Evolutionary Development of Fuzzy Cognitive Maps

Wojciech Stach, Lukasz Kurgan, Witold Pedrycz, and Marek Reformat
Department of Electrical and Computer Engineering
University of Alberta
Edmonton, Alberta T6G 2V4, Canada
{wstach, lkurgan, pedrycz, reform}@ece.ualberta.ca

Abstract—**Fuzzy cognitive maps (FCMs) form a convenient, simple, and powerful tool for simulation and analysis of dynamic systems. The popularity of FCMs stems from their simplicity and transparency. While being successful in a variety of application domains, FCMs are hindered by necessity of involving domain experts to develop the model. Since human experts are subjective and can handle only relatively simple networks (maps), there is an urgent need to develop methods for automated generation of FCM models. This study proposes a novel evolutionary learning that is able to generate FCM models from input historical data, and without any human intervention. The proposed method is based on genetic algorithms, and is carried out through supervised learning. The paper tests the method through a series of carefully selected experimental studies.**

Index Terms—**decision analysis, dynamic systems modelling, fuzzy cognitive maps, genetic algorithms**

I. Introduction

Fuzzy cognitive maps (FCMs) are constructs of Soft Computing introduced by Kosko in 1986 [9] as an extension of cognitive maps. They have been used for modeling and simulation of dynamic systems. FCMs represent a given system as a collection of concepts and mutual relations (dependencies), which are capable to incorporate and adapt human knowledge [17]. Modelling dynamic systems using FCMs exhibits several advantages. Most importantly FCMs are very simple and intuitive to understand, both in terms of the underlying formal model and its execution. They are also characterized by flexibility of system design and control, comprehensible structure and operation, adaptability to a given domain, and capability of abstract representation and fuzzy reasoning [11].

The FCMs were developed and used in numerous research and industrial areas, such as electrical engineering, medicine, political science, international relations, military science, history, supervisory systems, etc. Examples of specific applications include: diagnosis of diseases [23], analysis of electrical circuits [22], analysis of failure modes effects [18], fault management in distributed network environment [14], modeling of software development project [19] [20], and many others. However, development methods for FCM are far from being complete and well-defined, mainly because of the deficiencies that are present within the underlying design framework [12]. According to the literature, the development of FCM models almost always relies on human knowledge [1]. As a consequence, the developed models strongly depend on subjective beliefs of expert(s) from a given domain. A few algorithms for automated or semi-automated learning of FCMs have been proposed, but none of them provides a formalized approach assuring convergence [15]. Some of the proposed learning methods are quite limited as being applicable only to FCMs with binary states, other require multiple input datasets, which might be difficult to acquire. Most of them rely on human supervision during the learning process. Given these shortcomings, the objectives of this study are twofold:

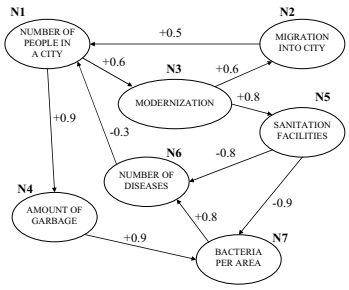
- We introduce a new learning method, which allows for the development of FCM model with *continuous* states based on experimental data, and without human intervention; and
- We carry out well-organized and thorough suite of experiments and come up with firm design guidelines

The remainder of this paper is organized as follows. Section II presents theoretical background concerning the model and learning methods, which includes relevant work. Section III introduces and provides background of the proposed learning approach, while Section IV presents comprehensive experimental evaluation and discussion of the achieved results. Finally, Section V offers conclusions and highlights future research directions.

II. Fuzzy Cognitive Maps (FCMs)

Following the generic concept introduced by Axelrod [2], their significant augmentation coming under the name of fuzzy cognitive maps (FCMs) was proposed by Kosko. The most significant enhancement lies in the way of reflecting causal relationships. Instead of using only the sign (+ or -), each edge is associated with a number (weight) that determines the degree of considered causal relation between the two concepts. This, in turn, allows implementing knowledge concerning the strength of such relationship, which now can be described by fuzzy terms, such as weak, medium, strong, or very strong. In other words, a weight of the directed edge from the node A to B quantifies how much concept A causes B [10]. The strength of relationship between two nodes (i.e., weight) is usually normalized to the [–1, 1] interval. The value of –1 represents full negative (inhibitory), +1 full positive, and 0 denotes no causal effect. As a result, a FCM model is fully described by set of nodes (concepts) and

edges (cause-effect relationships), represented by weights, between them. Apart from the graph representation, for computational purposes, the dependencies captured by this model can be equivalently expressed by a square matrix, called *connection matrix*, which stores all weight values for edges between corresponding concepts represented by corresponding rows and columns. An example of the FCM along with its connection matrix is shown in Fig. 1.



	N1	N2	N3	N4	N5	N6	N7
N1	0	0	0.6	0.9	0	0	0
N2	0.5	0	0	0	0	0	0
N3	0	0.6	0	0	0.8	0	0
N4	0	0	0	0	0	0	0.9
N5	0	0	0	0	0	-0.8	-0.9
N6	-0.3	0	0	0	0	0	0
N7	0	0	0	0	0	0.8	0

Fig. 1. FCM model describing public city health issues and its corresponding connection matrix [13]

It is instructive to recall a formal definition of a FCM and show all necessary notation. Let $\underline{\mathbf{R}}$ be the set of real numbers, $\underline{\mathbf{N}}$ denote the set of natural numbers, $K = [-1,1]$ and $L = [0,1]$.

A fuzzy cognitive map *F* is a 4-tuple (N, **E**, **C**, f) where

1) $N = \{N_1, N_2, ..., N_n\}$ is the set of *n* concepts forming the nodes of a graph

2) $\mathbf{E}: (N_i, N_j) \rightarrow e_{ij}$ is a function of $N \times N$ to K associating e_{ij} to a pair of concepts (N_i, N_j), with e_{ij} denoting a weight of directed edge from N_i to N_j if $i \neq j$ and e_{ij} equals zero if $i = j$. Thus $\mathbf{E}: (N \times N) = (e_{ij}) \in K^{n \times n}$ is a connection matrix.

3) $\mathbf{C}: N_i \rightarrow C_i$ is a function that at each concept N_i associates the sequence of its activation degrees such as for $t \in \underline{\mathbf{N}}, C_i(t) \in L$ given its activation degree at the moment t. $\mathbf{C}(0) \in L^n$ indicates the initial vector and specifies initial values of all concept nodes and $\mathbf{C}(t) \in L^n$ is a state vector at certain iteration t.

4) $f: \underline{\mathbf{R}} \rightarrow L$ is a transformation function, which includes recurring relationship on $t \geq 0$ between $\mathbf{C}(t+1)$ and $\mathbf{C}(t)$

$$\forall i \in \{1,...,n\}, C_i(t+1) = f\left(\sum_{\substack{j=1 \\ j \neq i}}^{n} e_{ji} C_j(t)\right) \quad (1)$$

(1) describes a functional model of FCM, which is used to perform simulations of the system dynamics. Simulation consists of computing state of the system, which is described by a state vector, over a number of successive iterations. The state vector specifies current values of all concepts (nodes) in a particular iteration. Value of a given node is calculated from the preceding iteration values of nodes, which exert influence on the given node through cause-effect relationship (nodes that are connected to the given node).

The transformation function is used to reduce unbounded weighted sum to a certain range, which is usually set to [0, 1]. The normalization hinders quantitative analysis, but allows for comparisons between nodes, which can be defined as active (value of 1), inactive (value of 0), or active to a certain degree (value between 0 and 1). Three most commonly used transformation functions concern bivalent, trivalent, and logistic nonlinearities.

Several types of simulation scenarios, which are dependent on transformation function, are possible [8]. Applying *discrete-output* transformation function (e.g. bivalent or trivalent function), the simulation heads to either a fixed state vector value, called *hidden pattern* or *fixed-point attractor*, or keeps cycling between a number of fixed state vector values, known as a *limit cycle*. Using a *continuous-output* transformation function (e.g., logistic one), the fixed-point attractor and limit cycle, as well as so called *chaotic attractor* can appear. The chaotic attractor appears when the FCM continues to produce different state vector values in successive cycles. In this study, we are concerned with the logistic transformation function:

$$f(x) = \frac{1}{1 + e^{-5x}} \quad (2)$$

Fig. 2 shows an example of this dynamic pattern. In a nutshell, simulation of a FCM results in a sequence of state vectors, which specify state of the modeled system in the successive iterations.

This paper introduces a learning method, which avoids disadvantages of the existing approaches. It uses a real-coded genetic (RCGA) algorithm to develop FCM connection matrix based on historical data consisting of one sequence of state vectors. With this regard, it is advantageous to identify main differences between the approach taken here and those already reported in the literature. A concise comparative summary of the learning scenarios is included in Table I. This table includes the methods considering essential design factors such as the learning objective, involvement of a domain expert, input historical data, type of transformation function, and a type of the learning strategy. It also shows, in the "number of nodes" column, for how many and of what size FCM model a given method was tested. All learning methods, except the RCGA, were tested on a single map of the indicated size. Entries in boldface point at the main disadvantages of a given learning method. We note that the

proposed method draws conclusions from the methods proposed in the past, and provides substantial advancement.

TABLE I
OVERVIEW OF LEARNING APPROACHES APPLIED TO FCMS

Algor	Ref	Learning goal	Human involved	Type of data used [a]	FCM type		Learning type
					Transform. function	# nodes	
DHL	[4]	Con. matrix	No	Single	**N/A**	**N/A**	Hebbian
BDA	[25]	Con. matrix	No	Single	**Binary**	5, 7, 9	modified Hebbian
NHL	[16]	Con. matrix	**Yes& No** [b]	Single	Continuous	5	modified Hebbian
GS	[12]	Con. matrix	No	**Multiple**	Continuous	7	Genetic
PSO	[17]	Con. matrix	No	**Multiple**	Continuous	5	Swarm
GA	[7]	**Initial vector**	N/A	N/A	Continuous	11	Genetic
RCGA	this paper	Con. matrix	No	Single	Continuous	4, 6, 8, 10	Genetic

a) *Single* – historical data consisting of one sequence of state vectors, *Multiple* – historical data consisting of several sequences of state vectors, for different initial conditions

b) Initial human intervention is necessary but later when applying the algorithm there is no human intervention needed

III. PROPOSED LEARNING METHOD

The essence of our learning is to optimize the connection matrix via genetic learning.

A. Problem Statement

Given set of concepts N, sequence of their activations degree, called input data, $\mathbf{C}(t)$ at certain iteration interval $t \in \{0,...,t_K\}$, and transformation function f

Our objective is to establish the connection matrix $\hat{\mathbf{E}}$, such that FCM model expressed by the 4-tuple $(N, \hat{\mathbf{E}}, \hat{\mathbf{C}}, f)$ minimizes the error between given sequence $\mathbf{C}(t)$ and sequence $\hat{\mathbf{C}}(t)$ obtained from simulation model performed according to the formula (1) subject to initial condition $\hat{\mathbf{C}}(0) = \mathbf{C}(0)$. The error is measured using criterion described in Section III.B

The aim of the proposed learning method is to eliminate human intervention during development of a FCM model. This process is performed by exploiting information from historical data to compute FCM model connection matrix that is able to mimic the data. The input (historical) data comprise of one sequence of state vectors over time. The *data length* is defined as the number of successive iterations (time points) of the given historical data. The input data is used to compute a FCM model, called *candidate FCM*, by applying a learning procedure that uses RCGA algorithms.

Assuming that concepts do not exhibit cause-effect relationships on themselves, the connection matrix of a FCM can be completely expressed by N(N-1) variables, where N denotes the number of concepts. The proposed learning algorithm uses input data to find the parameters. Input data is a sequence of states described by state vectors at a particular time (iteration). They illustrate the system's behavior over time, and are represented by a set of state vectors $\mathbf{C}(t)$ at time point *t*; see Fig. 2.

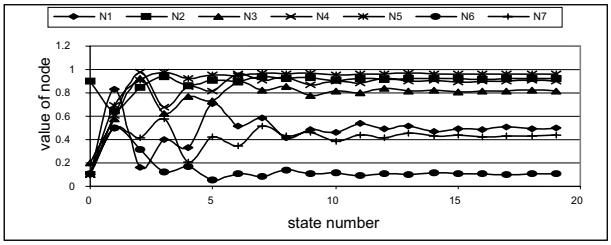


Fig. 2. Example input data

The proposed learning method constructs a connection matrix based on input data. The learned model objective is to generate the same state vector sequence for the same initial state vector, as it is defined by the input data. At the same time, the learned model generalizes the inter-relationship between concept nodes, which are inferred from the input data. Therefore, the FCM model is suitable to perform simulation for different initial state vectors, and quantify the degree and type of the cause-effect relationships between the concepts.

B. Proposed Genetic Algorithms Based Learning Method

The proposed learning method uses an extended GA called real-coded genetic (RCGA) algorithm, where a chromosome consists of floating point numbers. A useful summary about relevant GAs can be found in [3] [5] [6]. A high-level overview of the learning process is shown in Fig. 3.

if (Fitness function(best individual) > max_fitness or *t>max_generation)*
then *stopping_condition* = ***true****;*
where: *best individual* – the chromosome in the current generation with highest fitness function value, *t* – current generation number

Fig. 3. High-level diagram of the proposed learning method

The RCGA algorithm uses the input data to develop and optimize, with respect to the input data, connection matrix of a candidate FCM model. RCGA defines each of its chromosomes as a floating-point vector. The length of the chromosome corresponds to the number of variables. Each element of the vector is called *gene*. In case of the learning FCMs, each chromosome consists of N(N-1) genes, which are floating point numbers from the range [-1,1]:

$$\hat{\mathbf{E}} = \left[e_{12}, e_{13}, ..., e_{1N}, e_{21}, e_{23}, ..., e_{2N}, ..., e_{NN-1}\right]^T$$

where e_{ij} specifies the value of a weight for an edge from i^{th} to j^{th} concept node

The design of fitness function takes advantage of a specific feature of the FCM theory. At each iteration of FCM model

simulation state vector **C**(t+1) depends only on the state vector at the preceding iteration. Let us assume that the input data length is *K*. By grouping each two adjacent state vectors, *K-1* different pairs can be formed

$$C(t) \rightarrow C(t+1) \ \forall t = 0,...,K-1 \quad (3)$$

If we define **C**(t) as an *initial vector,* and **C**(t+1) as *system response, K-1* pairs in the form of {initial vector, system response} can be generated from the input data. The larger *K* is the more information about the system behavior we have and the more accurate learning can be performed [21]. The fitness function is calculated for each chromosome by computing the difference between system response generated using a candidate FCM and a corresponding system response, which is defined in the input data. This difference is computed across all *K-1* initial vector / system response pairs, and for the same initial state vector as

$$Difference = \frac{1}{(\mathrm{K}-1)\cdot \mathrm{N}} \sum_{t=1}^{K-1} \sum_{n=1}^{N} \left| C_n(t) - \hat{C}_n(t) \right|^2 \quad (4)$$

where

$\mathbf{C}(t) = \left[C_1(t), C_2(t), ..., C_n(t) \right]$ – given system response for $\mathbf{C}(\mathrm{t}-1)$ initial vector,

$\hat{\mathbf{C}}(t) = \left[\hat{C}_1(t), \hat{C}_2(t), ..., \hat{C}_n(t) \right]$ – system response of the candidate FCM for $\mathbf{C}(t-1)$ initial vector

Fitness function is defined as:

$$h(Difference) = \frac{1}{10000 \cdot Difference + 1} \quad (5)$$

Other RCGA parameters include:

- recombination method – single-point crossover;
- mutation method – randomly chosen from random mutation, non-uniform mutation, and Mühlenbein's mutation;
- selection method – randomly chosen from roulette wheel and tournament;
- probability of recombination: 0.9;
- probability of mutation: 0.5;
- *population_size*: 100 chromosomes;
- *max_generation*: 300000;
- *max_fitness*: 0.999;

The above values were established experimentally.

IV. Experiments and Results

A. Experimental Setup

The goal of the experiments is to assess quality of the proposed method for learning FCMs. Tests are divided into two groups: tests performed with *synthetic*, and with *real-life* data.

- synthetic tests use randomly generated input FCM models to generate synthetic input data, which is used to learn candidate FCM. A wide range of input FCM models was considered to accommodate different types of real-life FCM models. As a result, the tests were performed with FCMs with 4, 6, 8, and 10 nodes, and with densities of 20%, 40%, 60%, and 80%, which results in sixteen test configurations.
- real-life data tests were performed with large 7 nodes FCM model that have been reported in literature. In this case the FCM was predefined by the original author (domain expert). The experiments involved simulating the input FCM to generate input data, and later using the input data to generate a candidate FCM

A diagram presenting an overview of the test procedure is shown in Fig. 4.

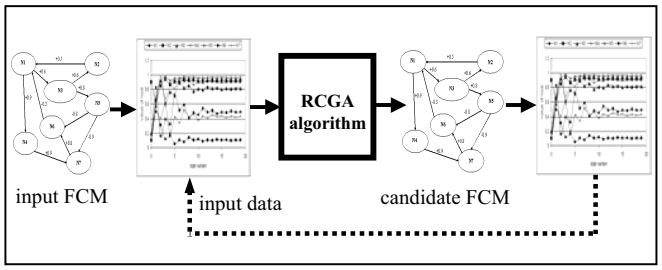


Fig. 4. High-level diagram of the experimental setup

In general, the goal of learning is to find FCM connection matrix that generates the same state vector sequence as the input data for a given initial state vector.

Error criterion, which measures similarity between the input data, and data generated by simulating the candidate FCM with the same initial state vector as for the input data is used to evaluate quality of learning. The criterion is defined as a normalized average error between corresponding concept values at each iteration between the two state vector sequences:

$$error = \frac{1}{(K-1)N} \sum_{t=1}^{K-1} \sum_{n=1}^{N} \left| C_n(t) - \hat{C}_n(t) \right| \quad (6)$$

where

$\mathrm{C_n}(t)$ is value of a node *n* at iteration *t* in the input data,

$\hat{\mathrm{C}}_\mathrm{n}(t)$ is value of a node *n* at iteration *t* from simulation of the candidate FCM

K is input data length

N is number of nodes

Since the methods listed in Table I have different learning goals and the description of test procedures they applied was insufficient to repeat exactly the same set of experiments, we were not able to perform comprehensive comparison tests. However, in order to put the quality of the proposed learning method into a perspective, a set of reference results were computed. This reference was generated by computing error values for randomly generated FCMs and comparing them with the input FCM. Ten random FCMs were generated for

each test category, and the average value of the two criteria was reported. Several minor assumptions were also made. For both, the input and candidate FCMs, all weight values smaller than 0.05 were rounded down to 0, since no real-life map considers such weak relationships to be of any relevance. Moreover, during simulations all nodes values are rounded to two digits after decimal point, which results from a trade-off between model comprehensibility and accuracy of relationship representation.

B. Experiments Outcome

B.1. Results for Synthetic Data

A total of almost 200 tests were performed. For each FCM size and density 10 tests were performed, and average results are reported. Table II shows summary of the results for the considered number of concept nodes and densities.

TABLE II
Experimental results for the synthetic data

nodes	density [%]	error ± stdev	nodes	density [%]	error ± stdev
4	20	0.000 ± 0.000	8	20	0.057 ± 0.043
4	40	0.000 ± 0.000	8	40	0.015 ± 0.021
4	60	0.000 ± 0.000	8	60	0.014 ± 0.020
4	80	0.000 ± 0.000	8	80	0.006 ± 0.008
6	20	0.005 ± 0.005	10	20	0.088 ± 0.095
6	40	0.005 ± 0.006	10	40	0.037 ± 0.048
6	60	0.004 ± 0.004	10	60	0.026 ± 0.039
6	80	0.003 ± 0.003	10	80	0.006 ± 0.009

Legend:
- nodes – number of nodes of the input FCM
- density [%] – ratio of non-zero weights to the total number of weights

To ease the analysis of the results, a relation between FCM parameters, i.e. size and density, and each of the evaluation criteria was presented in Fig. 5, which includes a table with the average values of the corresponding criterion across different sizes and densities of the input FCMs. The content of the table is also represented as graph, which additionally includes the baseline results.

		Number of nodes			
		4	6	8	10
Density	20%	0.000	0.005	0.057	0.088
	40%	0.000	0.005	0.015	0.037
	60%	0.000	0.004	0.014	0.026
	80%	0.000	0.003	0.006	0.006
baseline		0.391			

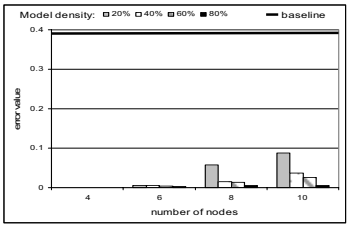


Fig. 5. Error values as a function of number of nodes and FCM density

Error values are relatively small, compared to the baseline values, for all considered experiments. We note that the error values slightly increase with the increasing size and decreasing density of the input FCM, but even for the 10 nodes and 30% dense FCM the value indicates finding high quality candidate FCM. The results show that the proposed learning method is able to find FCM that can closely mimic the input data proving usefulness of the genetic algorithm based learning for this problem.

B.2. Results for Real Life Data

The experiments were performed with FCM model proposed by Tsadiras, which concerns business industry and financial activities [24]. This FCM describes relationships among seven concepts, which were identified as important in the strategic planning process of a e-business company. The following concepts were considered: e-business profits, e-business sales, prices cutoffs, customer satisfaction, staff recruitments, impact from international e-business competition, and better e-commerce services. The density of the considered FCM was 40%.

The input data for learning, shown in Fig. 6, was generated with a randomly chosen initial state vector, and reaches the fixed-point attractor state after 10 iterations. The input data was applied to the proposed RCGA learning algorithm, resulting in a learning progress shown in Fig. 7.

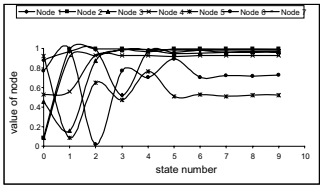


Fig. 6. Input data for e-business company FCM

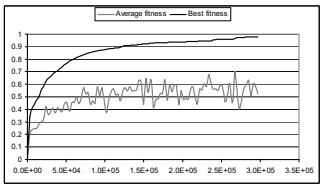
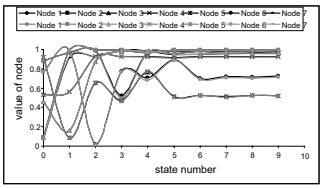


Fig. 7. Learning process for the e-business company FCM

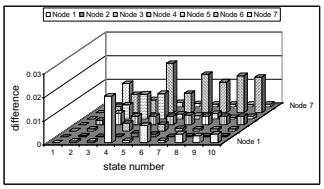
The learning results were evaluated identically as in case of the experiments with the synthetic data.

The error (± stdev) value was 0.004 (± 0.005).

Results achieved by the RCGA algorithm, in terms of comparison of state vector sequences and error defined by the difference between the input data and the generated sequence, for the e-business FCM, are presented; see Fig. 8.



a) Input data and plot obtained from simulation of candidate FCM



b) Difference between corresponding states vector values for plots from a)

Fig. 8. Selected results achieved for the e-business FCM.

The above results show high usefulness of this learning method. The quality of generated candidate FCM is comparable with the corresponding results obtained from synthetic data.

V. Conclusions and Further Directions

In this study, we have developed a comprehensive learning environment for the design of fuzzy cognitive maps. It has been demonstrated how genetic optimization help construct maps on a basis of numeric data. We showed and quantified the feasibility and effectiveness of the evolutionary approach via series of numeric experiments

The paper discussed relevant work, and proposes and tests a novel learning strategy, based on a real-coded genetic algorithm. The method is able to generate a FCM model from input data consisting of a single sequence of state vector values. A comprehensive set of tests was performed, including experiments with both synthetic and real-life data, and different sizes and densities of FCMs. The results show that the proposed learning method is very effective, and generates FCM models that can almost perfectly simulate the input data. We note that the quality of learning deteriorates with the increasing size of considered FCMs. In general, the proposed method achieved excellent quality for maps up to 6 nodes, while for maps up to 10 nodes that quality is still satisfactory. Since many different configurations of FCMs have been tested, the results can be also treated as a testbed for future learning methods.

The future work will concern the use of the learning method in a context of practical applications. Those could involve areas such as, e.g., stock exchange, sports bets, weather conditions or seismic hazards.

Acknowledgements

This research was supported by the Natural Sciences & Engineering Research Council of Canada (NSERC).

References

[1] J. Aguilar, "A survey about fuzzy cognitive maps papers," *International Journal of Computational Cognition,* vol. 3, no. 2, pp. 27-33, 2005

[2] R., Axelrod, *Structure of Decision: The Cognitive Maps of Political Elites*, Princeton University Press, 1976

[3] K. Deb, "An introduction to genetic algorithms," *Sadhana* vol. 24, no. 4, pp. 205- 230, 1999

[4] J.A. Dickerson, and B. Kosko, "Virtual worlds as fuzzy cognitive maps," *Presence*, vol. 3, no. 2, pp. 173-189, 1994

[5] D.E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, 1989

[6] F. Herrera, M. Lozano, and J.L. Verdegay, "Tackling real-coded genetic algorithms: operators and tools for behavioural analysis," *Artificial Intelligence Review*, vol. 12, no. 4, pp. 265-319, 1998

[7] M.S. Khan, and A. Chong, "Fuzzy cognitive map analysis with genetic algorithm," *Proceedings of the 1st Indian International Conference on Artificial Intelligence (IICAI-03)*, 2003

[8] M. Khan, and M. Quaddus, "Group decision support using fuzzy cognitive maps for causal reasoning," *Group Decision and Negotiation Journal*, vol. 13, no. 5, 2004, in press.

[9] B. Kosko, "Fuzzy cognitive maps," *International Journal of Man-Machine Studies*, vol. 24, pp. 65-75, 1986

[10] B. Kosko, *Neural Networks and Fuzzy Systems*, Prentice-Hall, 1992.

[11] D.E. Koulouriotis, I.E. Diakoulakis, D.M. Emiris, E.N. Antonidakis, and I.A. Kaliakatsos, "Efficiently modeling and controlling complex dynamic systems using evolutionary fuzzy cognitive maps (invited paper)," *International Journal of Computational Cognition,* vol. 1, no. 2, pp. 41-65, 2003

[12] D.E. Koulouriotis, I.E. Diakoulakis, and D.M. Emiris, "Learning fuzzy cognitive maps using evolution strategies: a novel schema for modeling and simulating high-level behavior," *IEEE Congress on Evolutionary Computation (CEC2001)*, pp. 364-371, 2001

[13] K.C. Lee, W.J. Lee, O.B. Kwon, J.H. Han, and P.I. Yu, "Strategic planning simulation based on fuzzy cognitive map knowledge and differential game," *Simulation*, vol. 71, no. 5, pp. 316-327, 1998.

[14] T.D. Ndousse, and T. Okuda, "Computational intelligence for distributed fault management in networks using fuzzy cognitive maps," *Proceedings of the IEEE International Conference on Communications Converging Technologies for Tomorrow's Application*, pp. 1558-1562, 1996

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

[16] E. Papageorgiou, C.D. Stylios, and P.P. Groumpos, "Fuzzy cognitive map learning based on nonlinear hebbian rule," *Australian Conference on Artificial Intelligence*, pp. 256-268, 2003

[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," *Proceedings of the IEEE 2003 Congress on Evolutionary Computation*, pp. 1440-1447, 2003

[18] C.E. Pelaez, and J.B. Bowles, "Applying fuzzy cognitive maps knowledge representation to failure modes effects analysis," *Proceedings of the IEEE Annual Symposium on Reliability and Maintainability*, pp. 450-456, 1995

[19] W. Stach, and L. Kurgan, "Modeling software development project using fuzzy cognitive maps," *Proceedings of the 4th ASERC Workshop on Quantitative and Soft Software Engineering (QSSE'04)*, pp. 55-60, 2004

[20] W. Stach, L. Kurgan, W. Pedrycz, and M. Reformat, "Parallel fuzzy cognitive maps as a tool for modeling software development project," *Proceedings of the 2004 North American Fuzzy Information Processing Society Conference (NAFIPS'04)*, pp. 28-33, Banff, AB, 2004

[21] W. Stach, L. Kurgan, W. Pedrycz, and M. Reformat, "Learning fuzzy cognitive maps with required precision using genetic algorithm approach," *Electronics Letters*, vol. 40, no. 24, pp. 1519-1520, 2004

[22] M.A. Styblinski, and B.D. Meyer, "Signal flow graphs versus fuzzy cognitive maps in application to qualitative circuit analysis," *International Journal of Man-Machine Studies*, vol. 35, pp.175-186, 1991

[23] R. Taber, "Knowledge processing with fuzzy cognitive maps," *Expert Systems with Applications*, vol. 2, pp. 83-87, 1991

[24] A.K. Tsadiras, "Using fuzzy cognitive maps for e-commerce strategic planning," *Proceedings of the 9th Panhellenic Conference on Informatics (EPY' 2003)*, 2003

[25] A. Vazquez, "A balanced differential learning algorithm in fuzzy cognitive maps," *Technical Report, Departament de Llenguatges I Sistemes Informatics, Universitat Politecnica de Catalunya (UPC)*, 2002