# Fuzzy Systems and Multideme Genetic Algorithm

I.Rojas, H.Pomares, J.González, E.Ros, Mouncef Filali, F.Rojas
Department of Architecture and Computer Technology. University of Granada. Spain.

**Abstract**

In this paper we propose a GA that is capable of simultaneously optimizing the structure of the system and tuning the parameters that define the fuzzy system. For this purpose, we use the concept of multiple-deme GAs, in which several populations with different structures (number of input variables) evolve and compete with other. In each of these populations, the element also has different numbers of membership functions in the input spaces and different numbers of rules. Instead of the normal coding system used to represent a fuzzy system, in which all the parameters are represented in vector form, we have performed coding by means of multidimensional matrices, in which the elements are real-valued numbers, rather than the traditional binary or Gray coding.

Keywords: **fuzzy, genetic algorithm, time-series prediction**

**Tópicos:**

Computación Evolutiva, Algoritmos Genéticos y Redes Neuronales
Aprendizaje Automático, Descubrimiento de Conocimiento y Minería de Datos

PAPER TRACK

# Fuzzy Systems and Multideme Genetic Algorithm

I.Rojas, H.Pomares, J.González, E.Ros, Mouncef Filali, F.Rojas
Department of Architecture and Computer Technology. University of Granada. Spain.

**Abstract**
In this paper we propose a GA that is capable of simultaneously optimizing the structure of the system and tuning the parameters that define the fuzzy system. For this purpose, we use the concept of multiple-deme GAs, in which several populations with different structures (number of input variables) evolve and compete with other. In each of these populations, the element also has different numbers of membership functions in the input spaces and different numbers of rules. Instead of the normal coding system used to represent a fuzzy system, in which all the parameters are represented in vector form, we have performed coding by means of multidimensional matrices, in which the elements are real-valued numbers, rather than the traditional binary or Gray coding.

## I. FUZZY MODELING BY GENETIC ALGORITHMS

Two primary tasks of fuzzy system construction are structure identification and parameter adjustment. The former determines the number of input variables, its number of partition or membership functions, and the number of fuzzy rules. The latter identifies the set of parameters to optima approximate the set of I/O vectors a la given fuzzy structure. The design of a fuzzy system involves the structure of the rules of the system, and the membership function parameters.

To typical examples using the technique of learning from examples ca be found by applying either neural networks or genetic algorithms in the design of the FP's. The former approach has some drawbacks: 1) it can only use numerical data pairs; 2) it does not always guarantee the optimal system performance due to easy trapping to local minimum solution; and 3) it is not easy to interpret the created fuzzy rules due to its internal representation of weights. Designing the FP's based on the GA's has been widely attempted since it can provide more possibilities of finding an optimal (or near-optimal) solution due to the implicit parallelism of GA's.

GAs have the potential to be used to evolve both the fuzzy rules and the corresponding fuzzy set parameters [10]. Some of the work of fuzzy systems and GAs concentrates exclusively on tuning of membership functions [7] or on the selecting an optimal set of fuzzy rules [9], while others attempt to derive rules and membership functions together [3]. To obtain optimal rule sets and optimal sets of membership functions, it is preferable that both are acquired simultaneously [5]. To optimize the whole fuzzy system simultaneously, two structures will be used: one to encode the membership functions and the other for the fuzzy rules.

### *A. Membership function coding*

The membership functions are encoded within an "incomplete" matrix in which each row represents one of the variables of the system, and where the columns encode the parameters of the membership functions. Because each of the input variables of the system has a different number of membership functions, the chromosome structure used to store the membership functions is not a "complete" matrix, as each of the m rows has a different number of columns $n_m$. As we have selected a triangular partition (TP), the only parameter that needs to be stored is the centre of the triangular function [11]. A fuzzy set $X_m^i$ is defined by a linguistic function in the form:

$$\mu_{X_m^i} = \begin{cases} \dfrac{x - c_m^{i-1}}{c_m^i - c_m^{i-1}} & \text{if } c_m^{i-1} < x \leq c_m^i \\ \dfrac{c_m^{i+1} - x}{c_m^{i+1} - c_m^i} & \text{if } c_m^i < x \leq c_m^{i+1} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

where $c_m^i$ represents the centre of the i membership function of the input variable *m*. Thanks to the membership function configurations selected, it is straightforward for a human operator to understand the final fuzzy system obtained. Because the number of membership functions is also optimized it is possible for some of the input variables to be removed, and these would thus have no membership function.

### *B. Fuzzy rules codification*

To encode fuzzy rules, rather than a string or vector where the numerical consequents of the conclusions will appear, we carried out spatial encoding in the form of a $n_1$ x...x $n_n$ matrix, noting that $n_m$ is the number of membership functions contained within each input variable. By using string linear encoding, rules that are close together within the antecedent and which, when fuzzy inference is performed are activated simultaneously, can be distantly encoded. Thus, in a planar structure, the neighborhood properties are destroyed when it is forced into a linear chromosome. In the behaviour of GAs, it is preferable for fuzzy rules that are similar in the antecedent to be encoded as neighbors. Therefore, and as is implicit in encoding, rules that are neighbors in the rule table create interference with each other. Fig.1 shows the complete fuzzy systems codification. Note that the genetic operators described in the following section take into account the spatial structure of the fuzzy rules. Finally, as learning from examples is used, the training data might not cover the whole input domain. This would arise from the huge quantity of data that would be needed, and also from the physical impossibility of obtaining such data. In this case, an incomplete rule base is obtained, and the non-existent rules are encoded in the consequent with a Non A Number (NaN) and thus are not taken into account in the fuzzy inference process.

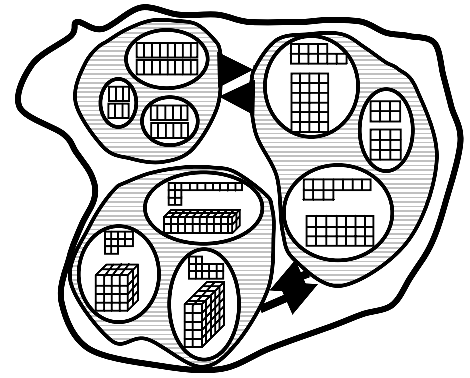

*Fig. 1Codification of a GA population with 3 demes*

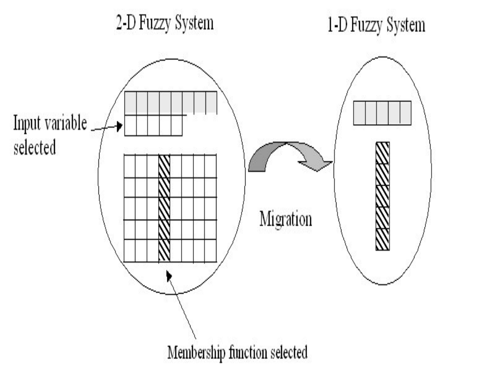


*Fig.2 Migration towards a fuzzy system with a lower dimensionality*

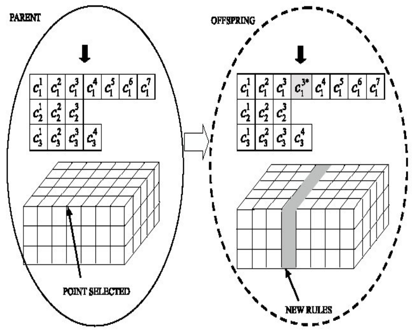


*Fig.3 Increasing the number of membership functions and rules*

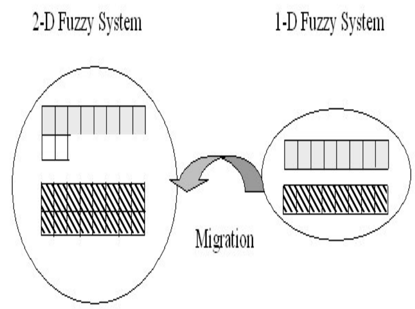


*Fig. 4 Migration towards a fuzzy system with a higher dimensionality*

### *C. Fitness function*

To evaluate the fuzzy system obtained, we have used the error approximation criterion, but to take into account the parsimony principle, that is, the number of parameters to be optimized in the system, we add a new term to describe the complexity of the derived fuzzy system. In the approach presented, GAs are used to search for an optimized subset of rules (both number of rules and the rule values) from a given knowledge base to achieve the goal of minimizing the number of rules used while maintaining the system performance. If we have various models based on the same set of examples, the most appropriate one is determined as that with the lowest description length. Another more flexible alternative is to define the fitness function as a linear combination of the error committed by the system and the number of parameters defining [4]:

$$fitness = W_E \cdot Error + W_C \cdot Complexity \qquad (2)$$

## II. MULTIPLE-POPULATION OR MULTIPLE-DEME GA

The theme of this article is that different structures of fuzzy systems may evolve and compete with each other, in such a way that even information obtained by fuzzy systems with different numbers of input variables may be shared. In general, for identification purposes, no a priori information about the structure of the fuzzy system is always obtained. Even the number of inputs (for example, in time-series prediction problems) is not always known. For this purpose, a multiple-population (or multiple-deme) GA configuration is used [1], in which each deme has a different number of input variables; within each deme there are fuzzy systems with different numbers of membership functions and rules. Basically, the configuration consists of the existence of several sub-populations which occasionally exchange individuals. Therefore it is necessary for there to exist intercommunication between the various demes that comprise the total genetic population. This exchange of individuals is called migration and is controlled by several parameters.

### *A. Migration between neighbour demes*

In this paper, two different situations of migration between demes are considered: the migration towards demes with a lower dimensionality and that towards those with a higher dimensionality. Fig.1 illustrates the case in which the exchange of individuals between demes only occurs between near neighbours, which is equivalent to say that the exchange occurs between fuzzy systems that differ by one in their input space dimensionality. The migration of a fuzzy system with a particular number of input variables towards a system with a lower dimensionality requires the previous, and random, selection of the variable to be suppressed (we term this variable m). The second step is then to determine, again in random fashion, one of the membership functions of this variable (termed j) and to construct the new, lower dimensionality, fuzzy system that only has the rules corresponding to the membership function j that has been selected (Fig.3). Thus the set of membership functions of the new fuzzy system is identical to that of the donor system, except that the variable m has been removed. The rules are determined by the following expression:

$$R^{Offpring}_{i_1 i_2 \ldots, i_{m-1}, i_{m+1} \ldots, i_N} = R_{i_1 i_2 \ldots i_{m-1}, j, i_{m+1} \ldots, i_N}; \; j \in [1, n_m] \qquad (3)$$

In the second case, the new fuzzy system (the new offspring) proceeds from a donor fuzzy system with a lower number of input variables (Fig.4). Here, it is not necessary to determine any donor system input variable, as in the migration described above, because the new offspring is created on the basis of the information obtained from the donor system, with the increase of a new variable; which is randomly selected from the set of variables in the higher dimensionality deme that are different to the deme with lower dimensionality. This new variable initially has a random number of homogeneously distributed membership functions, and its rules are an extension of the donor fuzzy system, taking the form: $R^{Offpring}_{\substack{i_1 i_2 \ldots, i_N, i_{N+1} \\ \forall i_{N+1}}} = R_{i_1 i_2 \ldots, i_N}$

## III. GENETIC OPERATORS.

To perform the crossover of the individuals within the same subsystem, we distinguish between the crossover of the membership functions and that of the rules.

*A. Crossover of the membership functions.*

When two individuals have been selected (which could be termed, i and i') within the same subsystem in order to perform the crossover of the membership functions, the following steps are taken:

1.-One of the input variables of the system (for example, m) is randomly selected.

2.-Let $n_m^i$ and $n_m^{i'}$ be the number of membership functions of system i and i' for the randomly selected variable m. Assume that $n_m^i$ # $n_m^{i'}$. Then from system i we randomly select two crossover points, p1 and p2, such that: 1#p1#p2# $n_m^i$. The membership functions that belong to the interval [p1,p2] of individual i are exchanged for the membership functions of individual i' that occupy the same position.

*B. Crossover of the rules.*

To achieve the crossover of the consequents of the membership functions, we substitute N-dimensional sub matrices within the two individuals selected to carry out the operation. One of the individuals is termed the receptor, R, whose matrix is to be modified, and the other is the donor, D, which will provide a randomly selected sub matrix of itself. The crossover operation consists of selecting a sub matrix S from the rule matrix of the donor individual such that a matrix $S^*$ of equal dimensions and located at the same place within the receptor individual is replaced by the new rules specified by matrix S. In other words, the new offspring O is equal to R except in the sub matrix of the rules given by matrix S, located at the point vector ($A_1$, $A'_1$, $A_{2,...,}A_N$). Therefore, the following steps are taken:

1.- Select two individuals R and D

2.- In order to perform the (N+1) points crossover operator, select a vector ($A_1$,$A'_1$, $A_{2,...,}A_N$), such as the sub matrices S and $S^*$ fulfill $S \subset R$ and $S^* \subset D$.

3.- Create an offspring interchanging the sub matrices S and $S^*$ in R.

*C. Mutation*

In mutation, the parameters of the fuzzy system are altered in a different way from what occurs within a binary-coded system. As the individual is not represented by binary numbers, the random alteration of some of the system's bits does not occur. Instead of this, there are perturbations of the parameters that define the fuzzy system. Firstly, when the fuzzy system that will be mutated has been selected, a parameter defining the fuzzy system (membership functions or rules) is randomly selected with a probability of $P_m$. Secondly, the parameter is modified according to the following expression:

$$
\begin{aligned}
&\mathrm{c}_\mathrm{m}^\mathrm{j} = \mathrm{c}_\mathrm{m}^\mathrm{j} + random(-\Delta \mathrm{c}_\mathrm{m}^\mathrm{j-1}, \Delta \mathrm{c}_\mathrm{m}^\mathrm{j+1}) \\
&\mathrm{R}_{\mathrm{i_1 i_2 \ldots i_N}} = \mathrm{R}_{\mathrm{i_1 i_2 \ldots i_N}} + random(-\Delta R, \Delta R)
\end{aligned}
\tag{4}
$$

where the values that perturb the membership functions are given by $\Delta \mathrm{c}_\mathrm{m}^\mathrm{j-1} = \dfrac{\mathrm{c}_\mathrm{m}^\mathrm{j} - \mathrm{c}_\mathrm{m}^\mathrm{j-1}}{\mathrm{b}}$ and $\Delta \mathrm{c}_\mathrm{m}^\mathrm{j+1} = \dfrac{\mathrm{c}_\mathrm{m}^\mathrm{j+1} - \mathrm{c}_\mathrm{m}^\mathrm{j}}{\mathrm{b}}$. The active radius '*b*' is the maximum variation distance and is used to guarantee that the order of the membership function locations remains unchanged (a typical value is *b*=2, meaning that, at most, a centre can be moved as far as the midpoint between it and its neighbour). The parameter $\Delta R$ is the maximum variation of the conclusion of the rules.

*D. Increasing the number of membership functions*

This makes it possible for all of the information contained in the chromosomes of a fuzzy system to be transferred to another system with greater structural complexity, as the number of membership functions of a particular, randomly-selected, variable increases.

To achieve this, the first step is to select an input variable at random (for example, m) and within this variable to select a position of the membership functions, j, where j0[1,$n_m$]. At this position j, a new membership function will be introduced, such that the order of the previously-defined functions remains unaltered. Thus, the new centre of this function is randomly selected but with the restriction: $c_m^j < c_m^* < c_m^{j+1}$. Thus the new distribution of membership functions for the variable m is now: $\{X_m^1, X_m^2, ..., X_m^j, X_m^*, X_m^{j+1} \ldots, X_m^{n_m}\}$.

With respect to the new rules that have been added, we perform a linear interpolation of the new rules with their immediate neighbours, adding a perturbation. Fig.4 represents the effects of this operator on a fuzzy system.. This is expressed in mathematical terms as:

$$
\begin{aligned}
&R^{Offpring}_{\substack{i_1 i_2 \ldots, i^j_m, \ldots, i_N \\ j\in[1,n_m];\, j\neq j^*}} = R_{i_1 i_2 \ldots i_m, \ldots, i_N} \\
&R^{Offpring}_{i_1 i_2 \ldots, i^{j^*}_m, \ldots, i_N} = \left[\frac{R_{i_1 i_2 \ldots i^j_m, \ldots, i_N} + R_{i_1 i_2 \ldots i^{j+1}_m, \ldots, i_N}}{2}\right] + random(-\Delta R, \Delta R)
\end{aligned}
\qquad (5)
$$

*E. Decreasing the number of membership functions*

A reduction in the number of membership functions within a fuzzy system simply removes, at random, one of the membership functions from a variable that is also randomly selected. As well as the membership function, the rules associated with it are also removed

## IV. SIMULATION RESULTS

We will use time series generated from a differential of difference equation governed by determinism (in which, once the initial value is given, the subsequent states are all determined) . This is the deterministic chaos of a dynamic system. The Mackey-Glass chaotic time series is generated from the following delay differential equation:

$$\frac{dx(t)}{dt} = \frac{ax(t-\tau)}{1+x^{10}(t-\tau)} - bx(t) \qquad (6)$$

Prediction of this time series is recognized as a benchmark for testing various neural-network architectures [8]. When J>17, the equation shows chaotic behaviour. Higher values of J yield higher dimensional chaos. For the sake of comparison with earlier work, we have selected the parameter Δ=j=6 and J=17.

We have considered five input candidates: x[t-24], x[t-18], x[t-12], x[t-6], x[t], to the system and the GAs have to find among them the more important inputs affecting the output x[t+6], taking into account the complexity of the final rule. Therefore, in this example $N^{max}$ is equal to five. We used 4 demes, with 2, 3, 4 and 5 input variables. As a result of predicting 6 steps ahead of the Mackey-Glass time series, the root mean square

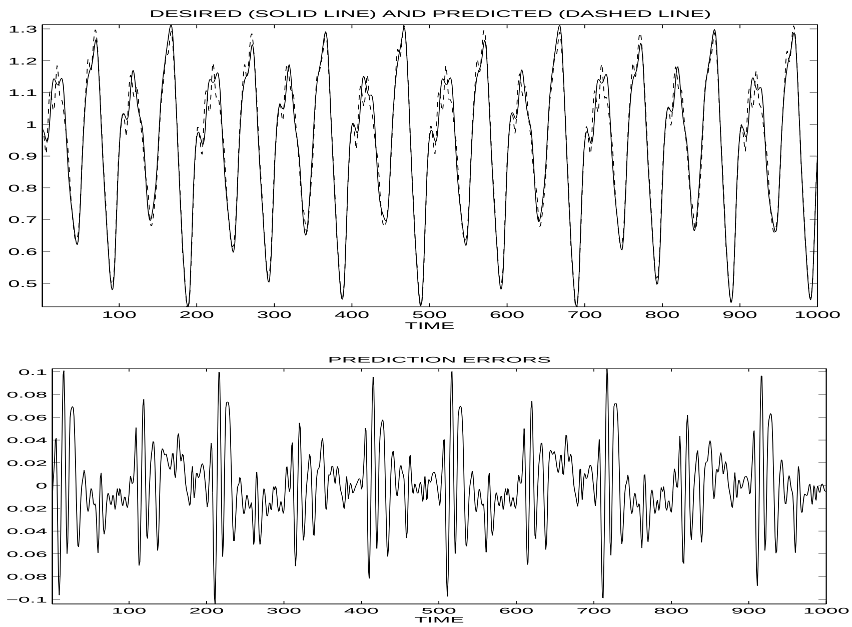


Fig.5 a) Result of the original and predicted Mackey-Glass time series (which are indistinguishable). b) Prediction error

| *Method* | | *Prediction Error (RMSE)* |
|---|---|---|
| Auto Regressive Model | | 0.19 |
| Cascade Correlation NN | | 0.06 |
| Back-Prop. NN | | 0.02 |
| Classical RBF (with 23 neurons) [2] | | 0.0114 |
| $6^{th}$-order Polynomial | | 0.04 |
| Linear Predictive Method | | 0.55 |
| Kim and Kim (Genetic Algorithm & Fuzzy System) [8] | 5 MFs | 0.049 |
| | 7 MFs | 0.042 |
| | 9 MFs | 0.038 |
| ANFIS & Fuzzy System [6] | | 0.007 |
| Wang et al. [12] | Product T-norm | 0.0907 |
| | Min T-norm | 0.0904 |
| Our approach, using only 3 input variables with 5, 6 and 8 membership functions | | 0.032 |

Table 1: Comparison results of the prediction error of different methods for

error and the correlation coefficient are 0.032 and 0.98. Fig.5.a shows the predicted and desired values (dashed and continuous lines respectively) for both training and checking data (which is indistinguishable from the time series here). As they are practically identical, the difference can only be seen on a finer scale (Fig.5.b). Table 1 compares the prediction accuracy of different computational paradigms presented in the bibliography for this benchmark problem (including our proposal), for various fuzzy system structures, neural systems and genetic algorithms.

## V. CONCLUSIONS

While the bibliography describes many methods that have been developed for the adjustment or fine-tuning of the parameters of a fuzzy system with partially or totally known structures, few have been dedicated to achieving both simultaneous and joint structure and parameter adjustment. The goal of this research is to design an optimal fuzzy system than can extract fuzzy rules and specify membership functions automatically by learning from examples. This method has the merits that it does not require both the precise mathematical modeling of fuzzy system and the human expert's help since the input-output characteristics of fuzzy system are approximated by learning the training examples. In this article, a real-coded genetic algorithm (GA) is proposed capable of simultaneously optimizing the structure of a system (number of inputs, membership functions and rules) and tuning the parameters that define the fuzzy system. A multideme GA system is used in which various fuzzy systems with different numbers of input variables and with different structures are jointly optimized. Communication between the different demes is established by the migration of individuals presenting a difference in the dimensionality of the input space of a particular variable. We also propose coding by means of multidimensional matrices of the fuzzy rules such that the neighborhood properties are not destroyed by forcing it into a linear chromosome.

**Acknowledgement**

This work has been partially supported by the CICYT Spanish Project DPI2001-3219.

## REFERENCES

[1] E.Cantú-Paz, "Topologies, migration rates, and multi-population parallel genetic algorithms". Proceedings of the Genetic and Evolutionary Computation Conference. San Francisco, CA: Morgan Kaufmann. pp. 91-98, 1999.

[2] K.B.Cho, B.H.Wang "Radial basis function based adaptive fuzzy systems and their applications to system identification and prediction", Fuzzy Sets and Systems, vol.83, pp.325-339, 1995.

[3] A. Homaifar, E. McCormick, "Simultaneous design of membership functions and rule sets for fuzzy controllers using genetic algorithms", IEEE Trans. Fuzzy Systems, 3, pp.129-139, 1995.

[4] H. Ishibuchi, T.Murata, I.B.Türksen, "A genetic-algorithm-based approach to the selection of linguistic classification rules", Proc. EUFIT'95, Aachen, Germany, pp.1414-1419, 1995.

[5] I.Jagielska, C. Matthews, T. Whitfort, "An investigation into the application of neural networks, fuzzy logic, genetic algorithms, and rough sets to automated knowledge acquisition for classification problems" *Neurocomputing*, vol.24 pp.37-54, 1999.

[6] J.S.R.Jang, C.T.Sun, E.Mizutani, "Neuro-Fuzzy and soft computing", Prentice Hall, ISBN 0-13-261066-3, 1997.

[7] C. Karr, E. Gentry, “Fuzzy control of pH using genetic algorithms”, IEE Trans. Fuzzy Systems, 1, pp.46-53, 1993
[8] D.Kim, C. Kim, "Forecasting time series with genetic fuzzy predictor ensemble", IEEE Transactions on Fuzzy Systems, vol.5, no.4, pp.523-535, November 1997.
[9] K.Kropp, U.G.Baitinger, "Optimization of fuzzy logic controller inference rules using a genetic algorithm", Proc. EUFIT'93, Aachen, Germany, pp.1090-1096, 1993.
[10] J.Liska, S.S.Melsheimer, "Complete design of fuzzy logic systems using genetic algorithm", Proc. FUZZ-IEEE'94, pp.1377-1382, 1994.
[11] I.Rojas, J.J. Merelo, H.Pomares, A.Prieto "Genetic algorithms for optimum designing fuzzy controller", *Proceedings of the 2nd. Int. ICSC Symposium on Fuzzy Logic and Applications (ISFL'97), International Computer Science Conventions*, February 12-14, Zurich, ICSC Academic Press, Canada, ISBN 3-906454-03-7, pp. 165- 170, 1997.
[12] L.X.Wang, J.M.Mendel, "Generating fuzzy rules by learning from examples", IEEE Trans. On Syst. Man and Cyber, vol.22, no.6, November/December, pp.1414-1427, 1992.