Fuzzy Logic Adaptation of a Hybrid Evolutionary Method for Pattern Recognition

Fevrier Valdez[1] Patricia Melin[2] Oscar Castillo[2]

[1]PhD Student of Computer Science in the Universidad Autónoma de Baja California
Tijuana, B.C.
[2]Computer Science in the Graduate Division, Tijuana Institute of Technology
Tijuana, B.C.
Email: fvaldez0125@yahoo.com.mx, pmelin@tectijuana.mx, ocastillo@tectijuana.mx

Abstract— *We describe in this paper a new hybrid approach for optimization combining Particle Swarm Optimization (PSO) and Genetic Algorithms (GAs) using Fuzzy Logic to integrate the results. The new evolutionary method combines the advantages of PSO and GA to give us an improved FPSO+FGA hybrid method. Fuzzy Logic is used to combine the results of the PSO and GA in the best way possible. The new hybrid FPSO+FGA approach is compared with the PSO and GA methods with a set of benchmark mathematical functions. The proposed hybrid method is also tested with the problem of modular neural network optimization. The new hybrid FPSO+FGA method is shown to be superior with respect to both the individual evolutionary methods.*

Keywords— Fuzzy Logic, Evolutionary Computing, Genetic Algorithms**.**

1 Introduction

We describe in this paper a new evolutionary method combining PSO and GA, to give us an improved FPSO+FGA hybrid method. We apply the hybrid method to mathematical function optimization to validate the new approach. Also in this paper, the application of a Genetic Algorithm (GA) [1] and Particle Swarm Optimization (PSO) [2] for the optimization of mathematical functions is considered. In this case, we are using the Rastrigin's function, Rosenbrock's function, Ackley's function, Sphere's function Griewank's function, Michalewics's function and Zakharov's function [4][13] to compare the optimization results between a GA, PSO and FPSO+FGA. We also consider the problem of neural network architecture optimization, which is very important in the applications.

The paper is organized as follows: in section 2 a description about the Genetic Algorithms for optimization problems is given, in section 3 the Particle Swarm Optimization is presented, the neural networks is presented in section 4, in section 5 we can appreciate the proposed method FPSO+FGA and the fuzzy system, in section 6 we can appreciate the full model of FPSO+FGA that was used for this research, in section 7 the simulations results are described, finally we can see in section 8 the conclusions reached after the study of the proposed evolutionary computing methods.

2 Genetic Algorithms for Optimization

John Holland, from the University of Michigan initiated his work on genetic algorithms at the beginning of the 1960s. His first achievement was the publication of *Adaptation in Natural and Artificial System* [7] in 1975.

He had two goals in mind: to improve the understanding of natural adaptation process, and to design artificial systems having properties similar to natural systems [8].

The basic idea is as follows: the genetic pool of a given population potentially contains the solution, or a better solution, to a given adaptive problem. This solution is not "active" because the genetic combination on which it relies is split between several subjects. Only the association of different genomes can lead to the solution.

Holland's method is especially effective because it not only considers the role of mutation, but it also uses genetic recombination, (crossover) [9]. The crossover of partial solutions greatly improves the capability of the algorithm to approach, and eventually find, the optimal solution.

The essence of the GA in both theoretical and practical domains has been well demonstrated [1]. The concept of applying a GA to solve engineering problems is feasible and sound. However, despite the distinct advantages of a GA for solving complicated, constrained and multi-objective functions where other techniques may have failed, the full power of the GA in application is yet to be exploited [12] [14].

In Fig. 1 we show the reproduction cycle of the Genetic Algorithm.

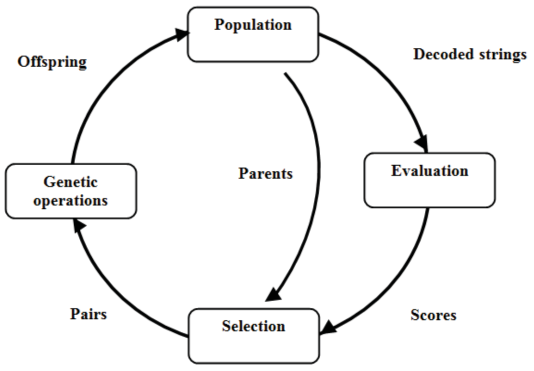

Figure 1: The Reproduction cycle

ISBN: 978-989-95079-6-8

3 Particle Swarm Optimization

Particle swarm optimization (PSO) is a population based stochastic optimization technique developed by Eberhart and Kennedy in 1995, inspired by social behavior of bird flocking or fish schooling [3].

PSO shares many similarities with evolutionary computation techniques such as Genetic Algorithms (GA) [6]. The system is initialized with a population of random solutions and searches for optima by updating generations. However, unlike GA, the PSO has no evolution operators such as crossover and mutation. In PSO, the potential solutions, called particles, fly through the problem space by following the current optimum particles [10].

Each particle keeps track of its coordinates in the problem space, which are associated with the best solution (fitness) it has achieved so far (The fitness value is also stored). This value is called *pbest*. Another "best" value that is tracked by the particle swarm optimizer is the best value, obtained so far by any particle in the neighbors of the particle. This location is called *lbest*. When a particle takes all the population as its topological neighbors, the best value is a global best and is called *gbest* [19].

The particle swarm optimization concept consists of, at each time step, changing the velocity of (accelerating) each particle toward its p*best* and *lbest* locations (local version of PSO). Acceleration is weighted by a random term, with separate random numbers being generated for acceleration toward *pbest* and *lbest* locations [17].

In the past several years, PSO has been successfully applied in many research and application areas. It is demonstrated that PSO gets better results in a faster, cheaper way compared with other methods [11] [18].

Another reason that PSO is attractive is that there are few parameters to adjust. One version, with slight variations, works well in a wide variety of applications. Particle swarm optimization has been used for approaches that can be used across a wide range of applications, as well as for specific applications focused on a specific requirement [20].

4 Neural Networks

The discipline of neural networks originates from an understanding of the human brain. The average human brain consists of 3 × 1010 neurons of various types, with each neuron connecting to up to 104 synapses [21]. Many neural-network models, also called connectionist models, have been proposed [6].

Neural networks are attractive since they consist of many neurons, with each one processing information separately and simultaneously. All the neurons are connected by synapses with variable weights. Thus, neural networks are actually parallel distributed processing systems.

Research on neural networks dates back to the 1940s when McCulloch and Pitts found that the neuron can be modeled as a simple threshold device to perform logic function [22].

In this paper, we used a supervised learning for training the neural network. Supervised learning is based on a direct comparison between the actual network output and the desired output. Some recent advances in supervised learning have been reviewed in [5].

5 FPSO+FGA Method

This method combines the characteristics of PSO and GA using several fuzzy systems for integration of results and parameter adaptation. In this section, the proposed FPSO+FGA method is presented.

The general idea of the proposed FPSO+FGA method can be seen in Fig. 2. The method can be described as follows:

1. A problem optimization to be considered is received in this case; we are testing with a modular neural network and mathematical functions.
2. The method is randomly initialized to start with FPSO or FGA.
3. The system will solve the problem by FPSO or FGA depending of Error (E) and Derivate of Error (DE) generated after applying FPSO or FGA.
4. The function evaluated is called FEVAL, then the system records the Errors (E and DE) obtained to compare with the value desired.
5. After that the function is evaluated, E and DE are taken as inputs to one fuzzy system called 'FSDM', which the main function is to decide if is suitable to continue solving the problem with FPSO or FGA depending of the information generated by the fuzzy rules to calculate when can be necessary do a change
6. Another two fuzzy systems receive the values of Error and DError as inputs to evaluate if it is necessary to change the values of the parameters in FPSO or FGA depending of the fuzzy rules. The fuzzy systems responsible to do change the value of parameters in FPSO and FGA are called 'fuzzyga' and 'fuzzypso'. On Fig. 2 the two fuzzy systems are represented by el block FSPA (Fuzzy System Parameter Adaptation) but while the method is running in the FPSO the fuzzy system changes the values of the cognitive acceleration 'c_1', and social acceleration 'c_2'; and while the method is running in the FGA the fuzzy system change the value of crossover and mutation.
7. Repeat the above steps until the termination criterion of the algorithm is met. Finally BVALUE receive the best value obtained and when the method is finished we are obtaining the best result.

The fundamental characteristic of this method is to adapt the parameters to reach the best result, in this case; we are doing changes on the parameters of crossover (cr), mutation (mu), cognitive acceleration (c_1) and social acceleration (c_2). We are taking this parameters because are very important to the convergence with good results of the method, for example 'c_1' and 'c_2' are working as a memory able of record the knowledge of the particles more important in a population in the FPSO; also 'cr' and 'mu' are important to reach the best solution in the FGA. The values of these four parameters are fuzzy because depending of the output of a fuzzy system able of adaptation parameters.

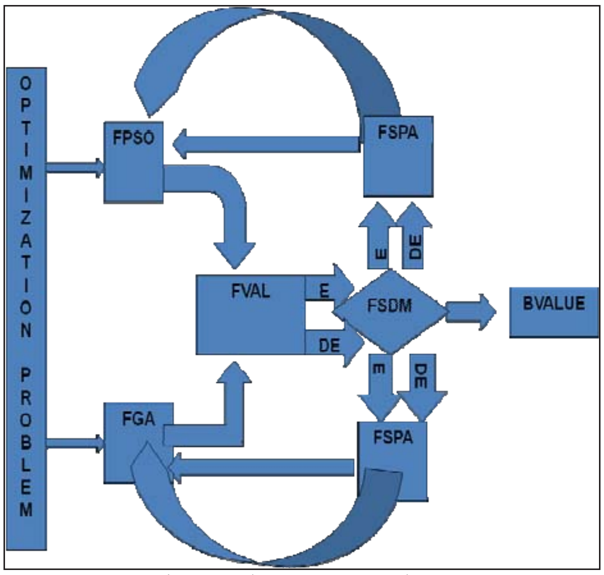

Figure 2: The FPSO+FGA scheme

6 Full Model of FPSO+FGA

The basic idea of the FPSO+FGA scheme is to combine the advantages of the individual methods using a fuzzy system for decision making and the others two fuzzy systems to improve the parameters of the FGA and FPSO when is necessary.

As can be seen in the proposed hybrid FPSO+FGA method, it is the internal fuzzy system structure, which has the primary function of receiving as inputs (Error and DError) the results of the outputs FGA and FPSO. The fuzzy system is responsible for integrating and deciding which are the best results being generated at run time of the FPSO+FGA. It is also responsible for selecting and sending the problem to the “fuzzypso” fuzzy system when the FPSO is activated or to the “fuzzyga” fuzzy system when FGA is activated. Also activating or temporarily stopping depending on the results being generated. The fuzzy system is of Mamdani type because it is more common in this type of fuzzy control and the defuzzification method is the centroid. In this case, we are using this type of defuzzification because in other papers we have achieved good results [4]. Also, the membership functions in the fuzzy main system were chosen of triangular form based on past experiences in this type of fuzzy control. However, in the simulation results we can see different architectures to the ‘fuzzyga’ and the ‘fuzzypso’; where, we are changing the membership functions to observe which is more reliable to this application. In this research, we are testing with triangular and gaussian membership functions.

6.1 *FPSO (Fuzzy Particle Swarm Optimization)*

This section presents a detailed description of the FPSO model. The classical representation scheme for GAs is binary vectors of fixed length. In the case of an n_x – dimensional search space, each individual consists of n_x variables with each variable encoded as a binary string. The swarm is typically modeled by particles in multidimensional space that have a position and a velocity. These particles fly through hyperspace (i.e., R^n) and have two essential reasoning capabilities: their memory of their own best position and knowledge of the global or their neighborhood's best. In Fig. 3 we can see a simulation of Ackley’s function. In a minimization optimization problem, "best" simply means the position with the smallest objective value. Members of a swarm communicate good positions to each other and adjust their own position and velocity based on these good positions. So a particle has the following information to make a suitable change in its position and velocity:

- A global best that is known to all and immediately updated when a new best position is found by any particle in the swarm.
- The neighborhood best that the particle obtains by communicating with a subset of the swarm.
- The local best, which is the best solution that the particle has seen.

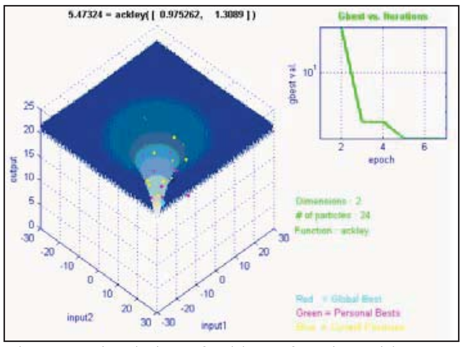

Figure 3: Simulation of ackley’s function with FPSO

In this case, the social information is the best position found by the swarm, referred as $\hat{y}$ (t). For gbest FPSO, the velocity of particle i is calculated as

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

Where $vi_j(t)$ is the velocity of particle i in dimension j = 1,…,n_x at time step t, $xi_j(t)$ is the position of particle i in dimension j at time step t, ‘c_1‘ and ‘c_2‘ represents the cognitive and social acceleration. In this case, these values are fuzzy because they are changing dynamically when the FPSO is running, and r_{1j} (t), r_{2j} ~U(0,1) are random values in the range [0,1]. We are adding a fuzzy system called ‘fuzzypso’ that is able of change the ‘c_1‘ and ‘c_2‘ rate.

6.2 *FGA (Fuzzy Genetic Algorithm)*

This section presents a brief description of the FGA model. Several crossover operators have been developed for GAs, depending on the format in which individuals are represented. For binary representations, uniform crossover, one point crossover and two point crossover are the most popular. In this case we are using two points crossover with fuzzy crossover rate because we are adding a fuzzy system called ‘fuzzyga’ that is able of change the crossover and mutation rate.

6.3 Definition of the Fuzzy Systems used in FPSO+FGA

‘fuzzypso’: In this case we are using a fuzzy system called ‘fuzzypso’, and the structure of this fuzzy system is as follows:
Number of Inputs: 2
Number of Outputs: 2
Number of membership functions: 3

Type of the membership functions: Triangular and Gaussian
Number of rules: 9
Defuzzification: Centroid
The main function of the fuzzy system called 'fuzzypso' is to adjust the parameters of the PSO. In this case, we are adjusting the following parameters: 'c_1' and 'c_2'; where:
'c_1' = Cognitive Acceleration
'c_2' = Social Acceleration
We are changing these parameters to test the proposed method. In this case, with 'fuzzypso' is possible to adjust in real time the 2 parameters that belong to the PSO.
'fuzzyga': In this case we are using a fuzzy system called **'fuzzyga'**, the structure of this fuzzy system is as follows:
Number of Inputs: 2
Number of Outputs: 2
Number of membership functions: 3
Type of membership functions: Triangular and Gaussian
Number of rules: 9
Defuzzification: Centroid
The main function of the fuzzy system called 'fuzzypso' is to adjust the parameters of the GA. In this case, we are adjusting the following parameters: 'mu', 'cr'; where:
'mu' = mutation
'cr' = crossover
'fuzzymain': In this case, we are using a fuzzy system called **'fuzzymain'**. The structure of this fuzzy system is as follows:
Number of Inputs: 2
Number of Outputs: 1
Number of membership functions: 3
Type of membership functions: Triangular
Number of rules: 9
Defuzzification: Centroid
The main function of the fuzzy system, called 'fuzzymain' is to decide on the best way for solving the problem, in other words if it is more reliable to use the FPSO or FGA. This fuzzy system is able to receive two inputs, called error and derror, it is to evaluate the results that are generated by FPSO and FGA in the last step of the algorithm. Fig. 4 shows the membership functions of the main fuzzy system that is implemented in this method. All fuzzy system consists of 9 rules. For example, one rule is if error is P and DError is P then best value is P (view Fig. 5). Fig. 6 shows the fuzzy system rule viewer. Fig. 7 shows the surface corresponding to the fuzzy main system. The other two fuzzy systems are similar to the main fuzzy system.

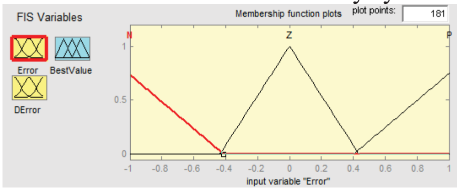

Figure 4: Fuzzy system membership functions

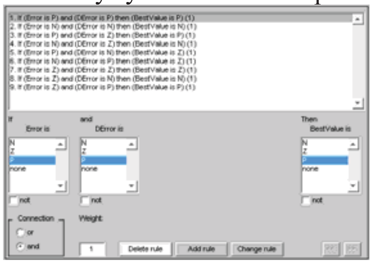
1. If (Error is P) and (DError is P) then (BestValue is P) (1)
2. If (Error is N) and (DError is N) then (BestValue is N) (1)
3. If (Error is P) and (DError is Z) then (BestValue is P) (1)
4. If (Error is N) and (DError is Z) then (BestValue is N) (1)
5. If (Error is P) and (DError is N) then (BestValue is Z) (1)
6. If (Error is N) and (DError is P) then (BestValue is Z) (1)
7. If (Error is Z) and (DError is Z) then (BestValue is Z) (1)
8. If (Error is Z) and (DError is N) then (BestValue is N) (1)
9. If (Error is Z) and (DError is P) then (BestValue is P) (1)

Figure 5: Fuzzy system rules

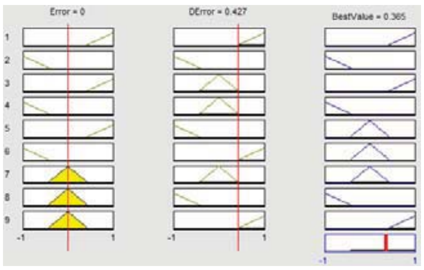

Figure 6: Fuzzy system rules viewer

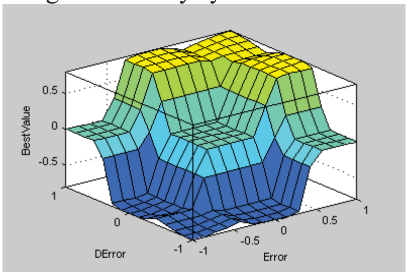

Figure 7: Surface of fuzzy system

7. Simulation Results for Modular Neural Network for Optimization

Several tests of the FPSO+FGA method for MNN optimization were made in the Matlab programming language.

All the implementations were developed using a computer with processor Intel core 2 quad of 64 bits that works to a frequency of clock of 2.5 GHz, 4 GB of RAM Memory and Windows Vista operating system.

We describe below simulation results of our approach for face recognition with modular neural networks (MNNs). We used two-layer feed-forward MNNs with the Conjugated-Gradient training algorithm [4]. The challenge is to find the optimal architecture of this type of neural network, which means finding out the optimal number of layers and nodes of the neural network [6]. We are using the Yale face database [20] that contains 165 grayscale images in GIF format of 15 individuals, for this paper only 10 subjects were used for training the MNN. There are 5 images per subject, one per different facial expression: center-light, happy, left-light, normal and right-light. In total 50 images were used (view Fig. 8). Three images per subject were used for training the MNN and the other two for the recognition. Regarding the genetic algorithm for NN evolution, we used a hierarchical chromosome for representing the relevant information of the network. First, we have the bits for representing the number of layers of the MNN; in this case, the initial topology was of 3 modules with 2 layers per module with 500 neurons in the first layer, 300 neurons in the second layer in each module. Therefore we used a representation the 2415 bits in total (view Fig. 9). The PSO is organized in a similar fashion, but there is less number of parameters. In Fig. 10 we can see the architecture of a MNN that we are using with the evolutionary proposed method FPSO+FGA.

The fitness function used in this case for the MNN combines the information of the error objective and also the information about the number of nodes as a second objective. This is shown in the following equation.

$$f(z)=\left(\frac{1}{\alpha * Ranking(ObjV1)+\beta * ObjV2}\right)*10 \qquad (1)$$

The first objective is basically the average sum of squared of errors as calculated by the predicted outputs of the MNN compared with real values of the function. This is given by the following equation.

$$f_1 = \frac{1}{N}\sum_{i=1}^{N}(Y_i - y_i)^2 \qquad (2)$$

The second objective is the complexity of the neural network, which is measured by the total number of nodes in the architecture.

The final topology of the neural network for the problem of face recognition is obtained by the hybrid evolutionary method FPSO+FGA. The comparison of the final objective values (errors) will be shown in the following section. In the final architecture, the result of the MNN evolution is a particular architecture with different number of nodes by layers. Several tests were made; we obtained optimized different architectures for this Modular Neural Network; the best architecture obtained was the following:

Layers = 2 x module

NNL1M1 = 90, NNL2M1 = 50. NNL1M2 = 100, NNL2M2 = 150. NNL1M3 = 70, NNL2M3 = 90. Total bits = 565

Where: NNL1M1= Number of neurons of the layer 1 in module 1. NNL1M1= Number of neurons of the layer 1 in module 1. NNL2M1= Number of neurons of the layer 2 in module 1. NNL1M2= Number of neurons of the layer 1 in module 2. NNL2M2= Number of neurons of the layer 2 in module 2. NNL1M3= Number of neurons of the layer 1 in module 3. NNL2M3= Number of neurons of the layer 2 in module 3. We can see in the Fig. 11 the binary representation for this optimized architecture. With this final topology the Neural Network was trained and the ten images were recognized. It can be seen in Table 1 the different architectures obtained with this method using a ‘fuzzyga’ and a ‘fuzzypso’ with triangular membership functions and a ‘fuzzymain’ with triangular membership functions, the other parameters to the 3 fuzzy systems are above described. Table 2, shows the simulation results with all fuzzy systems with gaussian membership functions. The proposed method optimizes the initial architecture proposed for the problem of face recognition.

Figure 8: Images of the Yale face database

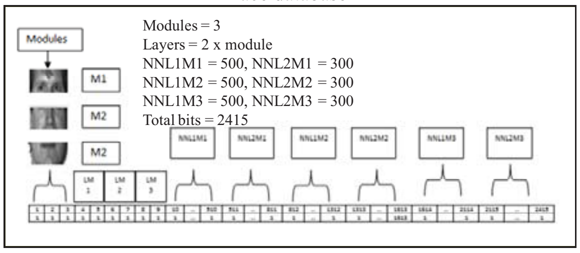

Figure 9: Binary representation for FPSO+FGA (No optimized)

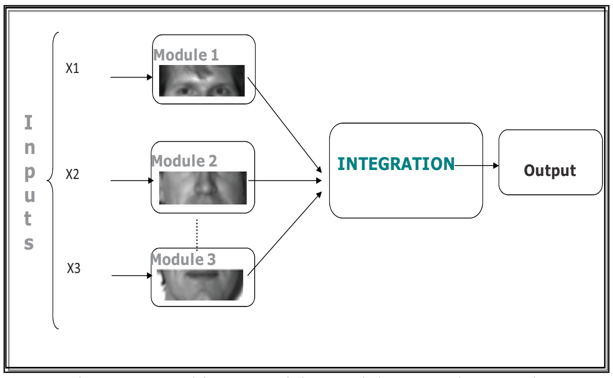

Figure 10: Architecture of the Modular Neural Network.

The Parameters of Table 1 and 2, are as follows:

LMod = Layers per module

NNL = Number of neurons Layer 1, 2 or 3.

GE = Goal Error.

RE = Reached Error.

IDENT = Number of images recognized.

VAR = Number of variables for the mathematical function.

It can be seen in Table 1 and Table 2 that this method is good alternative to solve this type of problems, in our case to optimize Modular Neural Networks. Also this FPSO+FGA have been applied, for optimization of complex mathematical functions to validate our approach. The parameters in FPSO+FGA; as crossover, mutation, cognitive acceleration and social are fuzzy, the population size was of 100 individuals, Table 3 shows the simulation results with 7 mathematical functions to testing this hybrid evolutionary method. The mathematical functions are evaluated with 2, 4, 8 and 16 variables.

Table 1. Simulation Results for the MNN with triangular membership functions

LMod	NN L1 M1	NN L2 M1	NN L1 M2	NN L2 M2	NN L1 M3	NN L2 M3	GE	RE	IDENT
2	20	60	80	50	60	120	0.01	0.03	8
2	90	50	100	150	70	90	0.01	0.005	10
2	70	40	80	40	90	30	0.01	0.02	8
2	150	135	200	90	84	40	0.01	0.003	9
2	100	120	100	145	100	70	0.01	0.001	10

Table 2. Simulation Results for the MNN with gaussian membership functions

LMod	NN L1 M1	NN L2 M1	NN L1 M2	NN L2 M2	NN L1 M3	NN L2 M3	GE	RE	IDENT
2	30	50	90	70	50	115	0.01	0.05	8
2	85	60	103	140	80	130	0.01	0.02	9
2	90	50	95	50	85	40	0.01	0.08	8
2	130	120	180	90	70	50	0.01	0.02	9
2	50	90	75	70	40	40	0.01	0.01	9

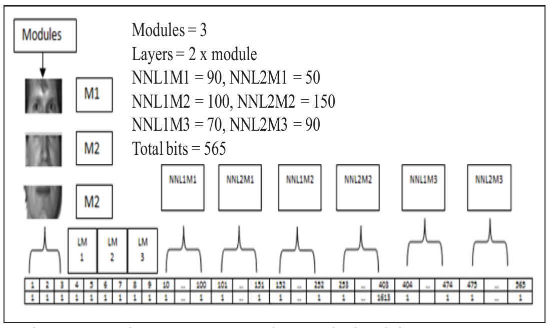

Figure 11: Binary representation optimized for FPSO+FGA

Table 3. Simulation results for Mathematical Functions

Math F	NVar = 2		NVar = 4		NVar = 8		NVAR = 16	
Math	BEST	MEAN	BEST	MEAN	BEST	MEAN	BEST	MEAN
Ras	1.45E-06	3.05E-04	0.0003	0.0755	0.01318	0.9311	0.5483	5.0946
Ros	1.17E-02	1.17E-02	0.0285	0.5991	0.15800	3.8925	0.2555	4.33334
Ack	8.42E-04	4.98E-03	8.42e-01	4.98E-02	0.7	1.56	2.35	2.63
Sph	5.75E-11	1.05E-10	1.946e-05	4.5109e-004	0.00059	0.0057	0.00248	0.0211
Gri	7.88E-11	1.07E-07	7.18e-06	1.1182e-004	0.00016	9.299e-004	0.00040	0.0043
Mich	-1.8010	-1.8201	-1.80129	-1.8002	-1.80130	-1.8005	-1.801301	-1.8004
Zak	6.00E-07	0.00168	3.237e-07	8.4129e-005	1.3308e-07	6.4901e-005	8.63410e-07	7.0065e-005

In Table 4, It can be seen a comparison results between the 3 methods analyzed in this research, we can observe FPSO+FGA as a good alternative to resolve optimization problems. Also, the Neural Network (NN) was included to test this approach. Table 4 shows the simulation results for two variables.

Table 4. Comparison results between GA, PSO and FPSO+FGA

Math Funct	GA	PSO	FPSO+FGA	Objective Value
Ras	2.15E-03	5.47E-05	3.05E-04	0
Ros	1.02E-05	1.97E-03	1.17E-02	0
Ack	2.98	2.98	4.98E-03	0
Sph	1.62E-04	8.26E-11	1.05E-10	0
Grie	2.552E-05	2.56E-02	1.07E-07	0
Mich	-1.7829	-7.44E-01	-1.8201	-1.8013
Zakh	0.00146674	8.10	0.00168	0
NN	3.33E-01	2.3E-01	1.01E-03	0

8 Conclusions

The analysis of the simulation results of the evolutionary method considered in this paper, FPSO+FGA lead us to the conclusion that for the optimization of Modular Neural Networks with this method is a good alternative because it is easier to optimize the architecture of Modular Neural Network than to try it with PSO or GA separately. This is, because the combination PSO and GA with fuzzy rules gives a new hybrid method FPSO+FGA. It can be seen in Table 1 that the second and five architectures obtained after applying FPSO+FGA recognizes the ten images; we are demonstrating that it is reliable uses for this type of applications. Recently we are working with more images for test the effectiveness of this approach. In Table 4 it can be seen a comparison with PSO and GA used separately, we can observed as FPSO+FGA was better.

Acknowledgment

We would like to express our gratitude to CONACYT, Universidad Autónoma de Baja California and Tijuana Institute of Technology for the facilities and resources granted for the development of this research.

References

[1] Man K.F., Tang K.S., and Kwong S., "Genetic Algorithms: Concepts and Designs", Springer Verlag. 1999.

[2] Eberhart, R. C. and Kennedy, J. A new optimizer using particle swarm theory. Proceedings of the Sixth International Symposium on Micromachine and Human Science, Nagoya, Japan. pp. 39-43, 1995.J.-G. Lu, “Title of paper with only the first word capitalized,” *J. Name Stand. Abbrev.*, in press.

[3] Kennedy, J. and Eberhart, R. C. Particle swarm optimization. Proceedings of IEEE International Conference on Neural Networks, Piscataway, NJ. pp. 1942-1948, 1995.

[4] Holland J.H., Adaptation in natural and artificial system, Ann Arbor, The University of Michigan Press, 1975.

[5] Valdez, F. and Melin P. “Parallel Evolutionary Computing using a cluster for Mathematical Function Optimization“, Nafips. San Diego CA USA, 598-602. June 2007.

[6] Castillo, O.; Melin, P. “Hybrid intelligent systems for time series prediction using neural networks, fuzzy logic, and fractal theory” Neural Networks, IEEE Transactions on Volume 13, Issue 6, Page(s): 1395 – 1408. Nov. 2002.

[7] D.B. Fogel, “An introduction to simulated evolutionary optimization”, IEEE transactions on neural networks, vol 5, n 1, Page(s):3 – 14, jan 1994.

[8] Goldberg D., Genetic Algorithms, Addison Wesley, 1988.

[9] Emmeche C., Garden in the Machine. The Emerging Science of Artificial Life, Princeton University Press, pp. 114. 1994.

[10] Angeline P. J., Using Selection to Improve Particle Swarm Optimization , In Proceedings 1998 IEEE World Congress on Computational Intelligence, Anchorage, Alaska, IEEE, 84-89. 1998.

[11] Angeline P.J., Evolutionary Optimization Versus Particle Swarm Optimization: Philosophy and Performance Differences, Evolutionary Programming VII, Lecture Notes in Computer Science 1447, Springer, :601-610. 1998.

[12] Back T., Fogel D.B., and Michalewicz Z., (Eds), Handbook of Evolutionary Computation. Oxford University Press, 1997.

[13] Montiel O, Castillo O, Melin P, Rodriguez A and Sepulveda R: “Human evolutionary model: A new approach to optimization.” Inf. Sci. 177 (10): 2075-2098, 2007.

[14] Castillo O., Valdez F. and Melin P., “Hierarchical Genetic Algorithms for topology optimization in fuzzy control systems“. International Journal of General Systems, Volume 36, Issue 5., pag 575-591, October 2007.

[15] Valdez F., Melin P. and Castillo O. “Evolutionary Computing for the Optimization of Mathematical functions”. Analysis and Design of Intelligent Systems Using Soft Computing Techniques. Advances in Soft Computing 41. June 2007.

[16] Castillo O, Huesca G, and Valdez F. “Proceedings of the International Conference on Artificial” Intelligence, IC-AI '04, Las Vegas, Nevada,USA, Volume 1, pag. 98-104,June 21-24, 2004.

[17] Veeramachaneni K, Osadciw L and Yan W ”Improving Classifier Fusion Using Particle Swarm Optimization”, IEEE Fusion Conference, Italy, July, 2006.

[18] M and Kennedy J. “The particle swarm-explosion, stability, and convergence in a multidimensional complex space”. IEEE Transactions on Evolutionary Computation, 6(1):58-73. 2002.

[19] Kennedy J and Mendes R. “Population structure and particle swarm performance“.Proceeding of IEEE conference on Evolutionary Computation, pag. 1671-1676. 2002.

[20] Wei W, Jiatao S, Zhongzxiu Y and Zheru . “Wavelet-based Illumination Compensation for Face Recognition using Eifenface Method Intelligent Control and Automation”. WCICA 2006. Volume 2, Issue , 21-23 June 2006 Page(s): 10356 – 10360, 2006.

[21] Muller B, Reinhardt J, StricklandM. Neural networks: An introduction. 2nd edn, Springer-Verlag, Berlin. 1995.

[22] McCulloch WS, Pitts W. A logical calculus of the ideas immanent in nervous activity. Bull of Math Biophysics 5:115–133. 1943.