathcal{N}(0, \sigma_{l,n})$ variable to the previous position;
  d. $\mathbf{x}$ (*i.e.*, $\mathbf{W}$) limitation $\in [\mathbf{W}_{\min}; \mathbf{W}_{\max}]$;
  d. Evaluation through cost function (Equation (27));
  e. Sort $\mathbf{W}$ and update best result;
**end**

$\mathbf{W}_{\min}$, $\mathbf{W}_{\max}$: matrices of weights (particle positions) constraints;
$\mathbf{A}_{\min}$, $\mathbf{A}_{\max}$: matrices of concepts constraints;
$\mathbf{W}_{opt}$: optimum weight matrix.