Proceedings of the 2007 IEEE Swarm Intelligence Symposium (SIS 2007)

Enhanced Learning in Fuzzy Simulation Models Using Memetic Particle Swarm Optimization

Y.G. Petalas[1,3], K.E. Parsopoulos[1,3], E.I. Papageorgiou[2,3], P.P. Groumpos[2,3], M.N. Vrahatis[1,3]
[1] Computational Intelligence Laboratory (CI Lab), Department of Mathematics,
University of Patras, GR–26110 Patras, Greece. {petalas,kostasp,vrahatis}@math.upatras.gr
[2] Department of Electrical and Computer Engineering, University of Patras,
GR–26500 Patras, Greece. {epapageo, groumpos}@ee.upatras.gr
[3] University of Patras Artificial Intelligence Research Center (UPAIRC),
University of Patras, GR–26110 Patras, Greece.

Abstract— **Fuzzy Cognitive Maps constitute an important simulation methodology that combines neural networks and fuzzy logic. The Fuzzy Cognitive Maps designed by the experts can be enhanced significantly through learning algorithms, which proved to increase their efficiency and accuracy of simulation. Recently, learning algorithms that employ Particle Swarm Optimization for the minimization of properly defined objective functions have been introduced. In this work, we enhance these learning schemes by incorporating local search in PSO, resulting in a Memetic Particle Swarm Optimization learning algorithm. Three variants of the memetic algorithm are applied successfully for the optimization of an Ecological Industrial Park simulation system and they are compared also with the established Particle Swarm Optimization learning schemes. Results are reported and discussed, deriving useful conclusions.**

I. INTRODUCTION

Simulation has proved to be very useful in understanding and studying both natural and artificial systems. The increasing power of modern computers rendered modeling of high–complexity systems viable, providing better insight into their functioning. Also, simulation has been successfully used to show the eventual real effects of alternative conditions and courses of action, contributing towards the performance optimization of a system [1]. Key issues in simulation are the acquisition of information about the simulated system, the determination of its main features and their interconnections, as well as the accuracy of the simulation outcomes.

Depending on the application at hand, there are different types of simulation tools. Fuzzy Cognitive Maps (FCMs) are fuzzy modeling and simulation tools that resemble neural networks. The flexibility and learning properties that stem from the fuzzy and neural representation of FCMs has rendered them an attractive tool for modeling complex systems and supporting decision tasks. Thus, after their introduction by Kosko in 1986 [2], FCMs have been used in a plethora of applications in diverse scientific and technological fields [3]–[11].

FCMs are designed by a group of experts that have deep knowledge of the problem at hand. Experts identify the key concepts of a system and represent them as nodes, while their interconnections are represented as connecting edges, i.e., FCMs represent the system as a directed graph. The strengths of the causal relationships among concepts are represented by weights on the edges. The values of the weights are important for the proper simulation of the system. In the initial design of the FCM, these values are determined through a fuzzification–defuzzification procedure that harnesses the opinions of all experts. However, in some cases, experts may be wrong in their estimations or have widely varied opinions [12]. In such cases, the initial weights of the FCM may not simulate the system properly. In such cases, the performance of FCMs can be significantly enhanced through *learning procedures*. These procedures modify further the weights of the FCM in order to optimize a specific criterion that is usually problem–dependent.

There is a small number of learning procedures proposed in the literature, borrowing ideas from neural networks training [2], [13] and evolutionary computation [12], [14], [15]. Particle Swarm Optimization (PSO) has proved to be very efficient for FCMs learning in different applications [11], [12], [16]–[18]. Recently, a Memetic PSO (MPSO) has been introduced for FCMs learning with promising results on a control problem [19].

In this work, we enhance the MPSO learning scheme and apply it for the optimization of an Ecological Industrial Park simulation system with respect to its pollution output concept. The rest of the paper is organized as follows: Section II provides a short description of FCMs, while Section III is devoted to the description of the memetic learning scheme and its components. The problem under investigation is described in Section IV. Experimental results are reported and discussed in Section V, and the paper concludes with Section VI.

II. FUZZY COGNITIVE MAPS

FCMs were introduced as fuzzy extensions of the Cognitive Maps proposed by Axelrod [20], for knowledge representation and modeling of human decision–making processes. Like their predecessors, traditional concept maps, FCMs consist of nodes that represent key concepts of the system, as well as signed links ("+" or "−") that represent the nature of the relationship among nodes. Weights on the links can further express the magnitude of the effect among the concepts. However, FCMs differ from traditional concept maps in the nature of their

1-4244-0708-7/07/$20.00 ©2007 IEEE

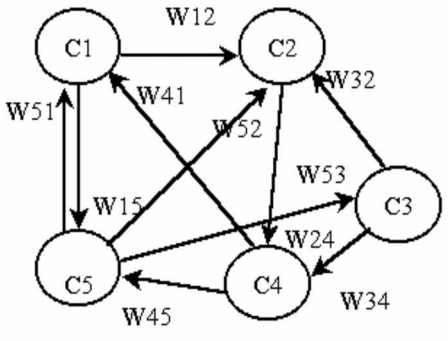

Fig. 1. A simple FCM with 5 concepts and 10 weights.

representation, which, for the FCMs, is causal as well as temporal. The causality modeling over time facilitates the exploration of the implications of complex conceptual models and offers a greater representation flexibility. Also, fuzzy logic allows the representation of fuzzy concepts and degree of causality, while feedback allows the user to explore the hidden properties of the map.

Let N be the number of concepts, C_i, $i = 1, 2, \ldots, N$, of an FCM. Each concept, C_i, has a value, A_i, $i = 1, 2, \ldots, N$, which lies in the range $[0, 1]$. An edge with direction from C_i to a different concept, C_j, is characterized by a weight, w_{ij}, $j = 1, 2, \ldots, N$, that lies in the range $[-1, 1]$. A positive weight, $w_{ij} > 0$, denotes a positive causality, which implies that an increment in concept C_i results in an increment in concept C_j, while a decrement in C_i results also in a decrement in concept C_j. On the other hand, a negative weight, $w_{ij} < 0$, implies a negative causality between the two concepts, i.e., an increment in C_i results in a decrement in C_j and vice versa.

Therefore, the FCM can be described merely by the matrix of its weights,

$$\mathbf{W} = \begin{pmatrix} w_{11} & w_{12} & \cdots & w_{1N} \\ w_{21} & w_{22} & \cdots & w_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ w_{N1} & w_{N2} & \cdots & w_{NN} \end{pmatrix},$$

and the values of its concepts. The absence of interconnection between two concepts can be represented by setting the corresponding weight equal to zero. A simple FCM with 5 concepts and 10 weights is depicted in Fig. 1.

The number of concepts, edges (along with their directions), weights, and their initial values, are provided by the group of experts that designs the FCM [6], [9]. In order to avoid the direct assignment of numerical values by the experts, which fosters the danger of introducing numerical errors, a procedure of fuzzification/defuzzification is used. Thus, experts can express their opinions using a linguistic notion that is converted into fuzzy functions, allowing the developer to capture more fine grain information about the representation [12]. The experts can also pose bounds on the values of both weights and concepts in order to ensure that their physical meaning is preserved. Special attention is usually given to the *output concepts*, i.e., concepts that correspond to crucial features of the system and constitute the core of the simulation investigation.

After the assignment of initial values to the concepts and weights, the FCM operates similarly to a discrete dynamical system, and it is let to converge. This is performed by applying the following rule [21], [22]:

$$A_i(t+1) = f\left(A_i(t) + \sum_{\substack{k=1 \\ k \neq i}}^{N} w_{ki}(t) A_k(t)\right), \qquad (1)$$

where t denotes time; $A_i(t+1)$ is the value of concept C_i at time $t+1$; $A_k(t)$ is the value of concept C_k at time t, and f is usually a sigmoid function. All concepts are updated synchronously at each time step, until their values are not modified any further. This is called a *stable state* of the FCM, and it is assumed that it can simulate the system accurately, i.e., the output concepts lie in specific bounds that are crucial for the desirable operation of the system.

In complex applications, the opinions among of the experts can be significantly different or some estimations can be wrong, thereby hindering the designed FCM from achieving a desirable stable state. In such cases, a proper revision of the weights may be necessary to achieve acceptable output concept values. This can be achieved through learning algorithms, which are means to increase the efficiency and robustness of FCMs, by selecting and modifying the FCM's weights.

In the literature, two different approaches for FCMs learning are reported. The first borrows ideas from neural networks training and consists of unsupervised learning algorithms [2], [13], while the latter exploits the computational strength of evolutionary algorithms. Genetic algorithms [14], as well as Particle Swarm Optimization and Differential Evolution [11], [12], [16]–[18] have been used with success in different applications. Recently, a first investigation of a simple Memetic PSO approach was conducted on a control problem, with promising results [19].

III. MEMETIC PARTICLE SWARM OPTIMIZATION

A. Particle Swarm Optimization

PSO is a population–based optimization algorithm that gained increasing attention from the scientific community during the last years, mainly due to its simplicity and efficiency in solving difficult problems. The population, called a *swarm*, consists of individuals (agents), called *particles*, which probe the search space simultaneously [23]. Each particle moves with an adaptable velocity and it can store in a memory the best position it has ever visited in the search space. At each iteration, the particle moves stochastically in a new position that depends on its velocity, its own best position, as well as on information (best position) shared with a group of other particles, called its *neighborhood*.

Depending on the information exchange scheme, there are two main PSO variants. In the *global* variant, the best position

ever attained by all individuals of the swarm is communicated to all the particles at each iteration. In the *local* variant, each particle is assigned a neighborhood consisting of some of the particles and the best position ever attained by the particles that comprise the neighborhood is communicated among them [23]. The neighborhoods are determined based on the particles' indices rather than their actual distance in the search space, in order to avoid the heavy computational burden of computing the distances among all particles at each iteration of the algorithm.

Assume an n–dimensional search space, $S \subset \mathbb{R}^n$, and a swarm consisting of M particles. The i–th particle is an n–dimensional vector,

$$x_i = (x_{i1}, x_{i2}, \ldots, x_{in})^\top \in S.$$

Its velocity is also an n–dimensional vector,

$$v_i = (v_{i1}, v_{i2}, \ldots, v_{in})^\top.$$

The best previous position encountered by the i–th particle in S is denoted by,

$$p_i = (p_{i1}, p_{i2}, \ldots, p_{in})^\top \in S.$$

Ring topology, where the particles are assumed to lie on a ring, i.e., x_1 is the immediate neighbor of x_M, is the most common choice for the determination of neighborhoods. Thus, if r is the neighborhood's radius, then the neighborhood of x_i is defined as,

$$\{x_{i-r}, x_{i-r+1}, \ldots, x_i, \ldots, x_{i+r-1}, x_{i+r}\}.$$

Let g_i denote the index of the particle that attained the best previous position among all the particles in the neighborhood of x_i, and t to be the iteration counter. Then, the swarm is updated using the equations [24],

$$v_i^{(t+1)} = \chi \left[v_i^{(t)} + \varphi_1 \left(p_i^{(t)} - x_i^{(t)} \right) + \varphi_2 \left(p_{g_i}^{(t)} - x_i^{(t)} \right) \right] \quad (2)$$

$$x_i^{(t+1)} = x_i^{(t)} + v_i^{(t+1)}, \quad (3)$$

where $i = 1, 2, \ldots, M$; χ is the *constriction factor* that controls the magnitude of the velocity; and φ_1 and φ_2 are two positive random vectors with their components uniformly distributed within ranges $[0, \lambda_1]$ and $[0, \lambda_2]$, respectively. It is assumed that all operations between vectors are performed componentwise. The stability analysis of Clerc and Kennedy [24] suggested that χ is derived analytically:

$$\chi = \frac{2\kappa}{\left| 2 - \varphi - \sqrt{\varphi^2 - 4\varphi} \right|},$$

where $\varphi = \lambda_1 + \lambda_2$. The values received for $\varphi > 4$ and $\kappa = 1$ are considered the most common settings of χ due to their good average performance [24].

B. The Memetic Scheme

Memetic Algorithms (MAs) are also population–based algorithms that imitate adaptation in natural systems, where evolutionary adaptation of individuals is combined with individual learning within a lifetime. In practice, MAs usually include a local search phase as part of their operation. An important aspect that has been investigated in Memetic Genetic Algorithms is the Baldwin effect [25], [26], i.e., the interaction between local and global search component of the algorithm, where individual learning speeds up evolutionary change.

Recently, a Memetic PSO (MPSO) that combines PSO with the simple Random Walk with Direction Exploitation local search method [27] was introduced and applied successfully on a plethora of test problems [28]. A pseudocode of MPSO is provided below [28]:

Input: M (swarm size), χ, λ_1, λ_2, S, F (objective function)
Set $t = 0$
Initialize $x_i^{(t)}, v_i^{(t)} \in S$, $p_i^{(t)} \leftarrow x_i^{(t)}$, $i = 1, \ldots, M$
Evaluate $F\left(x_i^{(t)}\right)$
Determine the indices g_i, $i = 1, \ldots, M$
While (stopping criterion is not satisfied) **Do**
  Update the velocities $v_i^{(t+1)}$, using Eq. (2)
  Set $x_i^{(t+1)}$ according to Eq. (3)
  Constrain each particle x_i in S
  Evaluate $F\left(x_i^{(t+1)}\right)$, $i = 1, \ldots, M$
  If $F\left(x_i^{(t+1)}\right) < F\left(p_i^{(t)}\right)$ **Then** $p_i^{(t+1)} \leftarrow x_i^{(t+1)}$
  Else $p_i^{(t+1)} \leftarrow p_i^{(t)}$
  Update the indices g_i
  When (local search is applied) **Do**
    Choose a best position $p_q^{(t+1)}$, $q \in \{1, \ldots, M\}$
    Apply local search on $p_q^{(t+1)}$ and obtain y
    If $F(y) < F\left(p_q^{(t+1)}\right)$ **Then** $p_q^{(t+1)} \leftarrow y$
  End When
  Set $t = t + 1$
End While

Also, an MPSO scheme for FCMs learning that employs PSO and the Hooke and Jeeves (HJ) algorithm as local search component was proved to be competitive to the standard PSO learning methods on an industrial control problem [19].

In this paper, we extend our investigation by proposing two MPSO schemes that combine PSO with the HJ and Solis and Wets (SW) local search, respectively, with different configurations regarding the frequency and point of application of the local search. The two local search methods were selected due to their ability to work on non–differentiable functions (since they need only function values), their simplicity, as well as their flexibility in fitting a specific problem. Brief descriptions of the two methods are provided in the following paragraphs.

HJ is a deterministic pattern search algorithm [27], [29] that

searches the space along the coordinate axis using a suitable step size. More specifically, starting from an initial point, x_1, with function value $F_1 = F(x_1)$, the search increases the first component of x_1 by some step length Δ, moving to a new point that is called the *trial point*. If simple decrease in the function value is found, the trial point becomes the current iterate and the search continues on the next component of x_1. If no decrease appears in the function value, then the next trial point is formed by subtracting Δ from that component of x_1. Again, if improvement is found at the trial point, it becomes the current iterate of the search. Otherwise, the trial point retains the first component of the initial point. The processes continues for all the components of the starting point.

Upon completion of the coordinate search on all dimensions, the current iterate becomes the *base point*, x_2, with value F_2. If $F_2 < F_1$, a pattern extending step relocates to the new point $2x_2 - x_1$. This new point act as a temporary base point, x_{temp}, in the sense that it becomes the current iterate whether or not its value improves F_2. The HJ algorithm carries out a coordinate search in all dimensions of the temporary base point. If at the end of the coordinate search on x_{temp}, the value of the current iterate is less than F_2, then the current iterate becomes the new base point, x_3, with value F_3, and a pattern extending move is made to $2x_3 - x_2$, from where the search continues as above. On the other hand, if the current function value at the termination of the coordinate search on x_2 fails to improve F_2, then x_2 becomes the current iterate, Δ is reduced and the HJ algorithm conducts a coordinate search on x_2. This procedure takes place until a stopping criterion is met.

SW is a stochastic algorithm [30] that starts with an initial point x_k, $k = 0$. Then, a random point, x, is sampled in a region around x_k, following a multivariate normal distribution. If the function value of x is worse than the current one, a new point $2x_k - x$ is created. If this point does not improve x_k, then the current point is kept and a new sample point is generated. The covariance of the distribution determines the "diameter" of the aforementioned region, and its center is the point x_k with the addition of a bias factor, slanting the sampling in favor of the direction where success has been recorded. The algorithm uses also some parameters to adapt the center and covariance of the normal distribution, thereby changing the sampling region of the points. These parameters are the number of successive successes and failures in decreasing the value of F. If the new solutions are often improving, the region is expanded, while in the opposite case it is contracted.

Several memetic schemes can be produced depending on the point and frequency of application of the local search scheme. In the next section, the problem of the Ecological Industrial Park is described.

IV. THE ECOLOGICAL INDUSTRIAL PARK PROBLEM

Our application study considers an FCM model that has been constructed to study the impact of developing an Ecological Industrial Park (EIP) [31]. EIPs are characterized by a network of synergistic resource linkages among facilities within a defined geographical area. They are designed such that industrial areas are developed mimicking a natural ecosystem (self–contained, self–sustaining, and producing zero waste through complex interactions of food chains). We use the proposed FCM model for the Lloydminster EIP, at the Western Canadian plains. In our study, we consider the pollution concept as the output concept of our model and our goal was to detect the appropriate weights such that it lies within some bounds.

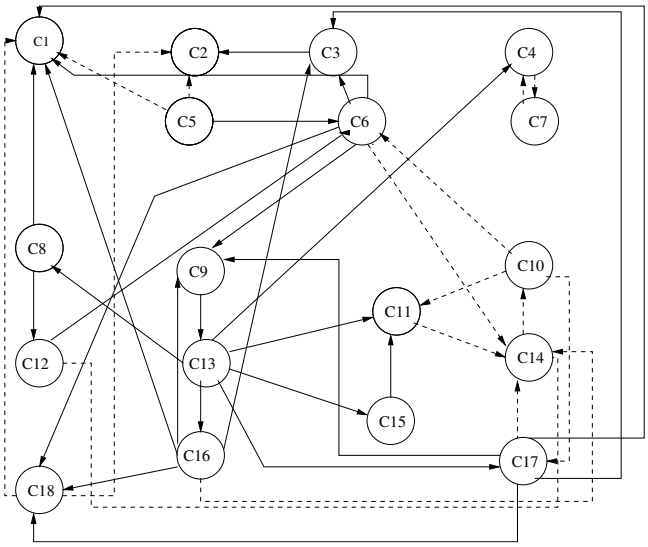

Fig. 2. The FCM for the Eco–Industrial Park.

The concepts of the FCM are the following:

C_1: Pollution
C_2: Water disposal
C_3: Unutilized byproducts/wastes
C_4: Demand
C_5: Byproducts/wastes provided by existing facilities
C_6: Secondary facilities
C_7: Availability
C_8: Vehicles
C_9: Employment
C_{10}: Property cost
C_{11}: Owned housing
C_{12}: Roads
C_{13}: Population
C_{14}: Available land
C_{15}: Rental cost
C_{16}: Schools and recreation facilities
C_{17}: Service facilities
C_{18}: Byproducts/wastes provided by co-locating facilities

The ranges of the 40 weights provided by the experts are reported in Table I. The objective function that is used for this

TABLE I
RANGES OF THE WEIGHTS OF THE FCM FOR THE ECOLOGICAL INDUSTRIAL PARK PROBLEM.

w_{32}	$[0.5, 1.0]$	w_{47}	$[-0.5, 0.0]$
w_{51}	$[-1.0, -0.5]$	w_{52}	$[-0.5, 0.0]$
w_{56}	$[0.0, 0.25]$	w_{61}	$[0.0, 0.25]$
w_{63}	$[0.0, 0.5]$	w_{69}	$[0.5, 1.0]$
$w_{6,14}$	$[-0.5, 0.0]$	$w_{6,18}$	$[0.0, 0.5]$
w_{74}	$[-0.5, 0.0]$	w_{81}	$[0.0, 0.25]$
$w_{9,13}$	$[0.5, 1.0]$	$w_{10,6}$	$[-0.5, 0.0]$
$w_{10,11}$	$[-1.0, -0.5]$	$w_{10,17}$	$[-0.5, 0.0]$
$w_{11,14}$	$[-0.5, 0.0]$	$w_{12,6}$	$[0.0, 0.25]$
$w_{12,14}$	$[-1.0, -0.5]$	$w_{13,4}$	$[0.5, 1.0]$
$w_{13,8}$	$[0.0, 0.5]$	$w_{13,11}$	$[0.0, 0.5]$
$w_{13,15}$	$[0.0, 0.5]$	$w_{13,17}$	$[0.0, 0.5]$
$w_{14,10}$	$[-0.5, 0.0]$	$w_{15,11}$	$[0.0, 0.5]$
$w_{16,1}$	$[0.0, 0.25]$	$w_{16,3}$	$[0.0, 0.5]$
$w_{16,9}$	$[0.5, 1.0]$	$w_{16,14}$	$[-0.5, 0.0]$
$w_{16,18}$	$[0.0, 0.5]$	$w_{17,1}$	$[0.0, 0.25]$
$w_{17,3}$	$[0.0, 0.5]$	$w_{17,9}$	$[0.5, 1.0]$
$w_{17,14}$	$[-1.0, -0.5]$	$w_{17,18}$	$[0.25, 0.55]$
$w_{18,1}$	$[-1.0, -0.5]$	$w_{18,2}$	$[-1.0, -0.5]$
$w_{8,12}$	$[0.25, 0.5]$	$w_{13,16}$	$[0.5, 0.75]$

TABLE II
RESULTS FOR CASE 1.

SS		PSO_g	PSO_g^{HJ}	PSO_g^{SW}	PSO_ℓ	PSO_ℓ^{HJ}	PSO_ℓ^{SW}
20	Suc	20	100	100	86	100	100
	Mean	196.0	285.3	5409.0	401.6	359.5	3161.4
	StD	88.6	85.9	7418.1	138.4	158.1	9266.5
	Min	100	151	99	120	212	100
	Max	480	768	36415	780	1166	78015
40	Suc	31	99	100	99	100	100
	Mean	361.3	355.6	8654.5	753.9	462.6	1447.4
	StD	160.2	73.4	11301.3	247.7	178.1	871.3
	Min	200	195	139	400	240	138
	Max	840	546	73138	1680	1280	5137
60	Suc	43	100	100	100	100	100
	Mean	477.2	412.2	9898.9	1026.6	566.5	1413.5
	StD	152.1	83.5	15841.1	262.6	198.2	722.2
	Min	240	247	180	480	287	180
	Max	840	747	102057	1620	1372	3778
80	Suc	36	100	99	100	100	100
	Mean	657.8	473.7	8056.5	1295.2	676.8	1718.9
	StD	210.5	77.4	14051.4	307.2	221.8	844.4
	Min	320	323	215	640	452	219
	Max	1600	742	67136	2000	1467	4277

problem is defined as [12]:

$$F(w) = \sum_{i=1}^{m} H\left(A_{\text{out}_i}^{\min} - A_{\text{out}_i}\right)\left|A_{\text{out}_i}^{\min} - A_{\text{out}_i}\right| + \sum_{i=1}^{m} H\left(A_{\text{out}_i} - A_{\text{out}_i}^{\max}\right)\left|A_{\text{out}_i}^{\max} - A_{\text{out}_i}\right|,$$

where H is the well–known Heaviside function and $A_{\text{out}_i}^{\min}$, $A_{\text{out}_i}^{\max}$, are bounds of the output concepts' values. In our experiments, the (pollution) output concept, C_1, was bounded in the range $[0.1, 0.2]$.

V. EXPERIMENTAL RESULTS

We investigated six algorithms, namely the standard global and local PSO, denoted as PSO_g and PSO_ℓ, respectively, as well as their memetic counterparts with HJ and SW local search, denoted as PSO_g^{HJ}, PSO_g^{SW}, PSO_ℓ^{HJ}, and PSO_ℓ^{SW}, respectively. The neighborhood radius for the local variants was always equal to 1. Also, the default set of PSO parameters, $\chi = 0.729$, $\lambda_1, \lambda_2 = 2.05$, was used [24]. The algorithms were compared in terms of the function evaluations required to reach the error goal 10^{-8}. There were 100 experiments performed for each problem, for different swarm sizes, 20, 40, 60 and 80. For each application of the local search, a maximum of 60 function evaluations were performed.

We investigated three cases regarding the application of the local search:

Case 1: Local search is applied on the best position at each iteration.

Case 2: Local search is applied on the best position at each iteration with probability 0.1.

Case 3: Local search is applied at each iteration on a best position that is selected through the well–known fitness proportionate (roulette wheel) selection.

All results are reported in Tables II–IV. At each table, and for each swarm size (denoted as SS) and algorithm, we report the number of successes in receiving the solution with the desired accuracy (out of 100 experiments), as well as the mean, standard deviation, minimum and maximum number of the expected number of required function evaluations, i.e., solely for the cases where the algorithm succeeded.

In Case 1, where local search was applied on the overall best position of the swarm at each iteration, the MPSO scheme with HJ exhibited the best performance, outperforming also the standard PSO learning schemes, with its global variant preceding the local one. The memetic methods based on the SW algorithm were also successful, although they required a significantly larger number of function evaluations on average. This can be attributed to the stochastic nature of the SW algorithm, in contrast to the deterministic local search of HJ. Also, the local variant of the SW–based memetic method was more efficient than the corresponding global variant. However, both variants had the highest standard deviation as well as the smallest minimum number of required function evaluations among all algorithms, which is indicative of their decreased robustness.

Similar performance was obtained also for Case 2, where local search was applied at each iteration on the overall best position with a probability equal to 0.1. The HJ–based schemes exhibited the best performance, having marginal differences between them. However, there was a significant decrease in the efficiency of PSO_g^{SW} and an improvement in

TABLE III
RESULTS FOR CASE 2.

SS		PSO_g	PSO_g^{HJ}	PSO_g^{SW}	PSO_ℓ	PSO_ℓ^{HJ}	PSO_ℓ^{SW}
	Suc	20	99	75	86	100	92
	Mean	196.0	341.1	7328.2	401.6	355.6	1807.3
20	StD	88.6	196.1	7981.4	138.4	96.9	4596.5
	Min	100	100	120	120	165	140
	Max	480	1445	24719	780	680	22119
	Suc	31	99	71	99	100	99
	Mean	361.2	586.4	9725.3	753.9	590.8	836.1
40	StD	160.1	432.1	13539.6	247.6	208.9	509.8
	Min	200	200	139	400	265	138
	Max	840	2146	45357	1680	1482	4539
	Suc	43	100	73	100	100	100
	Mean	477.2	726.2	9546.3	1026.6	764.2	1029.1
60	StD	152.1	538.7	14790.5	262.5	220.6	367.3
	Min	240	240	180	480	324	240
	Max	840	2825	65339	1620	1353	2220
	Suc	36	100	75	100	100	100
	Mean	657.7	835.4	17096.8	1295.2	975.1	1202.7
80	StD	210.4	590.7	25105.1	307.1	272.9	389.1
	Min	320	320	300	640	459	300
	Max	1600	3057	85839	2000	1665	2640

TABLE IV
RESULTS FOR CASE 3.

SS		PSO_g	PSO_g^{HJ}	PSO_g^{SW}	PSO_ℓ	PSO_ℓ^{HJ}	PSO_ℓ^{SW}
	Suc	20	98	100	86	100	100
	Mean	196.0	608.3	6378.6	401.6	901.5	1012.1
20	StD	88.6	116.1	12122.8	138.4	194.1	830.1
	Min	100	360	172	120	530	168
	Max	480	972	67125	780	1527	8077
	Suc	31	100	100	99	100	100
	Mean	361.2	733.2	6178.1	753.9	1164.0	1085.2
40	StD	160.1	182.4	10905.1	247.6	265.9	432.6
	Min	200	343	232	400	460	235
	Max	840	1703	44225	1680	1791	2313
	Suc	43	100	100	100	100	100
	Mean	477.2	876.3	7899.2	1026.6	1414.7	1326.1
60	StD	152.1	179.2	16443.5	262.5	282.5	519.8
	Min	240	530	294	480	685	294
	Max	840	1367	97012	1620	2167	3392
	Suc	36	100	99	100	100	100
	Mean	657.7	1018.6	7155.5	1295.2	1646.8	1452.5
80	StD	210.4	213.8	13776.7	307.1	303.7	517.7
	Min	320	515	352	640	805	350
	Max	1600	1633	76232	2000	2326	2712

the performance of PSO_ℓ^{HJ}, compared to Case 1. The SW–based schemes had again the smallest minimum number of required function evaluations.

In Case 3, where the selection of the best position for the application of local search was fitness proportionate, the PSO_ℓ^{SW} algorithm was more efficient than the corresponding variant with HJ, and the performance of PSO_g^{SW} was improved compared to the previous cases. The SW–based schemes retained also in this case the smallest minimum number of required function evaluations.

Summarizing the results, MPSO with HJ local search proved to be the most efficient algorithm in learning the FCM for the EIP problem. In the cases where local search was applied on the best position of the swarm at each iteration, the deterministic method of HJ enhanced significantly the performance of the memetic schemes, while the stochastic SW–based schemes were favored mostly in the roulette wheel selection case, an effect that can be attributed to the different nature of the two local search methods. Finally, although the global variants of the memetic algorithms required smaller mean number of function evaluations, local variants proved to be more robust in all cases, with the exception of PSO_ℓ^{SW} in Case 2 for small swarm sizes.

VI. CONCLUSIONS

We proposed an enhanced methodology for FCMs learning, based on a recently proposed Memetic Particle Swarm Optimization algorithm that utilizes the Hooke and Jeeves and Solis and Wets local search schemes. The algorithm was applied successfully for the detection of proper weights of an FCM that simulates an Ecological Industrial Park, such that the environmentally important concept of pollution lies within desirable bounds.

The algorithm proved to be heavily dependent on the local search scheme. HJ increased significantly the efficiency, outperforming both the standard PSO and the more randomized in nature SW–based learning schemes. Also, the frequency and point of application of the local search had an impact on the algorithm's performance, although milder than the local search algorithm itself.

Overall, MPSO with HJ exhibited the best performance, with its efficiency being more evident in higher swarm sizes, justifying its usefulness as an FCM learning scheme. Future research will consider further investigation of MPSO with different local search algorithms, as well as a deeper investigation of the effect of the local search on the algorithm's dynamics.

REFERENCES

[1] S. Hartmann, "The world as a process: Simulations in the natural and social sciences," in *Modelling and Simulation in the Social Sciences from the Philosophy of Science Point of View*, R. Hegselmann, U. Mueller, and K. G. Troitzsch, Eds. Kluwer, 1996, pp. 77–100.

[2] B. Kosko, "Fuzzy cognitive maps," *Int. J. Man–Machine Studies*, vol. 24, pp. 65–75, 1986.

[3] J. P. Craiger, D. F. Goodman, R. J. Weiss, and A. Butler, "Modeling organizational behavior with fuzzy cognitive maps," *Journal of Computational Intelligence and Organizations*, vol. 1, pp. 120–123, 1996.

[4] R. Taber, "Knowledge processing with fuzzy cognitive maps," *Expert Systems With Applications 2*, pp. 83–87, 1991.

[5] M. A. Styblinski and B. D. Meyer, "Fuzzy cognitive maps, signal fbw graphs, and qualitative circuit analysis," in *Proc. 2nd IEEE Int. Conf. Neural Networks*, San Diego, CA, USA, 1988, pp. 549–556.

[6] C. D. Stylios, V. Georgopoulos, and P. P. Groumpos, "Fuzzy cognitive map approach to process control systems," *J. Adv. Comp. Intell.*, vol. 3, no. 5, pp. 409–417, 1999.

[7] P. P. Groumpos and C. D. Stylios, "Modelling supervisory control systems using fuzzy cognitive maps," *Chaos, Solutions and Fractals*, vol. 11, pp. 329–336, 2000.

[8] C. D. Stylios and P. P. Groumpos, "The challenge of modelling supervisory systems using fuzzy cognitive maps," *J. Intelligent Manufacturing*, vol. 9, pp. 339–345, 1998.

[9] ——, "Fuzzy cognitive maps in modeling supervisory control systems," *J. Intelligent & Fuzzy Systems*, vol. 8, no. 2, pp. 83–98, 2000.

[10] V. Georgopoulos, G. Malandraki, and C. Stylios, "A fuzzy cognitive map approach to differential diagnosis of specific language impairment," *Artificial Intelligence in Medicine*, vol. 29, no. 3, pp. 261–278, 2003.

[11] K. E. Parsopoulos, E. I. Papageorgiou, P. P. Groumpos, and M. N. Vrahatis, "Evolutionary computation techniques for optimizing fuzzy cognitive maps in radiation therapy systems," in *Lecture Notes in Computer Science (LNCS)*. Springer Verlag, 2004, vol. 3102, pp. 402–413.

[12] E. I. Papageorgiou, K. E. Parsopoulos, C. D. Stylios, P. P. Groumpos, and M. N. Vrahatis, "Fuzzy cognitive maps learning using particle swarm optimization," *Journal of Intelligent Information Systems*, vol. 25, no. 1, pp. 95–121, 2005.

[13] E. I. Papageorgiou, C. D. Stylios, and P. P. Groumpos, "Active hebbian learning algorithm to train FCMs," *International Journal of Approximate Reasoning*, 2004, in press.

[14] M. S. Khan, S. Khor, and A. Chong, "Fuzzy cognitive maps with genetic algorithm for goal–oriented decision support," *International Journal of Uncertainty, Fuzziness and Knowledge-based Systems*, vol. 12, pp. 31–42, 2004.

[15] W. Stach, L. Kurgan, and W. Pedrycz, "Genetic learning of fuzzy cognitive maps," *Fuzzy Sets and Systems*, vol. 153, no. 3, pp. 371–401, 2005.

[16] E. I. Papageorgiou, K. E. Parsopoulos, P. P. Groumpos, and M. N. Vrahatis, "Fuzzy cognitive maps learning through swarm intelligence," in *Lecture Notes in Computer Science (LNAI)*, L. Rutkowski, J. Siekmann, R. Tadeusiewicz, and L. A. Zadeh, Eds. Springer Verlag, 2004, vol. 3070, pp. 344–349.

[17] K. E. Parsopoulos, E. I. Papageorgiou, P. P. Groumpos, and M. N. Vrahatis, "A first study of fuzzy cognitive maps learning using particle swarm optimization," in *Proceedings of the IEEE 2003 Congress on Evolutionary Computation*. Canberra, Australia: IEEE Press, 2003, pp. 1440–1447.

[18] K. Parsopoulos, E. Papageorgiou, P. Groumpos, and M. Vrahatis, "Recent advances in fuzzy cognitive maps learning using evolutionary computation techniques," in *Proceedings of the 1–st International Conference "From Scientific Computing to Computational Engineering" (IC-SCCE 2004)*, Athens, Greece, 2004.

[19] Y. G. Petalas, E. I. Papageorgiou, K. E. Parsopoulos, P. P. Groumpos, and M. N. Vrahatis, "Fuzzy cognitive maps learning using memetic algorithms," in *Proceedings of the International Conference of "Computational Methods in Sciences and Engineering" (ICCMSE 2005)*, 2005, pp. 1420–1423.

[20] R. Axelrod, *Structure of Decision: the Cognitive Maps of Political Elites*. Princeton, NJ: Princeton University Press, 1976.

[21] B. Kosko, *Fuzzy Engineering*. New York: Prentice Hall, 1997.

[22] C. D. Stylios and P. P. Groumpos, "Modeling complex systems using fuzzy cognitive maps," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 34, pp. 159–165, 2004.

[23] J. Kennedy and R. C. Eberhart, *Swarm Intelligence*. Morgan Kaufmann Publishers, 2001.

[24] M. Clerc and J. Kennedy, "The particle swarm–explosion, stability, and convergence in a multidimensional complex space," *IEEE Trans. Evol. Comput.*, vol. 6, no. 1, pp. 58–73, 2002.

[25] R. K. Belew, "Evolution, learning and culture:computational metaphores for adaptive algorithms," *Complex Systems*, vol. 4, pp. 11–49, 1990.

[26] G. E. Hinton and S. J. Nowlan, "How learning can guide evolution," *Complex Systems*, vol. 1, pp. 495–502, 1987.

[27] S. S. Rao, *Optimization: Theory and Applications*. Wiley Eastern, 1992.

[28] Y. G. Petalas, K. E. Parsopoulos, and M. N. Vrahatis, "Memetic particle swarm optimization," *Annals of Operations Research*, accepted for publication.

[29] R. Hooke and T. A. Jeeves, "Direct search solution of numerical and statistical problems," *J. ACM*, vol. 8, pp. 212–229, 1961.

[30] F. Solis and R. Wets, "Minimization by random search techniques," *Mathematics of Operations Research*, vol. 6, pp. 19–30, 1981.

[31] S. Fons, G. Achari, and T. Ross, "A fuzzy cognitive mapping analysis of the impacts of an eco–industrial park," *Journal of Intelligent and Fuzzy Systems*, vol. 15, no. 2, pp. 75–88, 2004.