MOGUL: A Methodology to Obtain Genetic Fuzzy Rule-Based Systems under the Iterative Rule Learning Approach*

O. Cordón[1,†,‡], M. J. del Jesus[2,§], F. Herrera[1,†], M. Lozano[1,†]
[1] *Department of Computer Science and Artificial Intelligence, E.T.S. de Ingeniería Informática, University of Granada, 18071-Granada, Spain*
[2] *Department of Computer Science, Escuela Politécnica Superior, University of Jaén, Jaén, Spain*

The main aim of this paper is to present *MOGUL*, a Methodology to Obtain Genetic fuzzy rule-based systems Under the iterative rule Learning approach. MOGUL will consist of some design guidelines that allow us to obtain different genetic fuzzy rule-based systems, i.e., evolutionary algorithm-based processes to automatically design fuzzy rule-based systems by learning and/or tuning the fuzzy rule base, following the same generic structure and able to cope with problems of a different nature. A specific evolutionary learning process obtained from the paradigm proposed to design unconstrained approximate Mamdani-type fuzzy rule-based systems will be introduced, and its accuracy in the solving of a real-world electrical engineering problem will be analyzed. © 1999 John Wiley & Sons, Inc.

1. INTRODUCTION

Nowadays, the most important applications of fuzzy set theory as developed by Zadeh in 1965[1] are fuzzy rule-based systems (FRBSs). These kinds of systems constitute an extension of classical rule-based systems, because they deal with fuzzy rules instead of classical logic rules. Thanks to this, they have been successfully applied to a wide range of problems from different areas presenting uncertainty and vagueness in different ways.[2–5]

An FRBS presents two main components: (1) the inference system, which puts into effect the fuzzy inference process needed to obtain an output from the FRBS when an input is specified, and (2) the fuzzy rule base (FRB) representing

* This research has been supported by CICYT TIC96-0778
† e-mail: ocordon, herrera, lozano@decsai.ugr.es
‡ Author to whom correspondence should be addressed.
§ e-mail: mjj@apolo.ujaen.es

INTERNATIONAL JOURNAL OF INTELLIGENT SYSTEMS, VOL. 14, 1123–1153 (1999)
 CCC 0884-8173/99/111123-31

the knowledge known about the problem being solved, constituted by a collection of fuzzy rules.

Two main tasks have to be performed to design an intelligent system of this kind for a concrete application: to select the fuzzy operators involved in the inference system, i.e., to define the way in which the fuzzy inference process will be performed, and to derive an adequate FRB about the problem being solved. The accuracy of the FRBS in solving this problem will depend directly on both components.

The first design task has been widely analyzed in the specialized literature, and a large quantity of theoretical and comparative studies have been carried out in order to deal with the problem of the selection of the best possible fuzzy operators in the inference system.[6–9]

As regards to the second design task, it seems to be a more difficult decision because the composition of the FRB depends directly on the problem being solved. Due to the complexity of the FRB derivation, a large quantity of automatic techniques have been proposed to put it into effect. In the last few years, many different approaches have been presented taking evolutionary algorithms (EAs),[10] usually genetic algorithms (GAs),[11] as a base, constituting the so called Genetic Fuzzy Rule-Based Systems (GFRBSs).[12,13] These kind of systems are considered nowadays as an important branch of the Soft Computing area[14] in view of the large number of contributions developed in the last few years (see Ref. 15, Section 3.13, Ref. 16, Section 13).

GFRBSs are based on combining the main feature of the FRBSs, interpolative reasoning, a consequence of the cooperation between the fuzzy rules composing the FRB, and of the EAs, the competition induced between the population members to get the best possible solution to the problem. Obtaining the best possible cooperation level in the FRB by inducing competition by means of the EA is referred to as the cooperation vs. competition problem (CCP).[17] All the GFRBSs have to deal with this problem to design accurate FRBSs.

In this paper, we present *MOGUL*, a Methodology to Obtain Genetic fuzzy rule-based systems Under the iterative rule Learning approach. This methodology is composed of some design guidelines that, when assumed, will allow us to obtain GFRBSs to design different types of FRBSs able to cope with problems presenting different characteristics. The GFRBSs obtained following the paradigm proposed will allow us to derive the whole FRB, that is, the fuzzy rules themselves and the membership functions involved in them, when a set of input-output data pairs about the problem being solved is available.

MOGUL will take a generic structure composed of different stages as a base, with the aim of simplifying the search space tackled by the EA. To do so, it will consider the iterative rule learning (IRL) approach,[18] which is based on decomposing the learning process into different stages, and, therefore, the FRBS evolutionary design processes obtained following it will be multi-stage GFRBSs. We will show the performance of one of them by using it to generate some FRBSs to solve an Electrical Engineering problem, comparing it with

classical methods, Neural Networks and other GFRBSs presenting different characteristics.

In order to put this into effect, we arrange this paper as follows. The next section presents some preliminaries by briefly introducing the different types of FRBSs, the GFRBSs, the IRL approach and the CCP. In Section 3, the basis followed by MOGUL are presented. In Sections 4, 5, and 6, the particular aspects related to each one of the three stages composing the GFRBSs obtained from MOGUL are described. In Section 7, a brief analysis is presented on the values of the parameters existing in them. Section 8 shows a specific evolutionary learning process for designing unconstrained approximate Mamdani-type FRBSs, obtained following the MOGUL assumptions. In Section 9, the experiments developed to solve the aforementioned problem are presented. Finally, in Section 10, some concluding remarks are pointed out.

2. PRELIMINARIES

2.1. Types of Fuzzy Rule-Based Systems

There are two different kinds of FRBSs in the literature, the Mamdani and TSK ones, depending on the expression of the consequent of the fuzzy rules composing the FRB. While Mamdani-type fuzzy rules consider a linguistic variable in the consequent,[19,20] TSK fuzzy rules are based on representing the consequent as a polynomial function of the inputs.[21] The generic expression of the TSK rules is the following:

$$\text{If } x_1 \text{ is } A_1 \text{ and } \ldots \text{ and } x_n \text{ is } A_n \qquad \text{then } y = p_1 \cdot x_1 + \cdots + p_n \cdot x_n + p_0$$

Focusing on the other system type, the FRB is composed of a collection of fuzzy rules with the following structure:

$$R_i\text{: If } x_1 \text{ is } A_{i1} \text{ and } \ldots \text{ and } x_n \text{ is } A_{in} \text{ then } y \text{ is } B_i$$

where $x_1, \ldots, x_n$ and y are the input variables and the output variable, respectively. Depending on the characteristics of these fuzzy rules, we can consider two different Mamdani-type FRBSs:

- On the one hand, we have the usual *descriptive* approach[19,20] when $x_1, \ldots, x_n$ and y are linguistic variables that have a term set of possible values associated presenting a real-world meaning. In this way, each A_{ij} or B_i corresponds to a linguistic term that has associated a fuzzy set defining its meaning and this mapping is uniform for all rules in the FRB. This FRBS has been widely used and has obtained very good results in many different applications. Anyway, it suffers some limitations due to the inflexibility of the concept of the linguistic variable.[22] The homogeneous partitioning of the input and output spaces when the input-output mapping varies in complexity within the space is inefficient and does not scale to high-dimensional spaces.[23] Therefore, its performance decreases when dealing with complex problems in which small changes in the input have strong changes associated in the output.[24]

- On the other hand, in the last few years a new approach, the *approximate Mamdani-type* FRBS,[2,12] has been proposed for avoiding these drawbacks. It is based on working directly with fuzzy variables in the fuzzy rules. In this case, each fuzzy rule presents its own semantics, i.e., the variables x_j and y, respectively, take a different fuzzy set A_{ij} and B_i as value and not a linguistic term from a global term set. Therefore, it is said that the rules present *free semantics*. According to,[23] the advantage of approximate representation is its expressive power for learning rules which present their own specificity in terms of the fuzzy sets involved in them. This is likely to be of benefit in tackling the course of dimensionality when scaling to multi-dimensional systems. Anyway, its drawback with respect to the descriptive FRBS is the loss of FRB readability. The approximate approach is considered in Refs. 2, 23–32.

2.2. Genetic Fuzzy Rule-Based Systems

EAs, especially GAs, have proven to be a powerful tool for automating the definition of the FRB, since adaptive control, learning, and self-organizative FRBSs can be considered in many cases as optimization or search processes. Their advantages have extended the use of EAs in the development of a wide range of approaches for designing FRBSs over the last few years. These approaches receive the general name of *GFRBSs*.[12,13]

EAs are applied to modify/learn the definition of the membership function shapes and/or the composition of the fuzzy rules in the way shown in Figure 1. Therefore, it is possible to distinguish three different groups of GFRBSs

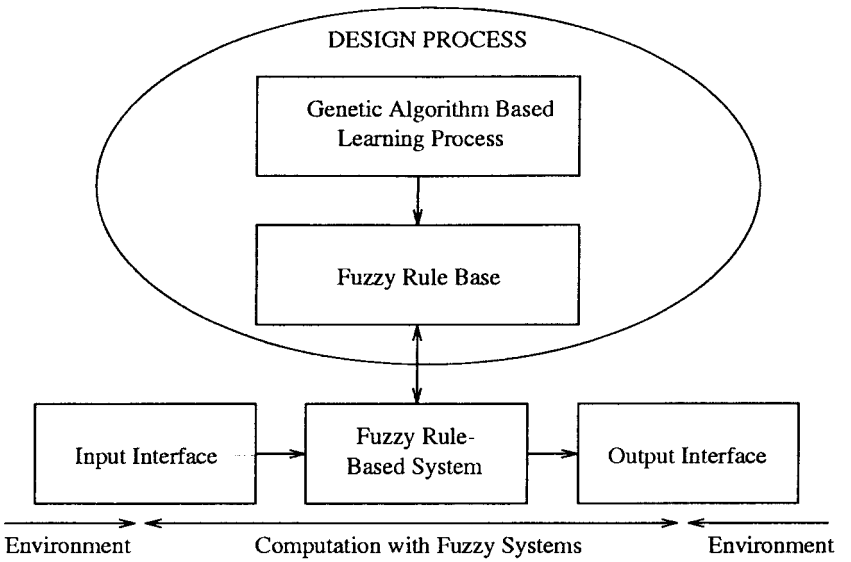

Figure 1. Genetic fuzzy rule-based systems.

depending on the FRB components included in the learning process:[12,13,65]

(1) *Genetic definition of the membership functions*.
(2) *Genetic derivation of the fuzzy rules*.
(3) *Genetic learning of the whole FRB*.

For a wider description of each family see Refs. 12 and 13 and for an extensive bibliography see Ref. 15, Section 3.13, and Ref. 16, Section 13. Different approaches may be found in Refs. 33–35.

Carse et al.[23] divide the third family into two different subgroups depending on the simultaneousness in the learning of both FRB components. Therefore, they differentiate between learning them in a single process or in different stages. We shall refer to the latter kind of systems as multi-stage GFRBSs.[18] The processes obtained from MOGUL belong to this family.

2.3. The Iterative Rule Learning Approach

The main problem that has to be solved to design a GFRBS consists of finding a suitable representation both capable of gathering the problem characteristics and suitably representing the potential solutions to it.

Classically, two genetic learning approaches, adopted from the field of genetic-based machine learning systems, have been used: the *Michigan*[36,37] and *Pittsburgh*[38] approaches. In the Michigan approach, the chromosomes are individual fuzzy rules and the FRB is represented by the entire population. The collection of fuzzy rules is adapted over time using some genetic operators applied at the level of the individual rule. This evolution is guided by a credit assignment system that evaluates the adaption of each single fuzzy rule. On the other hand, in the Pittsburgh approach, each chromosome represents an entire FRB and the evolution is developed by means of genetic operators applied at the level of fuzzy rule sets. The fitness function evaluates the accuracy of the complete FRBS encoded in the chromosome.

In the last few years, the *IRL* approach proposed in Ref. 39 has been used by some authors to obtain several GFRBSs following a new learning model.[18] In this latter model, as in the Michigan one, each chromosome in the population represents a single fuzzy rule, but only the best individual is considered to form part of the final FRB. Therefore, in this approach the EA provides a partial solution to the problem of learning, and, contrary to both previous ones, it is run several times to obtain the complete FRB. This is put into effect by including it in an iterative scheme based on obtaining the best current fuzzy rule for the system, incorporating this rule into the final FRB, and penalizing it before repeating the process. It ends up when the FRB is able to adequately represent the system.

This scheme is usually employed in GFRBSs based on inductive learning, in which the penalization of the fuzzy rules yet generated is made by removing from the training data set all those examples that are still covered by the FRB obtained until that time. On the other hand, as the learning processes using it do not envisage any relationship between the fuzzy rules generated, it is usual to

employ postprocessing to simplify and/or adjust the FRB obtained, thereby forming a multi-stage GFRBS.

A key characteristic of the IRL is that it substantially reduces the search space, because in each iteration only one fuzzy rule is searched. This allows us to obtain good solutions in GFRBSs for off-line learning problems.

A more complete description of the IRL and a short comparison of the three genetic learning approaches is to be found in Refs. 65 and 18.

2.4. The Cooperation vs. Competition Problem

One of the most interesting features of an FRBS is the interpolative reasoning it develops. This characteristic plays a key role in the high performance of FRBSs and is a consequence of the *cooperation among the fuzzy rules composing the FRB*. As is known, the output obtained from an FRBS is not usually due to a single fuzzy rule but to the cooperative action of several fuzzy rules that have been fired, because they match the input to the system to some degree.

On the other hand, the main feature of an EA is the *competition between members of the population representing possible solutions to the problem* being solved. In this case, this characteristic is due to the mechanisms of natural selection on which the EA is based.

Therefore, since a GFRBS combines both features, it works by *inducing competition to achieve the best possible cooperation*. This seems to be a very interesting way to solve the problem of designing an FRBS, because the different members of the population compete between themselves to provide a final solution presenting the best cooperation among the fuzzy rules composing it. The problem is to obtain the best possible way to put this operation mode into effect. This is referred to as *CCP*.[17]

The difficulty of solving the problem introduced depends directly on the genetic learning approach followed by the GFRBS. Multi-stage GFRBSs based on the IRL approach try to properly solve the CCP at the same time as reducing the search space by encoding a single fuzzy rule in each chromosome.[18] To put this into effect, these processes divide the genetic learning process into, at least, two stages. Therefore, the CCP is solved in two steps acting on two different levels:

- *The genetic generation stage forces competition between fuzzy rules*, as the genetic learning processes based on the Michigan approach, *to obtain an FRB composed of the best possible fuzzy rules*. The cooperation among them is only smoothly addressed by means of the rule penalization criterion.
- *The postprocessing stage forces cooperation between the fuzzy rules generated in the previous stage* by refining or eliminating the redundant or unnecessary fuzzy rules from the previously generated fuzzy rule set *in order to obtain the best possible FRB*.

As can be observed, the iterative operation mode followed by the genetic generation stage in multi-stage GFRBSs based on the IRL induces the forma-

tion of niches[40] and substantially reduces the size of the search space. The postprocessing stage deals with a simple search space as well because it only works on the FRB obtained from the previous stage.

An analysis on the way in which the CCP is solved by the other two genetic learning approaches may be found in Refs. 65 and 18.

3. MOGUL: AN EVOLUTIONARY PARADIGM TO DESIGN FUZZY RULE-BASED SYSTEMS

In this Section, we present the design guidelines that constitute the evolutionary methodology proposed, which were briefly introduced in Ref. 41. From a general point of view, MOGUL is based on working with the IRL approach and on some particular aspects considered in order to improve the accuracy of the final FRBS designed from the GFRBSs obtained following these assumptions. MOGUL may be used to generate different types of FRBSs: descriptive and approximate Mamdani-type, and TSK-type ones, to be more precise. This will allow the FRBS designer to obtain the most appropriate solution to the problem being solved.

In the following subsections, we shall briefly analyze the guidelines that constitute MOGUL. By taking into account these guidelines, the designer may obtain different GFRBSs with the same generic structure in different stages but composed of different individual processes. The nature of these specific processes may be defined by him depending on his needs and on the type of FRBS being designed, but the methodology ensures that accurate FRBSs are obtained from the resulting GFRBSs whenever each individual process satisfies the conditions that it imposes for each stage.

3.1. Properties Required for the Generated Fuzzy Rule Base

Several important statistical properties have to be verified by the FRB in order to obtain an FRBS presenting good behavior.[19,20] We will consider the satisfaction of two of them, *completeness* and *consistency*, in the GFRBSs obtained from MOGUL. Since we consider an inductive approach to building GFRBSs, both properties will be based on the existence of a training data set, E_p, composed of p numerical input-output problem variable pairs. These examples will present the following structure:

$$e_l = \left(ex_1^l, \ldots, ex_n^l, ey^l\right), \qquad l = 1, \ldots, p$$

A brief description of the said properties can be found below. For a wider description, refer to Ref. 24.

3.1.1. Completeness of a Fuzzy Rule Base

It is clear that an FRBS should always be able to infer a proper output for every possible system input. This property may be called *τ-completeness* in the field of inductive learning and it may be mathematically formulated using a real

value τ by means of the following expression:[31]

$$C_R(e_l) = \bigcup_{i=1\ldots T} R_i(e_l) \geq \tau, \qquad l = 1, \ldots, p$$

$$\left[R_i(e_l) = *\left(A_i(ex^l), B_i(ey^l)\right)\right]$$

$$A_i(ex^l) = *\left(A_{i1}\left(ex_1^l\right), \ldots, A_{in}\left(ex_n^l\right)\right)$$

where $*$ is a t-norm, and $R_i(e_l)$ is the *compatibility degree* between the rule R_i and the example e_l.

Given an FRB composed of T fuzzy rules R_i, the *covering value* of an example $e_l \in E_p$ is defined as

$$CV_R(e_l) = \sum_{i=1}^{T} R_i(e_l)$$

and we require the following condition

$$CV_R(e_l) \geq \epsilon, \qquad l = 1, \ldots, p$$

A good FRB must satisfy both the conditions presented above, to verify the *τ-completeness property* and to achieve an appropriate final *covering value*.

3.1.2. Consistency of a Fuzzy Rule Base

A generic set of *If–Then* rules is *consistent* if it does not contain contradictions. There is a need to relax the consistency property for considering it in FRBs. We do this by means of the *positive* and *negative example* concepts.[31,42] An example is considered positive for a fuzzy rule when it matches with its antecedent and consequent, and it will be considered a negative example when it matches with its antecedent and not with its consequent.

Let $E^+(R_i) = \{e_l \in E_p / R_i(e_l) \geq 0\}$ and $E^-(R_i) = \{e_l \in E_p / R_i(e_l) = 0$ and $A_i(ex^l) > 0\}$ be respectively the positive and negative example set for the rule R_i. Let $n^+_{R_i} = |E^+(R_i)|$ and $n^-_{R_i} = |E^-(R_i)|$. Given a parameter $k \in [0, 1]$, it is said that[41]

$$R_i \text{ is } k - \textit{consistent} \text{ when } n^-_{R_i} \leq k \cdot n^+_{R_i}$$

Hence, the way to incorporate the satisfaction of this property in the designed GFRBSs is to encourage the generation of k-consistent rules. Those rules not verifying this property will be penalized so as not to allow them to be in the FRB finally generated.

3.2. Guidelines to Solve the Cooperation vs. Competition Problem

In MOGUL, the multistage GFRBS usual way of solving the CCP in two stages,[18,42] introduced in the previous section, will be extended by considering some design criteria collected in several groups according to the learning stage

in which they are considered. We will introduce these criteria in the following subsections.

3.2.1. MOGUL Design Tasks Associated to the Generation Stage

To improve the behavior of the FRBS designed from the GFRBS, MOGUL will consider the following aspects in this stage:

- The designer is allowed to build the generation stage by using different kinds of algorithms and not only a GA as in the previous existing approaches following the IRL approach.[31,39,42] It is possible to employ a non-evolutionary inductive algorithm or an Evolution Strategy (ES)[10] instead of the usual GA. The operation mode is still the same but the difference is the speed of the generation process, which is higher in the latter cases.
- The usual operation mode of multi-stage GFRBSs[18,31,42] does not consider the cooperation between the fuzzy rules generated in the first stage. Each new fuzzy rule is generated without taking into account how it will cooperate with the previous ones obtained. Hence, the newly generated fuzzy rule may interact insufficiently or excessively with the previous ones, making the FRBS obtained perform badly. We will improve the fuzzy rule generation process when designing approximate Mamdani-type or TSK-type FRBSs.
  In the former case, we will consider a phenotypic niching criterion[40] which will allow us to generate the best possible fuzzy rule in each iteration taking into account both the goodness of this rule and the goodness of its cooperation with the previous ones generated. In the latter one, a local error measure will be considered which will promote the generation of fuzzy rules whose consequents will adjust better to the examples best covered by their antecedents. As may be seen, *both criteria allow us to deal with part of the cooperation problem in the first stage*.

3.2.2. MOGUL Design Tasks Associated to the Postprocessing Stage

Usually, the goal of the second learning stage is to improve the cooperation level of the fuzzy rules generated in the previous one by refining them or by removing the redundant or unnecessary ones. With the aim of improving the accuracy of the FRBSs designed, in MOGUL we will tackle both tasks: the simplification of the FRB and the refining of the fuzzy rules composing it, by adjusting their membership functions.

To do so, the postprocessing stage will be broken down into two different stages: the *genetic simplification process* and the *evolutionary tuning process*, as in the GFRBS proposed in Ref. 31. Therefore, the CCP is again divided into smaller subproblems to be solved better. Moreover, other design aspects are considered:

- The postprocessing stage will present two important characteristics. On the one hand, it will be designed by means of a GA based on the Pittsburgh learning approach, the one solving the CCP best, but significatively reducing the solution space by working only over the FRB generated in the first stage, i.e., not modifying the membership function definitions. In this way, it will simplify the RB obtained until now by removing the redundant or unnecessary fuzzy rules not cooperating adequately with the others. This operation mode will allow us to

obtain the best possible FRB composed of the best combination of the fuzzy rules generated in the first stage.
On the other hand, the other existing type of niching, the genotypic one,[40] will be considered to obtain not only a single FRB as output from the process but different ones presenting the best possible behavior. Due to this, we will refer to this second stage as the *multisimplification process*. A similar idea has been used in Ref. 43.

- The third stage will be composed of an EA that will again deal with a smaller search space because it will work only with the membership functions and not with the fuzzy rule structure.
  In this way, the evolutionary tuning process will be applied over the different FRB definitions generated in the previous process, and the most accurate will be the one given as the output of the multi-stage GFRBS. Therefore, an FRB not presenting the best behavior after the second stage, may be the best one after the third one due to the fact that the new membership function shapes make its rules cooperate in a better way.
  Working in this way we directly solve the problem presented in Ref. 43 to choose the best fuzzy model from the ones generated.

3.2.3. MOGUL Design Tasks Associated to the Composition of the Evolutionary Algorithms Considered

Finally, focusing on the EA search, we need to make use of suitable techniques to develop an accurate trek on the search spaces tackled in each stage to obtain the best possible solutions. Several factors have to be considered in order to reduce the search space complexity and to perform an adequate exploration and exploitation over it to allow the search process to be effective. A good analysis of these factors in GFRBS design is presented in Refs. 13 and 44. Amongst the techniques usually employed in genetic learning processes (as well as in other genetic processes) we will consider the following ones:

- *Choosing an adequate representation of the individuals*: It will depend on the learning stage and on the information encoded in the individual. In all cases, we want it to encode as much information as possible in individuals with a length as short as possible. In MOGUL, we will recommend the use of integer coding when representing linguistic labels, real coding, when representing fuzzy membership functions, and we shall propose a specific coding scheme, *angular coding* (see Section 4.1.4), to code the TSK rule consequent parameters. On the other hand, we will combine different schemes when the individual encodes different types of information.
- *Designing specific operators to perform a robust trek on the search space*: We will consider evolutionary operators specifically designed according to the coding scheme chosen, with a suitable exploration-exploitation rate. In this way, we will pay special attention to real-coded genetic operators[11,45,46] and we will propose the inclusion of an (1 + 1)-ES as another genetic operator to promote the exploitation of the best solutions found in each generation (see Refs. 25, 27, 28, and 47), following the assumptions of the so-called *genetic local search*.[48,49]
- *Designing an adequate fitness function*: It will depend on the learning stage and we will promote the use of multicriteria fitness functions, capable of adequately guiding the search over the space. In the generation stage, different frequentistic example covering criteria will be considered, allowing the generated FRB to verify the properties mentioned in Section 3.1. In the remaining stages, these kind

of criteria will be combined with global error measures to improve the cooperation between the rules, maintaining the satisfaction of the properties.
When it is not possible to work with frequentistic criteria, i.e., when designing TSK FRBSs (due to the non-fuzzy nature of the rule consequent), error measures will be used. In the first stage, a local error measure will be considered to deal with part of the cooperation problem, at the same time as inducing competition, while in the remaining ones the use of global error measures will allow us to improve the FRB cooperation level.

- *Deciding the way in which the available knowledge will be used in the learning process*: We will consider two different ways of incorporating this knowledge into the learning process:

  — Incorporating partial definitions obtained from expert knowledge into the learning process (see Section 3.4).
  — Using the available knowledge to generate the initial population of the EAs. In the first stage, the example set will be used directly to generate the initial population. In the remaining ones, the definitions obtained in the previous process will be encoded on one of the initial population individuals. Finally, in the evolutionary tuning process, the initial definitions of the fuzzy partitions will be used to define the intervals of adjustment of the membership function parameters.

- *Defining the niche scheme*: In MOGUL, two different possibilities are considered for this aspect. On the one hand, we use a phenotypic sharing scheme to deal with part of the cooperation problem in the generation stage when designing approximate Mamdani-type FRBSs. On the other hand, we will consider the other existing niche scheme, the genotypic one, to generate different FRB definitions in the multisimplification stage (see Section 5).

3.3. Structure of the Multistage Genetic Fuzzy Rule-Based System Obtained

In view of the ideas mentioned in the previous subsection, the generic structure of the multistage GFRBSs obtained from MOGUL will be composed of the following three stages:

(1) *A fuzzy rule generation process* that allows us to generate a set of fuzzy rules of any kind representing the knowledge existing in the training data set in a suitable form. In all cases, it will present two components: a *fuzzy rule generating method*, whose composition depends on the type of FRBS being designed (see Section 4.1), and an *iterative covering method* of the example set. The latter puts into effect the usual way genetic learning processes work based on the IRL approach. It penalizes each rule generated by the fuzzy rule generating method by considering its covering over the examples in the training set and removes the ones already covered from it.
(2) *A genetic multisimplification process* for selecting rules, in the case in which it is necessary to simplify the FRB obtained in the previous process. This second process is based on a binary coded GA with a genotypic sharing function and a fitness function based on two criteria, a global error measure and a criterion penalizing the non-satisfaction of the τ-completeness property. This process will obtain different simplified FRB definitions thanks to the genotypic niching scheme.

(3) *An evolutionary tuning process*, based on any kind of real coded EA and on a fitness function like the one used in the previous process. It will give the final FRB as output by adjusting the membership functions in each possible FRB obtained from the genetic multisimplification process. The type of tuning performed will depend on the nature of the FRBS being designed. Therefore, when generating a descriptive FRB, a global tuning of the fuzzy partition associated to each linguistic variable will be performed, but when working with an approximate approach, the membership functions involved in each fuzzy rule will be adjusted individually. On the other hand, in the case of a TSK FRB, the antecedent part will be adjusted in the same way as in the descriptive approach, and the preliminary definition of the consequent parameters obtained in the first stage will be refined as well. The most accurate FRB obtained in this stage will constitute the final output of the whole learning process.

Although this will be the generic structure that will usually have the GFRBSs obtained from MOGUL, it is possible that they will only be composed of two stages: a fuzzy rule generation process and a tuning process. This alternative structure will be used in the case in which the FRB generated in the first stage does not need to be simplified due to the fact that this process has been able to obtain an adequate cooperation level, even working at the level of individual rules following the IRL approach. This will ensure that the tuning process will only be employed to refine the cooperation between the rules generated in the first stage and that we do not finally obtain different definitions of the FRB from the GFRBS. An example of this may be found in the TSK GFRBS proposed in Ref. 47.

3.4. Use of the Available Knowledge in the Learning Process

One of the main characteristics of the FRBSs is that they are able to incorporate two different types of information, expert and numerical, into the design process.[50] Therefore, using MOGUL we have to be able to incorporate both kinds of information into the GFRBS obtained, in the case in which they are both available, as well as generating the FRB when we only have a training data set describing the problem and not any kind of expert information.

Hence, the GFRBSs obtained from MOGUL may work in different ways, depending on the kind and on the amount of information available. We are going to describe the different existing possibilities beginning with the case in which less information is available, only an example set, and finishing with the case in which we have enough expert knowledge to generate a whole preliminary FRB definition from it:

(1) *Example set*: The GFRBS is totally applied. To do so, in the case in which a descriptive Mamdani-type or a TSK-type FRBS is being designed, the input and output spaces, in the first case, and the input one, in the second, are first uniformly fuzzy partitioned into a number of linguistic terms.
(2) *Expert fuzzy partitions* (*linguistic terms*, *with their associated membership functions*) *and example set*: When initial fuzzy partitions are provided for all the linguistic variables of the system, the overall learning process is applied, as in the previous case, but the expert definitions are used instead of the uniformly obtained ones.

(3) *Partial FRB and example set* (*either with or without expert fuzzy partitions*): First, the GFRBS incorporates the expert fuzzy rules to the FRB to be generated in the first stage, independently of its kind (in the case in which a TSK FRBS is considered, the linguistic rule is transformed into a TSK one by taking the modal point of the consequent fuzzy set).[50] Then, this partial FRB is completed by blending it with the information obtained from the generation process (fuzzy rules learned from input-output data), then applying the genetic multisimplification process to obtain different simplified FRB definitions. Finally, the evolutionary tuning process is applied to adjust the membership functions, obtaining the final FRB.

(4) *Preliminary FRB and example set*: When a complete FRB has been derived from an expert, the third component, the evolutionary tuning process, may be used individually to obtain a more accurate FRBS by adjusting the previous definition of the membership functions.

4. THE FUZZY RULE GENERATION PROCESS

As we have seen in Section 3.3, the fuzzy rule generation process is composed of the *fuzzy rule generating method* and of the *iterative covering method*. We will devote the following two subsections to go deeper into the analysis of the composition of both methods.

4.1. The Fuzzy Rule Generating Method

This first method obtains the best fuzzy rule at each moment according to the training data set state. As mentioned, its composition will depend on the type of fuzzy rule considered. The following possibilities exist:

- *Design of descriptive Mamdani-type FRBSs*: In this case, the first stage works only with the linguistic labels and not with the membership function shapes. Therefore, the size of the search space is not very large, and we propose to use a nonevolutionary inductive algorithm instead of an EA. This algorithm will be based on a linguistic rule selection function composed of a number of example covering frequentistic criteria, some of which will be presented in the next subsection. A fuzzy rule generating method of this kind is to be found in Refs. 24 and 51.
- *Design of approximate Mamdani-type FRBSs*: When working with this type of rules, the generating method has to be able to generate the membership function shapes. Due to this, the search space size is larger than the previous one, and so the use of an EA is recommended. The fitness function guiding the search will be based on combining some of the said frequentistic criteria and the phenotypic niche criterion commented in Section 3.2.1, which will allow us to deal with part of the cooperation problem in this stage.
  Two different possibilities may be distinguished when generating an approximate Mamdani-type FRB: the *constrained* and *unconstrained* ones.[27,28] We say that the fuzzy rules present a *constrained free semantic* when they are generated with a free semantic (i.e., they are approximate Mamdani-type fuzzy rules) but based on an initial domain fuzzy partition that determines the intervals in which each point defining the membership functions may take on a value. On the other hand, when the only restriction imposed on the membership function locations and shapes is lying within a specific interval, the fuzzy rules present an *unconstrained free semantic*. The most extreme case is when the interval associated to each fuzzy set corresponds to the whole domain of the system variable.

In Refs. 25, 27, and 28 a GFRBS is to be found obtained from MOGUL to generate constrained approximate Mamdani-type FRBSs, whose generating method is implemented by using a genetic local search algorithm hybridizing a GA and an (1 + 1)-ES. On the other hand, another GFRBS generating unconstrained approximate FRBSs, with a generating method implemented by means of an (1 + 1)-ES, may be found in Refs. 24, 26, 28, and 31. The latter will be the multistage GFRBS introduced in this paper as an example of the application of MOGUL.

- *Design of TSK FRBSs*: Finally, in the case in which the FRBS to be designed is of the TSK type and the antecedent part of the rules uses linguistic variables, we are in an intermediate situation, due to the fact that the generating method does not have to generate the membership function shapes but rather has to obtain a preliminary definition of the TSK rule consequent parameters p_i. Therefore, following MOGUL guidelines, we use an EA combining two different information levels in each individual: representing the antecedent part by means of an integer coding and using angular coding (see subsection 4.1.4) to represent the consequent. In this case, the fitness function is only based on one criterion, a local error measure, capable of both generating accurate TSK rules and dealing with part of the cooperation problem in this first stage.
  An example of a GFRBS of this kind obtained from MOGUL is to be found in Refs. 47 and 52. In that process, the generating method is implemented by means of a (μ, λ)-ES.

In the next three subsections we shall present some of the criteria that may be used to design the different fitness functions mentioned. The fourth subsection will be devoted to briefly introducing the angular coding employed in the latter design case.

4.1.1. The Example Covering Frequentistic Criteria Considered for Generating Linguistic Rules

The example covering frequentistic criteria considered for designing a linguistic rule multicriteria fitness function have to give the better values to the rules that are promising to form part of an accurate FRB, verifying the τ-completeness, covering and consistency properties. Amongst others, we may consider the following ones:

(a) *High frequency value*:[31] The frequency of a fuzzy rule, R_i, through the set of examples, E_p, is defined as

$$\Psi_{E_p}(R_i) = \frac{\sum_{l=1}^{p} R_i(e_l)}{p}$$

(b) *High average covering degree over positive examples*:[31] The set of positive examples to R_i with a compatibility degree greater than or equal to ω is defined as

$$E_\omega^+(R_i) = \{e_l \in E_p / R_i(e_l) \geq \omega\}$$

with $n_\omega^+(R_i)$ being equal to $|E_\omega^+(R_i)|$. The *average covering degree* on $E_\omega^+(R_i)$ can be defined as

$$G_\omega(R_i) = \sum_{e_l \in E_\omega^+(R_i)} R_i(e_l)/n_\omega^+(R_i)$$

(c) *Penalization associated to the non satisfaction of the k-consistency property*:[51] This criterion penalizes those fuzzy rules with many negative examples with respect to the number of positive examples with a compatibility degree greater than or equal to ω. In this way, it penalizes the non satisfaction of the k-consistency property. The *penalty function on the negative examples set of the rule* R_i will be

$$g_n(R_i^-) = \begin{cases} 1, & \text{if } n_{R_i}^- \le k \cdot n_\omega^+(R_i) \\ \dfrac{1}{n_{R_i}^- - kn_\omega^+(R_i) + \exp(1)}, & \text{otherwise} \end{cases}$$

It has to be noted that the negative example set is always computed over the whole training data set E_p.

4.1.2. *The Low Niche Interaction Criterion Considered to Generate Approximate Mamdani-type Fuzzy Rules*

As mentioned earlier, the aim of this criterion is to deal with part of the cooperation problem in the generation process. Since the fitness function used to design approximate Mamdani-type FRBSs is initially based on some example covering frequentistic criteria (like the ones presented in the previous subsection) and in this case the generating method has to generate the membership function shapes, the rule competition induced by this method makes the supports of the fuzzy sets wider to cover more examples, thus having a higher fitness value. Hence, the approximate fuzzy rules in the FRB generated usually interacts excessively and do not cooperate in an adequate way.

To solve this problem in the approximate GFRBSs obtained from MOGUL, we include cooperation in the generating method by inducing a new kind of niches in the search space. In this way, if the fuzzy rule being generated at this time interacts with any or some of the previously obtained ones, the former is forced to share its payoff with the latter ones. The consequence of this operation mode is that the fuzzy sets generated will present a support narrower than in the previous case, and thus the fuzzy rules in the FRB obtained will cooperate better.

The criterion to do so will be based on a phenotypic sharing scheme, the most adequate one to deal with membership function shapes (see Refs. 24 and 28). Its formulation will be the following: with $N_i = (N_i x, N_i y)$ being the centers of the rules (niches) determined until now ($i = 1, \ldots, d$, where d is the number of generating process runs performed), and C being the individual encoding the fuzzy rule being adapted, R_i, the *low niche interaction rate* (LNIR) penalizes the fitness associated to C as follows:

$$LNIR(R_i) = 1 - NIR(R_i)$$

$$NIR(R_i) = Max_i\{h_i\}$$

$$h_i = *(A(N_i x), B(N_i y)), i = 1, \ldots, d$$

$$A(N_i x) = *(A_1(N_i x_1), \ldots, A_n(N_i x_n))$$

$$C \sim R_i\text{: If } x_1 \text{ is } A_i \text{ and } \ldots \text{ and } x_n \text{ is } A_n \text{ then } y \text{ is } B$$

Hence $LNIR(C)$ penalizes the excessive interaction between the fuzzy rules, which leads to bad cooperation between them. It is defined in $[0, 1]$ and gives the maximum value (no penalization) when R_i does not interact with any of the rules generated until now. The minimum value (maximum penalization) is obtained when this rule is equal to one of those generated previously.

4.1.3. The Local Error Measure Considered to Generate TSK Fuzzy Rules

With the aim of dealing with part of the cooperation problem in the first learning stage when designing TSK FRBSs with MOGUL, we recommend using a local error measure to ensure that the fuzzy rules generated are better adjusted to the examples matching them to a higher degree. Working in this way, the adjustment of the examples matching them to a lesser degree will be done by means of the combinated action of the different rules in the FRB.

Therefore, the fitness function may be designed by using this single criterion, due to the fact that it allows us to satisfy both goals to be achieved in this stage: generating TSK fuzzy rules with a good individual behavior and with a good cooperation level between them.

As an example of this criterion, we may use the one proposed in:[53]

$$\sum_{e_l \in E} h^l \cdot \left(ey^l - S(ex^l)\right)^2$$

where E is the set of input-output data pairs $e_l \in E_p$ located in the fuzzy input subspace defined by the rule antecedent, $h^l = T(A_1(ex_1), \ldots, A_n(ex_n))$ is the matching between the antecedent part of the rule and the input part of the current data pair ex^l (T is a t-norm), and $S(ex^l)$ is the output provided by the TSK rule when it receives ex^l as input.

4.1.4. Angular Coding

There is a problem when designing TSK FRBSs using EAs: the intervals in which the TSK rule consequent parameters p_i are defined are not known and this information is usually necessary to define the coding scheme of the possible solutions and to perform evolution on them using the evolutionary operators.

In MOGUL, we propose a new coding scheme, called *angular coding*, which was first presented in Refs. 47 and 52, based on considering the geometrical properties of the TSK rule consequents and on encoding the values of the angles instead of the tangent values for each TSK rule consequent parameter by using the mapping

$$C : R \rightarrow \left(-\frac{\pi}{2}, \frac{\pi}{2}\right)$$

$$C(x) = \arctan(x)$$

This allows us to have all the parameters p_i lying in the same known interval, $(-\frac{\pi}{2}, \frac{\pi}{2})$, and to represent the whole space of possible solutions, which is not done in other TSK FRBS design methods.

4.2. The Iterative Covering Method

The covering method is developed as an iterative process that allows us to obtain a set of fuzzy rules covering the example set. In each iteration, it runs the generating method, obtaining the best fuzzy rule according to the current state of the training set, considers the relative covering value this rule creates over it, and removes from it the examples with a covering value greater than ϵ. The covering method is performed as follows:

(1) *Initialization*:
  (a) *Introduce the values of* ϵ *and the other parameters considered in the different criteria used in the fitness functions* $(k, \omega, \ldots)$.
  (b) *If linguistic rules are available from expert knowledge*:
    (i) *Introduce those rules into the final rule set* B^g.
    (ii) *Set the initial value of the example covering degree* $CV[l]$, $l = 1, \ldots, p$, *considering the covering that the rules existing in* B_g *make over the example set* E_p.
    (iii) Remove those examples with $CV[l] \geq \epsilon$ from E_p.
  (c) Otherwise:
    (i) *Initialize* B^g *to empty*.
    (ii) *Set the example covering degree* $CV[l] \leftarrow 0$, $l = 1, \ldots, p$.
(2) *Over the set of examples* E_p, *apply the generating method*, *obtaining as output the best fuzzy rule* R_r *according to the current state of* E_p.
(3) *Introduce* R_r *in* B^g.
(4) *For every* $e_l \in E_p$ *do*
  (a) $CV[l] \leftarrow CV[l] + R_r(e_l)$,
  (b) *If* $CV[l] \geq \epsilon$ *then remove it from* E_p.
(5) *If* $E_p = \varnothing$ *then Stop else return to Step* 2.

5. THE GENETIC MULTISIMPLIFICATION PROCESS

As mentioned earlier, in this process we follow the idea of obtaining different simplified FRBs from the fuzzy rule set generated in the previous stage presenting the best possible cooperation between the fuzzy rules composing them. The *Sequential Niche Technique*[54] is used to induce niches in this GFRBS stage, with the genetic simplification process proposed in Ref. 31 being the basic optimization technique iterated in each run of the multisimplification process. The following subsections introduce the basic simplification algorithm and the particular aspects of the multisimplification one, respectively.

5.1. The Basic Genetic Simplification Process

The basic genetic simplification process was first proposed in Ref. 31. It is based on a binary coded GA, in which the selection of the individuals is performed using the stochastic universal sampling procedure together with an

elitist selection scheme, and the generation of the offspring population is put into effect by using the classical binary multipoint crossover (performed at two points) and uniform mutation operators.

The coding scheme generates fixed-length chromosomes. Considering the rules contained in the rule set B^g derived from the previous step counted from 1 to m, an m-bit string $C = (c_1, \ldots, c_m)$ represents a subset of candidate rules to form the FRB finally obtained as this stage output, B^s, such that,

$$If\ c_i = 1 \text{ then } R_i \in B^s \text{ else } R_i \notin B^s$$

Following MOGUL assumptions, the initial population is generated by introducing a chromosome representing the complete previously obtained rule set B^g, i.e., with all $c_i = 1$. The remaining chromosomes are selected at random.

As regards the fitness function, $F(C_j)$, it is based on two different criteria:

- On the one hand, we have a global error measure that determines the accuracy of the FRBS encoded in the chromosome. We usually work with the mean square error (SE), although other measures may be used. SE over a training data set, E_{TDS}, is represented by the following expression:

$$E(C_j) = \frac{1}{2|E_p|} \sum_{e_l \in E_p} \left(ey^l - S(ex^l)\right)^2$$

  where $S(ex^l)$ is the output value obtained from the FRBS using the FRB coded in C_j, $R(C_j)$, when the input variable values are $ex^l = (ex_1^l, \ldots, ex_n^l)$, and ey^l is the known desired value.
- On the other hand, since there is a need to keep the *τ-completeness property* considered in the previous stage, we shall ensure this condition by forcing every example contained in the training set to be covered by the encoded FRB to a degree greater than or equal to τ,

$$C_{R(C_j)}(e_l) = \bigcup_{j=1\ldots T} R_j(e_l) \geq \tau, \qquad \forall e_l \in E_p \text{ and } R_j \in R(C_j)$$

  where τ is the minimal training set completeness degree accepted in the simplification process. Usually, τ is less than or equal to ω, the compatibility degree used in the generation process.
  Therefore, we define a *training set completeness degree* of $R(C_j)$ over the set of examples E_p as

$$TSCD\left(R(C_j), E_p\right) = \bigcap_{e_l \in E_p} C_{R(C_j)}(e_l)$$

  The final expression of the fitness function is:

$$F(C_j) = \begin{cases} E(C_j), & \text{if } TSCD\left(R(C_j), E_p\right) \geq \tau, \\ \infty, & \text{otherwise} \end{cases}$$

5.2. The Genetic Multisimplification Process

In order to induce niching in the sequential niche algorithm, there is a need to define any kind of *distance metric* which, given two individuals, returns a value of how close they are.[54] We use a *genotypic sharing* due to the fact that the metric considered is the Hamming distance measured on the binary coding space. With $A = (a_1, \ldots, a_m)$ and $B = (b_1, \ldots, b_m)$ being two individuals, it is defined as follows:

$$H(A, B) = \sum_{i=1}^{m} a_i \cdot b_i$$

Making use of this metric, the *modified fitness function* guiding the search on the multisimplification process is based on modifying the value associated to an individual by the basic algorithm fitness function, multiplying it by a *derating function* penalizing the closeness of this individual to the previously obtained solutions. Hence, the modified fitness function used by the multisimplification process is the following:

$$F'(C_j) = F(C_j) \cdot G(C_j, S)$$

where F is the basic genetic simplification process fitness function, $S = \{s_1, \ldots, s_k\}$ is the set containing the k solutions yet found, and G is a kind of *derating function*. We consider the following, taking into account the fact that the problem we deal with is a minimization one:

$$G(C_j, S) = \begin{cases} \infty, & \text{if } d = 0 \\ 2 - \left(\dfrac{d}{r}\right)^{\beta}, & \text{if } d < r \text{ and } d \neq 0 \\ 1, & \text{if } d \geq r \end{cases}$$

where d is the minimum value of the Hamming distance between C_j and the solutions s_i included in S, i.e., $d = Min_i\{H(C_j, s_i)\}$, and the penalization is considered over the closest solution, r is the *niche radius*, and β is the *power factor* determining how concave ($\beta > 1$) or convex ($\beta < 1$) the derating curve is. Therefore, the penalization given by the derating function takes its maximum value when the individual C_j encodes one of the solutions already found. There is no penalization when the C_j is far away from S with a value greater than or equal to the niche radius r.

The algorithm of the genetic multisimplification process is shown below:

(1) *Initialization*: *Equate the multisimplification modified fitness function to the basic simplification fitness function*: $F'(C_j) \leftarrow F(C_j)$.
(2) *Run the basic genetic simplification process, using the modified fitness function, keeping a record of the best individual found in the run.*
(3) *Update the modified fitness function to give a depression in the region near the best individual, producing a new modified fitness function.*
(4) *If not all the simplified FRBs desired have been obtained, return to step* 2.

Hence, the number of runs of the sequential algorithm performed is the number of solutions desired to obtain. We allow the FRBS designer to decide this number as well as the values of parameters r and β.

6. THE EVOLUTIONARY TUNING PROCESS

Finally, as regards the third learning stage, MOGUL presents different design aspects with respect to the representation scheme, the adjustment of membership functions and TSK consequent parameters, and to the composition of the fitness function. We shall analyze these aspects in the following subsections.

6.1. Representation Scheme

As shown in Section 3.3, two different types of adjustment of the initial definitions of the membership functions depending on the nature of the FRB are considered in MOGUL. In the case of descriptive Mamdani-type or TSK FRBs, we shall globally tune the fuzzy partitions associated to the linguistic variables (see, for example, the genetic tuning process presented in Refs. 24 and 51); whilst in the case of the approximate Mamdani-type FRBs, each membership function in each fuzzy rule is individually tuned (as in the case of the genetic tuning process presented in Ref. 30).

In both cases, the coding scheme considered is the same, a real one, which is the most adequate one for dealing with membership functions, and the representation scheme is very similar. When working with linear membership functions, each individual membership function may be represented by means of three or four real values, depending on whether it is triangular or trapezoidal-shaped. The individuals are obtained by joining the partial definitions of the fuzzy sets existing in the initial fuzzy partitions, in the case of descriptive Mamdani-type or TSK FRBSs, or in the different fuzzy rules, in the case of the approximate ones.

On the other hand, when working with TSK FRBs, the individuals will include a part encoding the TSK rule consequent parameters to be refined by the EA as well. In this case, we call the process the *evolutionary refinement process* because it does not only tune the initial definitions of the membership functions but refines the initial definitions of the TSK rule consequent parameters as well. An example may be found in Ref. 47.

6.2. Definition of the Intervals of Adjustment

In this subsection, we are going to describe how the initial membership function definitions are considered to define the intervals of adjustment for each parameter. It is clear that these intervals have to be well defined in order to obtain meaningful membership functions from the evolutionary tuning process. Therefore, given a triangular fuzzy set defined by the 3-tuple (a, b, c), the

following two different ways for defining the intervals of adjustment are considered in MOGUL:

- On the one hand, a *fixed interval of adjustment*, depending on the values of the neighbor parameters, may be associated to each one:

$$a \in [a^l, a^r] = \left[a - \frac{b-a}{2}, a + \frac{b-a}{2}\right]$$

$$b \in [b^l, b^r] = \left[b - \frac{b-a}{2}, b + \frac{c-b}{2}\right]$$

$$c \in [c^l, c^r] = \left[c - \frac{c-b}{2}, c + \frac{c-b}{2}\right]$$

  Figure 2 graphically shows these intervals, which remain constant during the evolutionary tuning process.
- On the other hand, a *variable interval of adjustment* may be associated to each one of the three parameters. To do so, a global interval, obtained from its initial definition, is associated to each fuzzy set in the FRB:

$$[L, R] = \left[a - \frac{b-a}{2}, c + \frac{c-b}{2}\right]$$

  This one will be the only interval remaining constant during the EA run in this second case. The intervals associated to parameters a, b and c will vary depending on the specific individual and on the current neighbor values as shown below:

$$a \in [a^l, a^r] = [L, b], \qquad b \in [b^l, b^r] = [a, c], \qquad c \in [c^l, c^r] = [b, R]$$

  Figure 3 graphically shows the initial values of these intervals.

As may be clearly observed, the second possibility provides a greater degree of freedom in the membership function tuning, making them finally present a meaningful form in $[L, R]$. On the other hand, the first one only performs a more local adjustment. Fixed intervals are considered in the evolutionary tuning

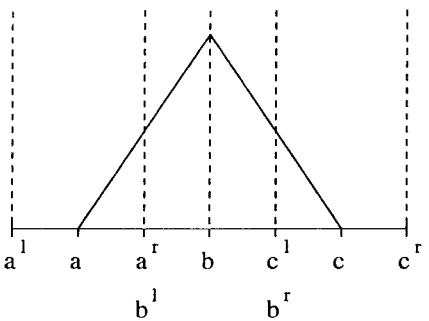

Figure 2. Definition of fixed intervals of adjustment.

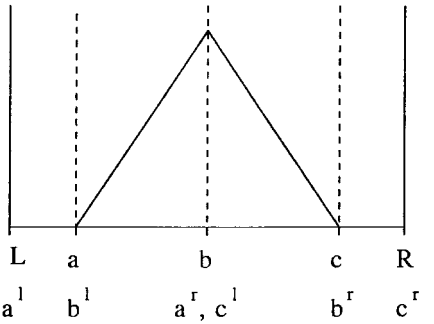

Figure 3. Initial definition of the variable interval of adjustment.

processes obtained from MOGUL presented in Refs. 24, 29, and 51, whereas the variable ones in the one presented in Ref. 47.

Finally, as regards the TSK rule consequent parameters adaption, we do not need to define any interval of adjustment, due to the use of the angular coding mentioned in Section 4.1.4.

6.3. Fitness Function

The composition of the fitness function to be used in this third stage will depend on the type of FRBSs being designed. If we are working with Mamdani-type FRBSs, regardless of whether they are descriptive or approximate, the FRBs that are to be tuned in this process verify the τ-completeness property. Therefore, there is a need to maintain the satisfaction of this property after the evolutionary tuning process run. In this case, we will work with the same fitness function used in the multisimplification process, based on two criteria:

$$F(C_j) = \begin{cases} EM(C_j), & \text{if } TSCD\big(R(C_j), E_P\big) \geq \tau, \\ \infty, & \text{otherwise} \end{cases}$$

where $EM(\cdot)$ is a global error measure (we shall again work with the SE) and $TSCD$ is the criterion penalizing the lack of the τ-completeness property (see Section 5.1).

On the other hand, when designing TSK FRBSs, the FRBs to be refined do not verify the completeness property, so the fitness function may only be composed of one criterion, the global error measure:

$$F(C_j) = EM(C_j)$$

7. ANALYSIS OF THE MOGUL PARAMETERS

In this section, we are going to analyze the behavior that the GFRBSs obtained from MOGUL may present depending on the possible values for the

parameters existing in them. As may be seen in the previous sections, there are two main parameters in these GFRBSs: the parameter $\epsilon \in \mathbf{R}^+$, which defines the degree to which an example has to be covered to be removed from the training data set in the generation process, and the parameter $\tau \in (0, 1]$, associated to the τ-completeness property in the multisimplification and tuning processes. The influence of the others (for example, k and ω, the parameters associated to the k-consistency property and to the positive example set in the generating method) is not as important, as may be drawn from the results of the experiments performed in Ref. 51.

Both parameters, ϵ and τ, fit the number of rules that will form part of the FRB generated (and, consequently, the FRBS accuracy). Therefore, the user may generate the FRB with the desired accuracy-complexity balance by controlling the values of both parameters in the following way:

- When the main design requirement is system accuracy, the value of ϵ must be high. This will make the obtaining of an FRBS performing well, having a big number of rules in the FRB.
- On the other hand, when the main requirement is to obtain an FRB with low-complexity (for example, in some real-time control applications), the choice must be the opposite, a low value for ϵ. In this case, the FRB generated will have less rules, but this will make the designed FRBS perform worse.
- The value of τ must always be low due to the fact that this gives more freedom to the multisimplification and tuning processes, allowing us to obtain more accurate and simpler FRBs from MOGUL.

In Refs. 24, 25, 26, 28, and 51, different experimental results are to be found from some multistage GFRBSs obtained from MOGUL that confirm these assumptions in practice.

8. EXAMPLE OF APPLICATION OF MOGUL

Different multistage GFRBSs following the methodology proposed may be found in Refs. 24–28, 31, 47, 51, 52, and 55. In this contribution, we will briefly describe one of them which is able to generate unconstrained approximate Mamdani-type FRBSs. The composition of this specific GFRBSs obtained from MOGUL as proposed in this contribution, is described below (for a more complete description refer to Refs. 24 and 26).

(1) *An evolutionary generation process* composed of a fuzzy rule generating method based on an inductive algorithm with an (1 + 1)-ES that locally tunes the fuzzy rule membership functions, and the covering method introduced in Section 4.2. The fitness function is composed of the three frequentistic criteria, allowing to select the most promising rule to verify the completeness and consistency properties, and the low niche interaction rate, a criterion to deal with the cooperation between the fuzzy rules generated in this first stage. These four criteria have been introduced in Section 4.1.
(2) *The genetic multisimplification process* presented in Section 5.

(3) *A genetic tuning process*, based on a Real Coded GA and the same fitness function considered in the previous stage, a combination of the SE and the criterion penalizing the lack of the completeness property presented in Section 5.1. The approximate genetic tuning process used was presented in Ref. 29 and is based on working with the fixed intervals of adjustment (see Section 6.2). In this paper, the GA uses the max-min-arithmetical crossover operator[45] and Michalewicz's nonuniform mutation.[11] The selection mechanism considered is the stochastic universal sampling procedure together with an elitist selection scheme.

9. PRACTICAL APPLICATION OF THE PRESENTED GENETIC FUZZY RULE-BASED SYSTEM TO A REAL-WORLD ELECTRICAL ENGINEERING PROBLEM

In order to analyze the accuracy of MOGUL, we are going to use the GFRBS proposed in the previous section to solve a real-world Electrical Engineering problem consisting of obtaining a model relating the length of line in a rural population with its characteristics.[55–57] We shall compare the behavior of our multistage GFRBS in solving the problem with the one presented by classical methods, neural networks and another multistage GFRBSs presenting different characteristics.

To do so, first we shall introduce the application in the next subsection. Then, the different techniques considered to solve it will be analyzed. Finally, the results obtained will be compared.

9.1. The Electrical Engineering Problem as Considered

In Spain, electrical industries do not charge the energy bill directly to the final user, but they share the ownership of a company (called R.E.E., Red Eléctrica de España) which receives all payments and then distributes them according to complex criteria (amount of power generation by every company, number of customers, etc.).

Recently, some of these companies have asked to redistribute the maintenance costs of the network. Since maintenance costs depend on the total length of electrical line each company owns, and their type (high, medium, urban low and rural low voltage) it was necessary to know the exact length of every kind of line each company was maintaining.

High and medium voltage lines can be easily measured. But low voltage line is contained in cities and villages, and it would be very expensive to measure it. This kind of line is usually very convoluted and, in some cases, one company may serve more than 10 000 small centers of population. An indirect method for determining the length of line is needed.

Therefore, there is a need to find a relationship between the population and size of a certain area and the length of line in it, making use of some known data, that may be employed to predict the real length of line in any given village.

We shall try to solve this problem by generating different kind of models determining the unknown relationship. To do so, we were provided with the

measured line length, the number of inhabitants and the mean distance from the center of the town to the three furthest clients in a sample of 495 rural center.[56,57] Our variables are named as shown in Table I.

9.2. Analyzing Different Techniques to Solve the Problem

In order to apply classical regression methods, some hypotheses have to be formed (see Refs. 55 and 57). By assuming them, we build two different theoretical simplified models represented by the equations:

$$\tilde{l}_i/R_i = s + k(\theta)A_i; \qquad \frac{\tilde{l}_i}{R_i} = k_1 A_i^{k_2}$$

whose parameters can be estimated by means of a least squares linear and an exponential regression, respectively, to a set of pairs $(x, y) = (A_i, l_i/R_i)$.

In the experiments performed, the parameters of the polynomial models were fitted by Levenberg-Marquardt, while exponential and linear models were fitted by linear least squares.

With respect to the use of Neural Networks, we have worked with the multilayer perceptron, which was trained with the QuickPropagation algorithm. The number of neurons in the hidden layer was chosen to minimize the test error.

Finally, different ways of developing a fuzzy modeling of the problem introduced are going to be compared by using several FRBS design methods following different types of fuzzy models.

As regards the descriptive fuzzy model, the two following processes are considered:

- **(D1)** a two-stage GFRBS based on obtaining a preliminary FRB by means of Wang and Mendel's (WM) method[58] in the first stage, and tuning the definition of the membership functions by means of the descriptive genetic tuning process presented in Refs. 24 and 51 (see Section 6) in the second one, and
- **(D2)** a two-stage GFRBS based on obtaining a preliminary FRB by means of Thrift's genetic learning process[59] in the first stage, and refining the definition of the membership functions by means of the same descriptive genetic tuning process used above.

Table I. Notation considered for the problem variables.

Symbol	Meaning
A_i	Number of clients in population
R_i	Radius of i population in the sample
n	Number of populations in the sample
l_i	Line length, population i
$\tilde{l}_i$	Estimation of l_i

On the other hand, when working with the approximate one, the following two multistage GFRBSs are employed:

(**A1**) a two-stage GFRBS based on obtaining a complete FRB by generating a preliminary definition by means of the weighted counting algorithm,[2] and refining it by adjusting the membership function definitions using the approximate genetic tuning process proposed in Ref. 31 and used in Ref. 24 (see Section 6), and
(**A2**) the three-stage unconstrained approximate GFRBS described in Section 8.

The initial fuzzy partitions considered (two corresponding to the input variables and one associated to the output one) for the four GFRBSs are formed by *five fuzzy sets* (as shown in Fig. 4), and the adequate scaling factors to translate the generic universe of discourse into the one associated with each problem variable.

With respect to the parameter values considered for GFRBS (**A2**), $\epsilon = 1.5$, $\omega = 0.05$, $k = 0.1$, and $\tau = 0.1$. On the other hand, the t-norm $*$ used in the fuzzy rule generation process is the Minimum, the ES is applied until there is no improvement in 100 generations (the parameter c of the $\frac{1}{5}$-success rule is equal to 0.9). The genetic multisimplification process generates three different FRBs per run (each time, the basic GA runs over 500 generations for a population of 61 individuals), the niche radius, r, is equal to 10 percent of the number of rules in the initial FRB, and the power factor, β, is equal to 0.5. The approximate genetic tuning process runs over 1000 generations, the genetic population is formed by 61 individuals, the value of Michalewicz's non-uniform mutation parameter b is 5.0, and the crossover and mutation rates are, respectively, $P_c = 0.6$ and $P_m = 0.1$ (this last one per individual).

As regards the GFRBS (**D1**), (**D2**) and (**A1**), the parameters considered in the second stage, the genetic tuning process, regardless of its descriptive or approximate nature, are the same as shown above. While the WM generation process does not consider any parameter, the values associated to Thrift's one in (**D2**) and the weighted counting algorithm one in (**A1**), are the following:

- **Thrift**: Population size: 61, $P_c = 0.6$, $P_m = 0.1$ and number of generations: 1000.
- **Weighted Counting Algorithm**: $\alpha = 0.5$.

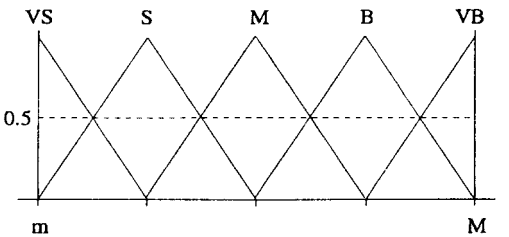

Figure 4. Fuzzy partition considered in the experiments performed.

Finally, the FRBS reasoning method used in all the processes has been the same. We have selected the *Minimum t-norm* playing the role of the implication and conjunctive operators, and the *Center of Gravity weighted by the matching* strategy acting as the defuzzification operator.[7]

9.3. Comparison Between Methods

To compare classical methods, GFRBS fuzzy modeling, and Neural Modeling we have randomly divided the sample into two sets comprising 396 and 99 samples. The SE values over these two sets are labeled as *training* and *test*. In this case, we define SE as

$$\frac{1}{2 \cdot N} \sum_{i=1}^{N} \left(\tilde{l}_i - l_i \right)^2$$

The results obtained in the different experiments performed with the GFRBSs considered are collected in Table II where $\#R$ stands for the number of rules in the corresponding FRB, and SE_{tra} and SE_{tst} for the values obtained in the SE measure computed over the training and test data sets, respectively.

In view of the results obtained, we should remark on: (a) the good performance of the genetic multisimplification process since, in the second and third iterations, it allows us to generate fuzzy models with better approximate and predictive behavior, i.e., less value in the SE over both data sets, than the first one obtained; and (b) the good performance of the genetic tuning processes, that broadly improve the accuracy of the preliminary FRB definitions.

Once we have individually analyzed the behavior presented by the proposed GFRBS, we are going to compare its accuracy with the remaining techniques considered. Table III shows the results obtained by all of them in the problem.

In view of the results shown, the unconstrained approximate GFRBS proposed in this paper has presented the best behavior. It outperforms the other techniques considered by obtaining the best values in the SE computed over both data sets, the training and test ones. Therefore, the unconstrained approximate FRBS generated is the model that best approximates the real system and that presents the best generalization capabilities.

Table II. Results obtained by the multistage GFRBSs in the problem being solved.

	Generation			Multisimplification			Tuning	
GFRBS	$\#R$	SE_{tra}	SE_{tst}	$\#R$	SE_{tra}	SE_{tst}	SE_{tst}	EC_{prue}
D1	13	298446.0	282058.1				175337.9	180102.7
D2	25	218591.9	204426.8				154314.0	199551.3
A1	20	356434.3	311195.0				175887.2	180211.4
A2	31	431904.0	435649.5	19	226403.9	222550.9	148036.9	191339.5
				20	227261.6	227105.6	142108.6	166578.7
				16	227232.9	225789.7	136826.4	177612.3

Table III. Results obtained in the problem being solved.

Method	SE_{tra}	SE_{tst}
Linear	287775	209656
Exponential	232743	197004
2nd order polynomial	235948	203232
3rd order polynomial	235934	202991
3 layer perceptron 2-25-1	169399	167092
D1	175337	180102
D2	154314	199551
A1	175887	180211
A2	142108	166578

We should point out that the fuzzy model obtained is more accurate to a high degree than the neural one, which is the second best model in view of its generalization level. Although the results do not differ too much in this characteristic (166578 vs. 167092), the value obtained by the constrained approximate FRBS in the SE over the training data set shows a large performance advantage for it over the neural network (142108 vs. 169399).

Hence, we have been able to generate a model that is more accurate and easier to interpret at the same time. This is due to the fact that, although approximate Mamdani-type FRBs are less readable than descriptive ones, approximate FRBSs are easier to interpret than neural networks for the following two main reasons:

- The approximate fuzzy model is locally interpretable. We are always able to know which fuzzy rules in the FRB are fired when the system receives a specific input.
- The parameters involved in an approximate fuzzy model have a real-world meaning understandable by a human since they define membership functions. However, it is difficult to interpret the meaning of the neural network weights.

10. CONCLUDING REMARKS

MOGUL, an evolutionary methodology for designing FRBSs by learning the FRB from examples, consisting of different guidelines to obtain multistage GFRBSs based on the IRL approach has been presented. A specific evolutionary process to design unconstrained approximate Mamdani-type FRBSs derived from this paradigm has been introduced and its application to a real-world Electrical Engineering problem has been shown and compared with classical methods, Neural Networks and other GFRBSs. The proposed GFRBS has obtained very good results.

MOGUL will allow different users to obtain their own GFRBSs able to deal with their specific problems. Therefore, any user may add his particular requirements to MOGUL guidelines for designing any kind of FRBS to solve his problem in an adequate way. To do so, the user only has to design his own evolutionary process for each one of the GFRBS learning stages, ensuring that it verifies MOGUL assumptions.

The performance of MOGUL has been demonstrated in fuzzy modeling problems until now. At the moment, we are directly applying it to fuzzy classification problems[60] and extending it to include new fuzzy reasoning methods for classification problems in the learning process,[62,63] obtaining good results. As future work, we intend to use MOGUL to design a new kind of FRBS, the approximate TSK one, in which the antecedent part of the fuzzy rule presents an approximate nature and the consequent part is a linear combination of the inputs, to propose new definitions of the LNIR criterion in order to improve the behavior of the approximate GFRBSs obtained, and to extend it to deal with Mamdani-type fuzzy rules with a different structure, as the disjunctive normal form where each linguistic variable may have different linguistic values associated in the same rule (see Refs. 41, 63, and 64.).

We would like to thank to Luciano Sánchez, from Oviedo University, for the Electrical Engineering application from Hidroeléctrica del Cantábrico and for solving it by means of classical and neural techniques.

References

1. Zadeh, L. A. Inform and Control 1965, 8, 338–353.
2. Bardossy, A.; Duckstein, L. Fuzzy Rule-Based Modeling With Application to Geophysical, Biological and Engineering Systems; CRC Press: Boca Raton, FL, 1995.
3. Industrial Applications of Fuzzy Technology; Hirota, K., Ed., Springer-Verlag: Berlin, 1993.
4. Fuzzy Modelling. Paradigms and Practice; Pedrycz, W., Ed., Kluwer Academic Press: 1996.
5. An Introduction to Fuzzy Logic Applications in Intelligent Systems; Yager, R. R.; Zadeh, L. A., Eds., Kluwer Academic Press: 1992.
6. Cao, Z.; Kandel, A. Fuzzy Sets and Systems 1989, 31, 151–186.
7. Cordón, O.; Herrera, F.; Peregrín, A. Fuzzy Sets and Systems 1997, 86, 15–41.
8. Cordón, O.; Herrera, F.; Peregrín, A. Fuzzy Sets and Systems 1999 (to appear).
9. Kiszka, J.; Kochanska, M.; Sliwinska, D. Fuzzy Sets and Systems 1985, 15, 111–128, 223–240.
10. Bäck, T. Evolutionary Algorithms in Theory and Practice; Oxford University Press: Oxford, 1996.
11. Michalewicz, Z. Genetic Algorithms + Data Structures = Evolution Programs; Springer-Verlag: Berlin, 1996.
12. Cordón, O.; Herrera, F. In Genetic Algorithms in Engineering and Computer Science; Periaux, J.; Winter, G.; Galán, M.; Cuesta, P., Eds., John Wiley and Sons: New York, 1995; 33–57.
13. Herrera, F.; Magdalena, L. In Genetic fuzzy systems; Mesiar, R.; Riecan, B., Eds., Tatra Mountains Mathematical Publications: 1997, 13, 93–121. Lecture Notes of the Tutorial: Genetic Fuzzy Systems. Seventh IFSA World Congress (IFSA'97).
14. Bonissone, P. P. Soft Computing 1997, 1, 6–18.
15. Cordón, O.; Herrera, F.; Lozano, M. In Genetic Algorithms and Fuzzy Logic Systems. Soft Computing Perspectives; Sanchez, E.; Shibata, T.; Zadeh, L., Eds., World Scientific: Singapore, 1997; 209–241.
16. Cordón, O.; Herrera, F.; Lozano, M. In Fuzzy Evolutionary Computation; Pedrycz, W., Ed., Kluwer Academic Press: Dordrecht, The Netherlands, 1997; 57–77.
17. Bonarini, A. In Fuzzy Modelling: Paradigms and Practice; Pedrycz, W., Ed.; Kluwer Academic Press: Dordrecht, The Netherlands, 1996; 265–283.

18. González, A.; Herrera, F. Mathware Soft Comput 1997, 4, 233–249.
19. Driankov, D.; Hellendoorn, H.; Reinfrank, M. An Introduction to Fuzzy Control; Springer-Verlag: Berlin, 1993.
20. Lee, C. C. IEEE Trans Syst Man Cybernet 1990, 20, 404–435.
21. Takagi, T.; Sugeno, M. IEEE Trans Syst Man Cybernet 1985, 15, 116–132.
22. Bastian, A. Internat J of Uncertain Fuzziness Knowledge-Based Systems 1994, 2, 463–484.
23. Carse, B.; Fogarty, T. C.; Munro, A. Fuzzy Sets and Systems 1996, 80, 273–294.
24. Cordón, O.; Herrera, F. Internat J Approx Reasoning 1997, 17, 369–407.
25. Cordón, O.; Herrera, F. In Fuzzy Logic and Soft Computing; Herrera, F.; Verdegay, J. L., Eds., Physica-Verlag: 1996; 251–278.
26. Cordón, O.; Herrera, F. Proc Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU'96), Granada, Spain, 1996; 733–738.
27. Cordón, O.; Herrera, F. Fuzzy Sets and Systems (to appear).
28. Cordón, O.; Herrera, F. In Fuzzy Model Identification. Selected Approaches; Driankov, D.; Hellendoorn, H., Eds., Springer-Verlag: Berlin, 1997; 215–250.
29. Herrera, F.; Lozano, M.; Verdegay, J. L. Internat J Approx Reason 1995, 12, 299–315.
30. Delgado, M.; Gómez Skarmeta, A. F.; Martín, F. IEEE Trans Fuzzy Systems 1997, 5, 223–233.
31. Herrera, F.; Lozano, M.; Verdegay, J. L. Fuzzy Sets and Systems 1998, 100, 143–158.
32. Magdalena, L.; Velasco, J. R. In Fuzzy Logic and Soft Computing; Herrera, F.; Verdegay, J. L., Eds., Physica-Verlag: 1996; 172–201.
33. Herrera, F.; Verdegay, J. L., Eds., Genetic Algorithms and Soft Computing; Physica-Verlag: 1996.
34. Fuzzy Evolutionary Computation; Pedrycz, W., Ed., Kluwer Academic Press: 1997.
35. Genetic Algorithms and Fuzzy Logic Systems. Soft Computing Perspectives; Sanchez, E.; Shibata, T.; Zadeh, L., Eds., World Scientific: Singapore, 1997.
36. Booker, L. Intelligent Behaviour as an Adaption to the Task Environment, PhD thesis, University of Michigan, 1982.
37. Holland, J. H.; Reitman, S. Pattern-Directed Inference Systems; Waterman, D. A.; Hayes Roth, F., Eds., Academic Press, 1978.
38. Smith, S. F. A Learning System Based on Genetic Adaptive Algorithms, PhD thesis, University of Pittsburgh (1980).
39. Venturini, G. Proc European Conference on Machine Learning (ECML'92), Vienna, Austria, 1992; 280–296.
40. Deb, K.; Goldberg, D. E. Proc of the Second International Conference on Genetic Algorithms, Lawrence Erlbaum, Hillsdale, NJ, 1989; 42–50.
41. Cordón, O.; del Jesus, M. J.; Herrera, F.; Lozano, M. Proc Second IEE/IEEE International Conference on Genetic Algorithms and Engineering Systems: Innovations and Applications (GALESIA'97), Glasgow, U.K., September 1997; 139–144.
42. González, A.; Pérez, R. Fuzzy Sets and Systems 1998, 96, 37–51.
43. Krishnakumar, K.; Satyadas, A. In Genetic Algorithms in Engineering and Computer Science; Periaux, J.; Winter, G.; Galán, M.; Cuesta, P., Eds., John Wiley and Sons, New York, 1995; 305–320.
44. Satyadas, A.; Krishnakumar, K. Proc First Industry/University Symposium on High Speed Civil Transport Vehicles. North Carolina, USA, December 1994.
45. Herrera, F.; Lozano, M.; Verdegay, J. L. Fuzzy Sets and Systems 1997, 92, 21–30.
46. Herrera, F.; Lozano, M. Internat J Intell Syst 1996, 11, 1013–1040.
47. Cordón, O.; Herrera, F. IEEE Trans Syst Man Cybernet 1999, 29.
48. Jog, P.; Suh, J. Y.; Gutch, D. V. Proc Third International Conference on Genetic Algorithms (ICGA) 1989, 110–115.
49. Suh, J. Y.; Gutch, D. V. Proc Second International Conference on Genetic Algorithms (ICGA) 1987, 100–107.
50. Wang, L. X. Adaptive Fuzzy Systems and Control; Prentice-Hall: London, 1994.

51. Cordón, O.; Herrera, F.; Lozano, M. In L.N.C.S. 1141, Proc. Fourth International Conference on Parallel Problem Solving from Nature; Voight, H. M.; Ebeling, W.; Rechemberg, E.; Schwefel, H. P., Eds., PPSN IV: Berlin, Germany, 1996; 720–729.
52. Cordón, O.; Herrera, F. Proc Sixth IEEE International Conference on Fuzzy Systems (FUZZ-IEEE'97), Barcelona, Spain, July 1997; Vol. I, 509–514.
53. Yen, J.; Gillespie, W. Proc Sixth International Fuzzy Systems Association World Congress (IFSA'95), Sao Paulo, Brazil, July 1995; 121–124.
54. Beasly, D.; Bull, D. R.; Martin, R. R. Evolutionary Computation 1993, 1, 101–125.
55. Cordón, O.; Herrera, F.; Sánchez, L. Applied Intelligence 1999, 10, 5–24.
56. Sánchez, L. Interval-valued GA-P algorithms, IEEE Trans Evolutionary Computation (to appear).
57. Sánchez, L. Estudio de la red asturiana de baja tensión rural y urbana (in Spanish), Confidential Report (1997), Hidroeléctrica del Cantábrico Research and Development Department.
58. Wang, L. X.; Mendel, J. M. IEEE Trans Syst Man Cybernet 1992, 22, 1414–1427.
59. Thrift, P. Proc Fourth International Conference on Genetic Algorithms (ICGA'91) 1991, 509–513.
60. Cordón, O.; del Jesus, M. J.; Herrera, F.; López, E. Proc Seventh International Fuzzy Systems Association World Congress (IFSA'97), Prague, Checz, June 1997; Vol. 2, 424–429.
61. Cordón, O.; del Jesus, M. J.; Herrera, F. Internat J Approx Reasoning 1999, 20, 21–45.
62. Cordón, O.; del Jesus, M. J.; Herrera, F. Internat J Int Syst 1998, 13, 1025–1053.
63. Magdalena, L.; Monasterio-Huelin, F. Internat J Approx Reason 1997, 16, 335–358.
64. Magdalena, L. Internat J Approx Reason 1997, 17, 327–350.
65. Cordón, O.; Herrera, F.; Hoffmann, F.; Magdelena, L. Genetic Fuzzy Systems. Evolutionary Tuning and Learning of Fuzzy Knowledge Bases; World Scientific (in preparation).