Genetic Learning of Fuzzy Rule-Based Classification Systems Cooperating with Fuzzy Reasoning Methods

Oscar Cordón*
Department of Computer Science and Artificial Intelligence, University of Granada, 18071 Granada, Spain

María José del Jesus†
Department of Computer Science, University of Jaén, 23071 Jaén, Spain

Francisco Herrera‡
Department of Computer Science and Artificial Intelligence, University of Granada, 18071 Granada, Spain

In this paper, we present a multistage genetic learning process for obtaining linguistic fuzzy rule-based classification systems that integrates fuzzy reasoning methods cooperating with the fuzzy rule base and learns the best set of linguistic hedges for the linguistic variable terms. We show the application of the genetic learning process to two well known sample bases, and compare the results with those obtained from different learning algorithms. The results show the good behavior of the proposed method, which maintains the linguistic description of the fuzzy rules. © 1998 John Wiley & Sons, Inc.

1. INTRODUCTION

In a supervised inductive learning process, a pattern classification system is developed by means of a set of classified examples used to establish a correspondence between the new pattern descriptions and the class labels. This system is a description of the concepts learned and it can be expressed as a logic expression, a set of fuzzy rules, a decision tree, or a neural network, among others. When the classification system is composed of a set of fuzzy rules, it is called a fuzzy rule-based classification system (FRBCS).

* E-mail: ocordon@decsai.ugr.es.
† E-mail: mjjesus@ujaen.es.
‡ E-mail: herrera@decsai.ugr.es.

INTERNATIONAL JOURNAL OF INTELLIGENT SYSTEMS, VOL. 13, 1025–1053 (1998)
 CCC 0884-8173/98/101025-29

Fuzzy rule-based systems (FRBSs) have been successfully applied in pattern classification problems.[1,2] The interest in using FRBSs arises from the fact that they provide a good platform to deal with noisy, imprecise, or incomplete information, which is often handled in any human-cognition system. These kinds of systems are an effort to reconcile the empirical precision of traditional engineering techniques and the interpretability of artificial intelligence.

An FRBCS is formed by two main components: (1) a knowledge base (KB), i.e., a rule base (RB) and a data base (DB) for a specific classification problem, and (2) a fuzzy reasoning method (FRM), which classifies a new pattern using the information given by the KB. As regards the KB design process and, specifically, the RB design process, several proposals have been widely used, some of which can be found in Refs. 2–7. The most commonly used fuzzy inference method, maximum matching,[3,5–9] classifies an example using the rule consequent with the highest association degree, losing the information given by other rules to a lesser degree. The generalization power provided by an FRBCS that includes an FRM that uses the information given by all the rules has been proven.[1,2,10–12]

In this paper, we introduce a multistage genetic learning process to obtain linguistic FRBCSs. It is based on a specific genetic learning methodology, MOGUL (*m*ethodology to *o*btain *g*enetic fuzzy rule-based systems *u*nder the iterative rule *l*earning approach).[13] The genetic learning process integrates FRMs cooperating with the KB and learns the best set of linguistic hedges for the linguistic variable terms. According to MOGUL guidelines, the process contains the three following steps:

- The first step generates an RB regardless of the FRM used. The fuzzy partitions for the linguistic variables are predefined by the classification system builder.
- In the second step, a genetic algorithm-based process selects the best subset of fuzzy rules and learns the best set of linguistic hedges for the linguistic variables cooperating with the FRMs.
- Finally, a genetic tuning process of the fuzzy partitions is carried out with the linguistic hedges learned in the previous stage.

This automatic knowledge extraction process is based on genetic algorithms (GAs) in two of its three stages. GAs are search algorithms that use operations found in natural genetics to guide the trek through a search space.[14] GAs are theoretically and empirically proven to provide robust search capabilities in complex spaces, offering a valid approach to problems requiring efficient and effective searching.

We will show the behavior of the learning algorithm by applying it to two sample bases widely studied, Iris and Pima. The results will be compared with other learning algorithms C4.5,[15] CN2,[16] LVQ,[17] and the Wang and Mendel fuzzy rule learning process for classification problems, WM-FRLP.[8,2]

To do this, we organize this paper as follows. In Section 2, we explain the components of an FRBCS and the assumptions of MOGUL. Section 3 introduces the three stages of the genetic learning algorithm. In Section 4, the results

of the experiments as well as their analysis are shown. Finally, in Section 5, some conclusions are reported.

2. PRELIMINARIES

In this section, we describe the components of the FRBCS, the KB, and the FRM, and we briefly introduce MOGUL, the genetic learning methodology based on the iterative rule learning approach.[13,18]

2.1. Fuzzy Rule-Based Classification Systems

A pattern classification problem consists of assigning a class C_j from a predefined class set $C = \{C_1, \ldots, C_M\}$ to an object described as a point in a certain feature space $x \in S^N$.

The problem of designing a classification system is to find an optimal mapping

$$D: S^N \to C$$

in the sense of a certain criterion $\delta(D)$ that determines the classification system performance. This mapping is achieved by taking correctly classified examples, training examples, as a starting point, with the final goal being the design of a classifier that assigns class labels with the smallest possible error across the whole feature space. Finally, the system performance on testing data is calculated to have an estimation of the classification system true error.

Classification systems can be divided into two main groups depending on their later use: classification systems that are supposed to work autonomously, and those that are used as a helping tool for the user in the decision processes. In the former, the basic characteristic of the design process is the performance level, i.e., the correct classification percentage. Other dimensions, such as comprehensibility, robustness, versatility, modifiability, and coherence with previous knowledge, have to be considered in the latter, due to the fact that they may be essential to allow the system to be accepted for use.

The goal of the proposed learning method is to develop classification systems included in the second group, which can be considered "human-centered"[19], and the desired characteristics, cited previously, determine the structure of the fuzzy rules used as well as the process by which they are obtained. This is why we consider a descriptive FRBCS composed of a descriptive KB and a specific FRM. The KB composition is a set of fuzzy rules in which the linguistic variables involved in their antecedents have a term set of possible associated values that present a real-world meaning.

The FRBCS learning process, structure, and use are introduced in Figure 1.

In the next subsection, we will describe the components of the KB. In addition, we will introduce some of the FRMs analyzed by the authors[11] and used in the learning algorithm.

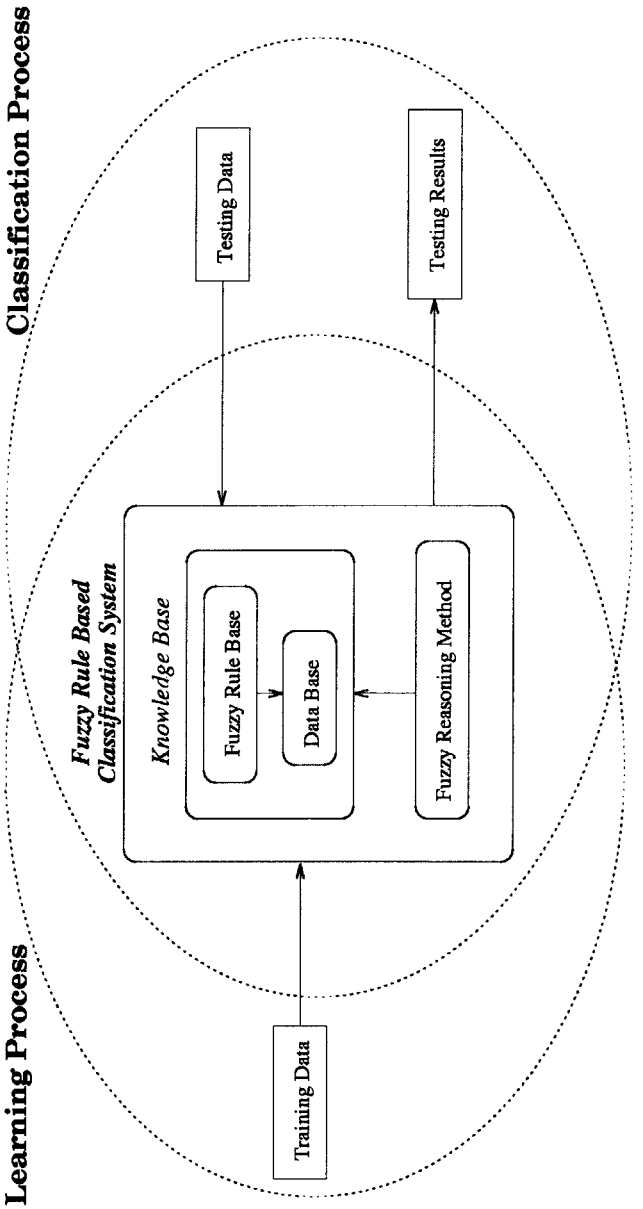

FIGURE 1. Fuzzy rule-based classification process from training to testing.

2.1.1. The Knowledge Base

As mentioned, the KB is composed of an RB and a DB. Their structures are explained as follows.

Rule Base. We can generate RBs with one of the following three types of rules:

(a) *Fuzzy rules with a class in the consequent*.[3,5] This kind of rule has the structure

$$R_k\text{: If } x_1 \text{ is } A_1^k \text{ and } \cdots \text{ and } x_N \text{ is } A_N^k\text{, then } Y \text{ is } C_j$$

where $x_1, \ldots, x_N$ are the outstanding selected features for the classification problem, $A_1^k, \ldots, A_N^k$ are linguistic labels used to discretize the continuous domain of the variables, and Y is the class C_j to which the pattern belongs.

(b) *Fuzzy rules with a class and a certainty degree in the consequent*[6]:

$$R_k\text{: If } x_1 \text{ is } A_1^k \text{ and } \cdots \text{ and } x_N \text{ is } A_N^k\text{, then } Y \text{ is } C_j \text{ with } r^k$$

where r^k is the certainty degree of the classification in the class C_j for a pattern belonging to the fuzzy subspace delimited by the antecedent. This certainty degree can be determined by the ratio

$$\frac{S_j^k}{S^k}$$

where S_j^k is the sum of the association degrees for the class C_j patterns belonging to the fuzzy region delimited by the if part. S^k is the same sum for patterns in any class.

(c) *Fuzzy rules with a certainty degree for all classes in the consequent*[7]:

$$R_k\text{: If } x_1 \text{ is } A_1^k \text{ and } \cdots \text{ and } x_N \text{ is } A_N^k\text{, then } \left(r_1^k, \ldots, r_M^k\right)$$

where r_j^k is the soundness degree for the rule k to predict the class C_j for a pattern belonging to the fuzzy region represented by the antecedent of the rule. This certainty degree can be determined by the same ratio used in the type (b) rules.

The last type of rule extends types (a) and (b) by using different values for $(r_1^k, \ldots, r_M^k)$. Considering

$$r_h^k = 1 \qquad r_j^k = 0 \qquad j \neq h \qquad j = 1, \ldots, M$$

we have the first case, and with

$$r_h^k = r^k \qquad r_j^k = 0 \qquad j \neq h \qquad j = 1, \ldots, M$$

we have the second.

In the rest of the paper, the developed learning algorithm will be presented considering an RB composed of (c) type rules. The process is analogous for the other two types of rules.

Data Base. The DB contains the definition of the fuzzy sets related to the linguistic terms used in the RB. This fact leads us to specify the next points:

- The number of linguistic terms for each variable considered.
- The membership function of the fuzzy sets related to these linguistic terms.

The number of linguistic terms as well as the membership functions of the associated fuzzy sets are specified by the FRBCS designer to obtain the suitable granularity level. It is known that the discretization carried out in the variable domain exerts a strong influence on the system classification capacity.[20] This fact, as we will explain, can be solved with the final tuning process in the proposed learning method.

On the other hand, it is difficult, even for an expert in the problem, to know exactly the most suitable meaning for a specific linguistic label and that most appropriate for proper system behavior. This is a determining factor in automatic knowledge extraction processes because the semantic representation of the linguistic hedges establishes a numerical value about the suitability of the concept the hedges represent. A way to solve this problem without losing the linguistic character of the classification system is to use linguistic hedges, which let us modify the prefixed membership function to adapt it to the training data. Zadeh[21] highlighted the utility of linguistic hedges for knowledge representation in approximate reasoning processes. In Refs. 1, 2, and 22, the reader can find some examples of linguistic modifiers used in FRBSs.

A linguistic hedge is a function that lets us alter the membership functions for the fuzzy sets associated to the linguistic labels, giving as a result a more or less precise definition depending on the case. Two of the most well known modifiers are the concentration linguistic modifier "very" and the dilation linguistic modifier "more or less." The first modifier leads to a reduction in the membership degree of a value in the fuzzy set to which it is applied. The second modifier "more or less," is a fuzzy dilation operator because, on the contrary, it increases the degree of membership. Their expressions are

$$\mu_{\text{very } A_i^k}(x) = \left(\mu_{A_i^k}(x)\right)^2$$

$$\mu_{\text{more or less } A_i^k}(x) = \sqrt{\mu_{A_i^k}(x)}$$

and their effects on a normalized fuzzy set with a triangular membership function are shown in Figure 2.

2.1.2. Fuzzy Reasoning Methods

As mentioned earlier, an FRM is an inference procedure that derives conclusion from a set of fuzzy if–then rules and a pattern. The power of fuzzy reasoning is that we can achieve a result even when we do not have an exact match (to a degree 1) between a system observation and the antecedents of the rules. In Ref. 11, we presented a general model of reasoning that involves

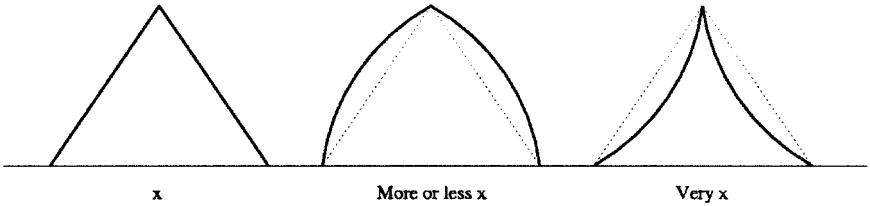

Figure 2. Linguistic hedges.

different possibilities as reasoning methods, and we proposed six alternative FRMs as some particular new proposals inside the general reasoning model. This model is the following.

Considering a new pattern $E^t = (e_1^t, \ldots, e_N^t)$ and a RB $R = \{R_1, \ldots, R_L\}$, the steps of the general reasoning model are as follows:

1. Matching degree. To calculate the *strength of activation of the if part for all rules in the RB with the pattern* E^t, using a t-norm[23,24],

$$R^k(E^t) = T\left(\mu_{A_1^k}(e_1^t), \ldots, \mu_{A_N^k}(e_N^t)\right) \qquad k = 1, \ldots, L$$

2. Association degree. To calculate the *association degree of the pattern* E^t *with the* M *classes according to each rule in the RB*,

$$b_j^k = h\left(R^k(E^t), r_j^k\right) \qquad j = 1, \ldots, M,\ k = 1, \ldots, L$$

3. Weighting function. To *weight* the obtained values through a function g. One possibility is to increase the higher values of the association degree and penalize the lower ones:

$$B_j^k = g\left(b_j^k\right) \qquad j = 1, \ldots, M,\ k = 1, \ldots, L$$

4. Pattern classification soundness degree for all classes. We use an aggregation function[23,24] that combines—for each class—the positive degrees of association calculated in the previous step and produces a system soundness degree for the classification of the pattern in this class:

$$Y_j = f\left(B_j^k,\ k = 1, \ldots, L \text{ and } B_j^k > 0\right) \qquad j = 1, \ldots, M$$

   with f being an aggregation operator verifying $\min \le f \le \max$. It is clear that if we select f as the maximum oeprator, we have the classical FRM.
5. Classification. We apply a decision function F over the soundness degree of the system for the pattern classification for all classes. This function will determine the class label l corresponding to the maximum value:

$$C_l = F(Y_1, \ldots, Y_M) \quad \text{such as} \quad Y_l = \max_{j=1,\ldots,M} Y_j$$

The general reasoning model is represented graphically in Figure 3.

Some of the proposed alternatives for the different operators in the FRM, are described in Table I.

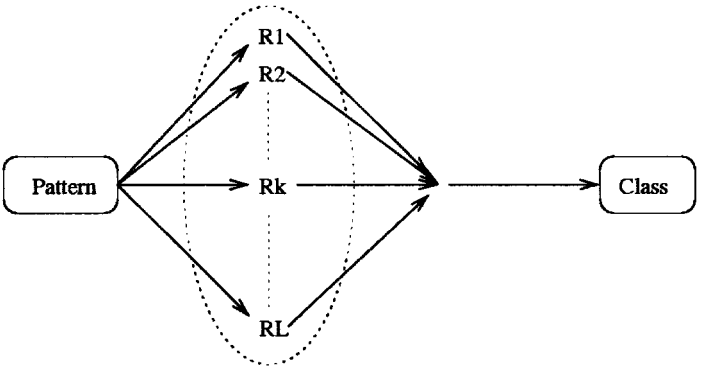

Figure 3. General fuzzy reasoning model.

Table I. Proposals for the operators in the FRM.[11]

1. *Compatibility Degree*	$R^k(E^t) = \min_{i=1,\ldots,N} \mu_{A_i}^k(e_i^t)$
2. *Association Degree*	$h(R^k(E^t), r_j^k) = R^k(E^t) \cdot r_j^k$
3. *Weighting Function: Alternatives*	$g_1(x) = x \quad \forall x \in [0,1]$
	$g_2(x) = \begin{cases} x^2 & \text{if } x < 0.5 \\ \sqrt{x} & \text{if } x \geq 0.5 \end{cases}$
4. *Aggregation*[a] *Function: Proposals*	
Normalized sum	$f_1(a_1,\ldots,a_{s_j^t}) = \dfrac{\sum_{i=1}^{s_j^t} a_i}{f_{1_{\max}}}$
	$f_{1_{\max}} = \max_{j=1,\ldots,M} \sum_{i=1}^{s_j^t} a_i$
Arithmetic mean	$f_2(a_1,\ldots,a_{s_j^t}) = \dfrac{\sum_{i=1}^{s_j^t} a_i}{s_j^t}$
Quasiarithmetic mean	$f_3(a_1,\ldots,a_{s_j^t}) = H^{-1}\left[\dfrac{1}{s_j^t}\sum_{i=1}^{s_j^t} H(a_i)\right]$
	$H(x) = x^p \quad p \in R$
Sowa and-like	$f_4(a_1,\ldots,a_{s_j^t}) = \alpha \cdot a_{\min} + (1-\alpha)\dfrac{1}{s_j^t}\sum_{i=1}^{s_j^t} a_i$
	$\alpha \in [0,1] \quad a_{\min} = \min\{a_1,\ldots,a_{s_j^t}\}$
Sowa or-like	$f_5(a_1,\ldots,a_{s_j^t}) = a \cdot a_{\max} + (1-\alpha)\dfrac{1}{s_j^t}\sum_{i=1}^{s_j^t} a_i$
	$\alpha \in [0,1] \quad a_{\max} = \max\{a_1,\ldots,a_{s_j^t}\}$
Badd	$f_6(a_1,\ldots,a_{s_j^t}) = \dfrac{\sum_{i=1}^{s_j^t} a_i^{\alpha+1}}{\sum_{i=1}^{s_j^t} a_i^{\alpha}} \quad \alpha \in R$

[a] $a_1,\ldots,a_{s_j^t}$ are the values to aggregate for an example E^t with regard to class C_j.

2.2. Basic Aspects of the Methodology: MOGUL

In Ref. 13, we presented MOGUL, a methodology consisting of some design guidelines that allow us to obtain different genetic fuzzy rule-systems (GFRBSs) able to cope with problems of different natures (fuzzy modelling, fuzzy control, and fuzzy classification, among others).

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

In this subsection, we are going to briefly review the key aspects of MOGUL, mainly focusing on the genetic learning approach and on the generic structure of the GFRBSs obtained using the methodology.

The main problem that has to be solved to design a GFRBS is finding a suitable representation capable of gathering the problem characteristics and representing the potential solutions. Classically, two genetic learning approaches, adopted from the field of genetic-based machine learning systems, have been used: the *Michigan*[25] and the *Pittsburgh*[26] approaches. In the Michigan approach, the chromosomes are individual fuzzy rules and the KB 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 KB 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 KB encoded in the chromosome.

In the last few years, a new genetic learning approach, iterative rule learning (IRL), has been proposed.[18] It is based on coding a single rule per chromosome, but, contrary to the Michigan approach, only the best individual in the GA run is considered to form part of the final KB. Therefore, in this approach the GA provides a partial solution to the problem of learning, and, contrary to both previous approaches, it is run several times to obtain the complete KB. Each time a new fuzzy rule is generated, the space zone in which it is located is penalized so that it will not be considered in subsequent runs. This operation mode substantially reduces the search space, because in each sequence of iterations only one rule is searched for. This allows us to obtain good solutions in GFRBSs for off-line learning problems. The IRL approach is followed by MOGUL.

The problem of the IRL approach is that the cooperation between the rules of the generated KB, a main characteristic of the FRBS because of the interpolative reasoning it develops, may not be as good as desired due to the fact that the iterative nature of the generation process does not envisage any relationship between the generated fuzzy rules. To solve this problem, GFRBSs

based on the IRL approach can use different tactics[27]:

- Adding new criteria to the evaluation of the rule to include this kind of cooperation within the IRL approach.
- Dividing the genetic learning process into at least two stages, thereby forming a multistage GFRBS. Therefore, the aim of the second stage (the postprocessing stage) is to obtain the best possible cooperation between the fuzzy rules generated in the first stage (the generation stage) to obtain the best possible KB.

Both tactics are considered in MOGUL, depending on the type of FRBS being designed (MOGUL allows us to work with different ones: descriptive Mamdani-type, approximate Mamdani-type, and TSK). When dealing with descriptive FRBSs, as in this paper, the second tactic is used. Usually, the second learning stage improves the cooperation level of the fuzzy rules generated in the previous one by refining them or by removing the redundant or unnecessary rules. With the aim of improving the accuracy of the FRBSs designed, in MOGUL we will tackle both tasks, the simplification of the KB and the refinement 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 multisimplification process* (genetic multiselection process in the GFRBS proposed in this paper, as we will show in the next sections) and the *evolutionary tuning process*. The former is capable of generating not only a single simplified KB definition as output from the process, but different definitions that present the best possible cooperation between the fuzzy rules composing them, and thereby the best possible behavior. Then, the evolutionary tuning process will be applied over these definitions and the most accurate will be the definition given as the output of the multistage GFRBS. Therefore, a KB that does not present the best behavior after the second stage may be the best after the third stage due to the fact that the new membership function shapes make its rules cooperate in a better way.

The remaining main aspects of MOGUL are the following:

- The designer is allowed to build the generation stage by using different kinds of algorithms, rather than only a GA, as in the previously existing processes following the IRL approach. It is possible to employ a nonevolutionary inductive algorithm (as in this paper) or an evolution strategy[28] 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 former case.
- Several important statistical properties have to be verified by the KB in order to obtain an FRBS that presents good behavior.[5] The satisfaction of *completeness* and *consistency* is considered in the GFRBSs obtained from MOGUL in order to improve the behavior of the generated KBs.
- Focusing on the EA search, there is a 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 to reduce the search space complexity and to perform an adequate exploration and exploitation over it to allow the search process to be effective. MOGUL proposes the use of many techniques.

- The available knowledge is incorporated into the genetic learning process in order to improve its performance by either directly incorporating partial definitions obtained from expert knowledge or using the available knowledge to generate the initial population of the EAs considered in each stage.

For more information about MOGUL, refer to Ref. 13.

3. GENETIC LEARNING PROCESS

The FRBCS genetic learning process is based on the MOGUL methodology presented in the previous section. As mentioned, it has been used here to design a descriptive FRBCS. To carry out this task, we have determined the algorithms for the different steps in the general methodology, including a linguistic hedge learning method for a better description of the system, and for the cooperation of the FRM in the KB design.

According to the methodology, the learning algorithm can be divided into three stages:

- A *fuzzy rule generation process*, which obtains a set of linguistic classification rules representing the knowledge existing in the training samples.
- A *genetic multiselection process*, which generates several KB definitions integrating the FRMs and learning the linguistic hedges.
- A *genetic tuning process*, where the best values for the membership functions parameters are obtained.

In the following subsection, we elaborate these three stages.

3.1. Fuzzy Rule Generation Process

This process has two components:

- A fuzzy rule generating method that generates the fuzzy rules from training examples.
- An iterative covering method that puts into effect the usual operation mode of the first phase of the evolutionary learning processes based on the IRL approach.

The next subsections show both methods.

3.1.1. Fuzzy Rule Generating Method

This method starts with a predefined DB, constituted of a uniform fuzzy partition with triangular membership functions crossing at height 0.5 for each variable. The system designer specifies the number of linguistic terms that form each partition, in order to obtain the desired granularity level. An example of this kind of partition for a linguistic variable with five labels is shown in Figure 4.

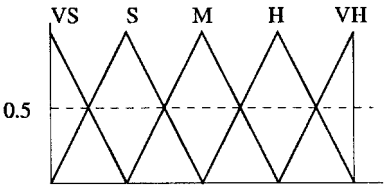

Figure 4. A uniform fuzzy partition with triangular membership functions.

Each time the fuzzy rule generating method is run, a set of candidate fuzzy rules is obtained. The method finds the fuzzy rules that best cover each example from the training set. The consequent creation is shown in Section 2. Therefore, a set of candidate fuzzy rules is created and the best rule is selected from it, according to a multicriteria function that considers the following criteria[29]:

(a) *A high frequency value*. The frequency of a classification rule R_k, in a set of examples $E = \{E^1, \ldots, E^p\}$, is defined as

$$\Psi_E(R_k) = \frac{\sum_{l=1}^{p} b_j^k(E^l)}{p}$$

since $E^l = (e_1^l, \ldots, e_N^l, C_j)$ and b_j^k is the association degree of the example E^l with the class C_j to which it belongs, according to the rule R_k.

(b) *A high covering degree over the positive examples*. The set of positive examples for a rule R_k with an association degree greater than or equal to ω is defined as

$$E_\omega^+(R_k) = \left\{E^l \in E \mid b_j^k(E^l) \geq \omega\right\}$$

where $N_\omega^+(R_k) = |E_\omega^+(R_k)|$. The average covering degree on $E_\omega^+(R_k)$ may be defined as

$$G_\omega(R_k) = \sum_{E^l \in E_\omega^+(R_k)} b_j^k / n_\omega^+(R_k)$$

(c) *Penalization associated to the nonsatisfaction of the k-consistency property*. The set of the negative examples for R_k is defined as

$$E^-(R_k) = \left\{E^l \in E \mid b_j^k(E^l) = 0 \text{ and } R^k(E^l) > 0\right\}$$

An example is considered negative for a rule when it best matches some other rule that has the same antecedent, but a different consequent. The negative examples are always considered over the complete training set.

This last criterion penalizes 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 nonsatisfaction of the k-con-

sistency property.[5] The *penalty function on the negative examples set of the rule* R_k will be

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

where $n_{R_k}^- = |E^-(R_k)|$ is the number of negative examples.

The three criteria are combined into an evaluation function using an aggregation function, increasing in its three components. The good behavior of the product operator was proven[10] using the expression

$$F(R_k) = \Psi_E(R_k) \cdot G_\omega(R_k) \cdot g_n(R_k^-)$$

The next algorithm summarizes the *fuzzy rule generating method*.

1. Set up the set of candidate rules to the empty set.
2. For each training example $E^l \in E$, generate the fuzzy rule R_k that best covers that example, having for each attribute the linguistic label that has the best matching. Include this rule in the set of candidate rules, B^c, if it has not been included before.
3. Evaluate all the fuzzy rules contained in B^c and select the one with the highest value of the rule selection function.

3.1.2. *Covering Method*

This method is based on an iterative algorithm that allows us to obtain a set of rules that cover the set of examples. In each iteration, it runs the *fuzzy rule generating method* to obtain the best fuzzy classification rule according to the current state of the training set, considers the relative covering value that this rule provokes over it, and removes from it the examples with a covering value greater than a value ϵ provided by the designer.

The *covering method* is developed as follows:

1. Initialization
   (a) Introduce the value of the main parameters of this method, i.e., k, ω, and ϵ.
   (b) Set up the examples covering degree to 0, i.e., $CV[l] \leftarrow 0$, $l = 1, \ldots, p$.
   (c) Set up the final set of rules B^g to the empty set.
2. Over the set of examples E, apply the *fuzzy rule generating method*, obtaining as output the best fuzzy classification rule R_r according to the current state of E.
3. Introduce the rule R_r in B^g.
4. For each example $E^l = (e_1^l, \ldots, e_N^l, C_j) \in E$ do
   (a) $CV[l] \leftarrow CV[l] + b_j^r(E^l)$.
   (b) If $CV[l] \ge \epsilon$, then remove it from E.
5. If $E = \varnothing$ then stop; else return to step 2.

3.2. The Multiselection Genetic Process

The method to obtain rules described previously generates a set of fuzzy rules that verifies the completeness and k-consistency properties.[13,5,30] However, due to its iterative nature, the resulting KB may have unnecessary or redundant rules that could cause an incorrect operation. To solve this problem, a multiselection genetic process is proposed. From the KB generated in the last stage, this process obtains different simplified KBs, with the best cooperation between the rules composing them.

This multiselection process includes

- The sequential niche technique[31] to induce niches, using as the basic optimization technique the genetic selection process proposed in Ref. 30, iterated in each run of the multiselection process.
- A search process that looks for the best set of modifiers associated to the linguistic labels of the variables.
- The intervention of the FRM used by the system in the rule and modifier selection.
- A local search posterior to each selection process, so that for the best individual, i.e., the best KB, it looks for the best modification, adding or eliminating a rule, and/or modifying a linguistic hedge.

Different KB definitions are obtained by selecting the rules that best cooperate from the initial fuzzy rule set and by selecting the best hedges for them by means of the abovementioned subprocesses.

In the following subsections, the genetic method and the composition of the multiselection process are analyzed.

3.2.1. The Basic Genetic Selection Method

The genetic selection process eliminates unnecessary rules from the RB obtained in the last phase and looks for the best set of hedges to modify these fuzzy rules. The learning of the hedges may be carried out from two different points of view:

- To obtain a hedge for each fuzzy set related to a linguistic label in the fuzzy partitions of the DB. In this case, this set of hedges is shared for all rules in the RB.
- To obtain the best set of hedges for each fuzzy rule in the RB.

In the first case, the semantic related to the linguistic variables is uniform for all rules and it is specified in the DB. In the second case, the meaning is specific for each individual rule, but it keeps the descriptive nature of the FRBCS. In the following text, the first kind of hedges will be referred to as Hedges I, and the latter as Hedges II.

The selection process is based on a GA in which the selection of the individuals is developed 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 with two outstanding parts, one related to the selected rules and the other referring to the hedges associated to the linguistic labels. Considering the rules contained in the rule set B^g derived from the previous step, counted from 1 to m, and depending on the hedge learning process that we want to carry out, there are two different coding schemes:

- *Hedges I*: The chromosome length is $h = m + \sum_{i=1}^{N} l_i$, where l_i is the number of linguistic labels for the variable i. A chromosome $C_j = (c_1, \ldots, c_h)$ is divided into two parts. The first part has as many bits as the number of rules generated in the previous phase, i.e., m bits. $c_1, \ldots, c_m$ represents a subset of candidate rules to form the RB finally obtained as this stage output, B^s, such that

$$\text{if } c_1 = 1, \text{ then } R_i \in B^s; \text{ else } R_i \notin B^s$$

  In addition, the second part will have as many genes as different linguistic terms are considered for each variable. For these genes, as many digits will be used as the number of different hedges to be considered. In Figure 5, this coding scheme and the resulting KB are described.
- *Hedges II*: The chromosome length is $h' = m \cdot (N + 1)$, where N is the number of variables. The chromosome is again divided into two parts. In the first part we follow the coding scheme in the last point. The $m \cdot N$ remaining genes represent the hedges for each of the rules. In Figure 6, this coding scheme as well as the type of resulting KB are described.

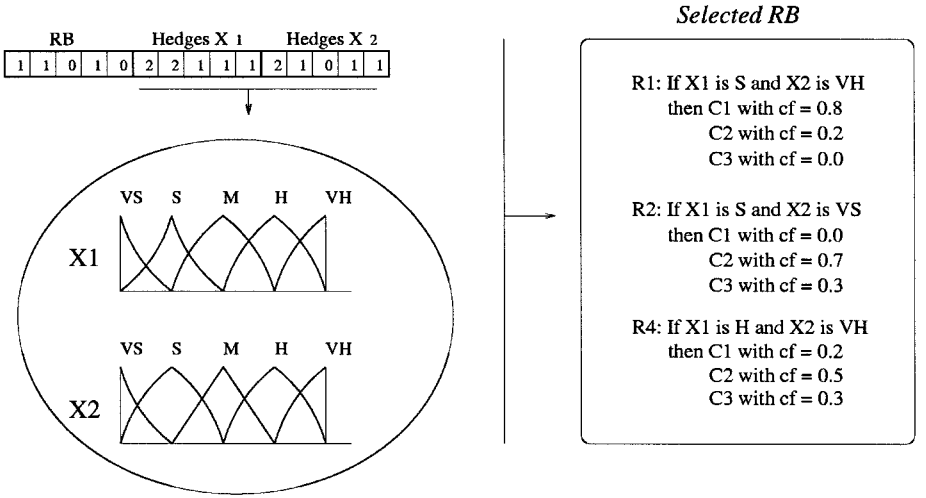

Figure 5. Hedges type I.

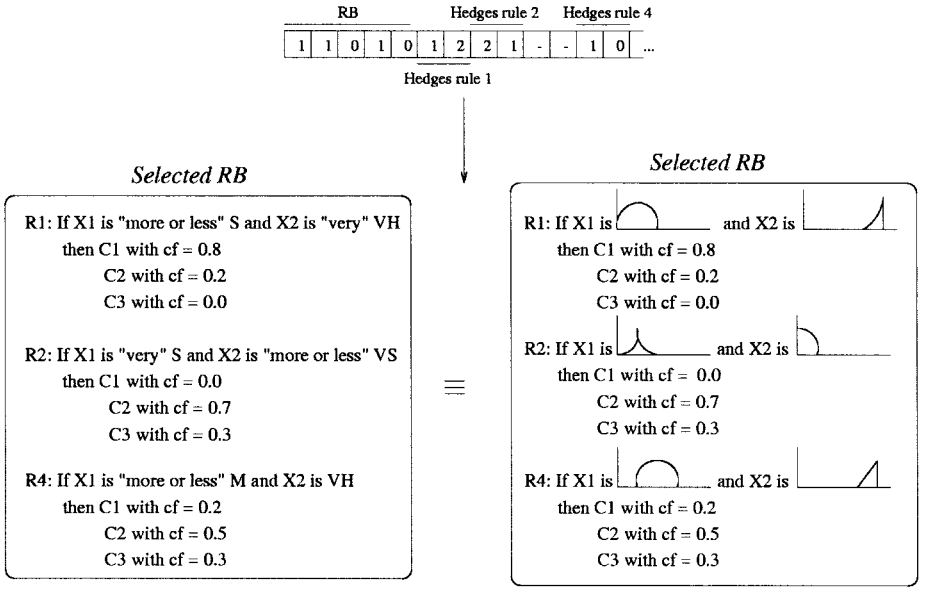

Figure 6. Hedges type II.

The initial population is generated by introducing a chromosome that represents the complete previously obtained rule set B^g, that is, with all $c_i = 1$ $i \in \{1, \ldots, m\}$, without hedges. For each type of hedge used, a chromosome that represents the complete RB and has all the genes that code the linguistic hedges with the value of the mentioned hedge is included. The remaining chromosomes are selected at random.

The fitness function, Error$(\cdot)$, is based on an application-specific measure usually employed in the design of the classifiers, the classifier's error rate[32] over a training data set E. An empirical error rate can be defined as the ratio of the number of errors to the number of cases examined:

$$\text{Error}(C_j) = \frac{\text{number of errors}}{\text{number of cases}}$$

There is a need to keep the *completeness property* considered in the previous stage. We ensure this condition by forcing every example $E^l = (e_1^l, \ldots, e_N^l, C_g)$ contained in the training set to be covered by the encoded KB, $R(C_j)$, to a degree greater than or equal to τ,

$$C_{R(C_j)}(E^l) = \bigcup_{j=1\cdots T} b_g^j(E^l) \geq \tau \qquad \forall E^l \in E \text{ and } R_j \in R(C_j)$$

where τ is the minimal training set completeness degree accepted in the selection process.

Therefore, we define a *training set completeness degree* (TSCD) of $R(C_j)$ over the set of examples E as

$$\mathrm{TSCD}\big(R(C_j), E\big) = \bigcap_{E^l \in E} C_{R(C_j)}(E^l)$$

and the final fitness function that penalizes the lack of the completeness property is

$$F\big(C_j\big) = \begin{cases} \mathrm{Error}\big(C_j\big) & \text{if } \mathrm{TSCD}\big(R\big(C_j\big), E\big) \geq \tau \\ 1 & \text{otherwise.} \end{cases}$$

3.2.2. The Multiselection Genetic Process

The multiselection genetic process takes as a base the sequential niche technique[31] to induce niches in the search space to obtain different KB definitions.[13] In each stage, the genetic selection process proposed in the last subsection is used.

Each time the genetic selection process obtains a new KB definition, the multiselection process penalizes the search space zone where it is located to not generate this definition in future runs. A genotypic sharing scheme[33] is used to penalize individuals according to their proximity to the previous solutions found. To do so, there is a need to define a *distance metric*, which, given two individuals, returns a value of how close they are. In Ref. 13, we proposed using the Hamming distance because we worked only with the first part of the chromosome, the part encoding the selected rules, and thus had a binary-coded chromosome. In the present case, chromosomes are not binary encoded because their second part encodes the linguistic hedges. Therefore, we propose to use the following distance function: With $A = (a_1, \ldots, a_h)$ and $B = (b_1, \ldots, b_h)$ being two individuals, the distance function is defined as

$$D(A, B) = \sum_{i=1}^{h} d_i$$

$$d_i = \begin{cases} 1 & \text{if } a_i \neq b_i \\ 0 & \text{otherwise} \end{cases}$$

Making use of this distance function, the *modified fitness function* that guides the search on the multiselection process is based on modifying the value

associated to an individual by the basic algorithm fitness function, multiplying it by a *derating function* that penalizes the closeness of this individual to the solutions previously obtained. Hence, the modified fitness function used by the multiselection process is

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

where F is the basic genetic selection process fitness function, $S = \{s_1, \ldots, s_k\}$ is the set containing the k solutions (KB definitions) yet found, and G is a kind of *derating function*. Taking into account the fact we are dealing with a minimization problem, we consider

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

where d is the minimum value of the distance between C_j and the solutions s_i included in S, i.e., $d = \text{Min}_i\{H(C_j, s_i)\}$, and the penalization is considered over the closest solution, r is the *niche radius*, and β is the *power factor* that determines the concavity ($\beta > 1$) or convexity ($\beta < 1$) of the derating curve. 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 penalty when C_j is far away from S in a value greater than or equal to the niche radius r.

Moreover, in addition to the original definition of the multiselection process previously introduced,[13] a local search algorithm is considered to individually optimize each one of the KB definitions obtained, inserting or eliminating a rule, and/or changing a hedge, changes that will lead to improve KB behavior. As may be observed, this is a very simple and quick optimization process.

The local search is carried out at the end of each iteration stage in the multiselection process. It is divided into two phases: First of all, the rule selection is optimized by means of a search in the RB space with distance 1 to the optimum, i.e., with one rule more or one less in the RB obtained as a result of one of the stages of the multiselection process. To reduce the search space when the RB part is optimized, the best set of hedges with distance 1 to the set of hedges that belongs to the KB represented by the optimum is looked for.

The algorithm of the genetic multiselection process follows:

1. Initializing. Equate the multiselection modified fitness function to the basic selection fitness function. $F'(C_j) \leftarrow F(C_j)$.
2. Run the basic genetic selection process using the modified fitness function and keep a record of the best individual found in the run.

3. Run the local optimization process to optimize the KB definition generated.
4. Update the modified fitness function to give a depression in the region near this individual, producing a new modified fitness function.
5. If all the simplified KBs desired have not been obtained, return to step 2.

Hence, the number of runs of the sequential algorithm performed (stages of the multiselection process) is the number of solutions to be obtained, i.e., the number of selected KBs to generate. We allows the FRBCS designer to decide this number as well as the values of the parameters r and β.

3.3. The Genetic Tuning Process

In this stage, the parameters that define the fuzzy set membership functions are optimized by means of a genetic tuning process.[13] This method, which is performed in a superior DB level to keep the linguistic approach of the resulting classification system, constitutes a solution to the problem of finding the search space partition that best represents the knowledge about the problem. Therefore, starting from a set of predefined fuzzy partitions—uniform and with triangular membership functions as shown in Figure 4, in our case—a new set of fuzzy partitions is found where the supports are modified in width and location.

Each chromosome forming the genetic population will encode a different DB definition that will be combined with the existing RB and hedge definitions to evaluate the individual adaptation. The GA designed for the tuning process presents real coding issue.[34] It uses the stochastic universal sampling as a selection procedure and Michaelewicz's nonuniform mutation operator. The max–min arithmetical crossover operator which makes use of fuzzy tools to improve the GA behavior, is employed.

As mentioned, the primary fuzzy sets considered in the initial fuzzy partitions are triangular shaped. Thus, each of the membership functions has an associated parametric representation based on a 3-tuple of real values, and a primary fuzzy partition can be represented by an array composed of $3 \cdot L$ real values, with L being the number of terms forming the linguistic variable term set. The complete DB for a problem in which N input linguistic variables are involved is encoded into a fixed length real coded chromosome C_r built by joining the partial representations of each of the variable fuzzy partitions as shown in the following:

$$C_{ri} = \left(a_{i1}, b_{i1}, c_{i1}, \ldots, a_{iL_i}, b_{iL_i}, c_{iL_i}\right)$$

$$C_r = C_{r1}C_{r2} \cdots C_{rN}$$

The initial gene pool is created by making use of the initial DB definition. This definition is encoded directly into a chromosome, denoted as C_1. The remaining individuals are generated by associating an interval of performance, $[c_h^l, c_h^r]$ to every gene c_h in C_1, $h = 1 \cdots \sum_{i=1}^{N} L_i \cdot 3$. Each interval of performance will be the interval of adjustment for the corresponding gene, $c_h \in [c_h^l, c_h^r]$.

If $(t \bmod 3) = 1$, then c_t is the left-hand value of the support of a fuzzy number. The fuzzy number is defined by the three parameters (c_t, c_{t+1}, c_{t+2}), and the intervals of performance are

$$c_t \in \left[c_t^l, c_t^r\right] = \left[c_t - \frac{c_{t+1} - c_t}{2}, c_t + \frac{c_{t+1} - c_t}{2}\right]$$

$$c_{t+1} \in \left[c_{t+1}^l, c_{t+1}^r\right] = \left[c_{t+1} - \frac{c_{t+1} - c_t}{2}, c_{t+1} + \frac{c_{t+2} - c_{t+1}}{2}\right]$$

$$c_{t+2} \in \left[c_{t+2}^l, c_{t+2}^r\right] = \left[c_{t+2} - \frac{c_{t+2} - c_{t+1}}{2}, c_{t+2} + \frac{c_{t+2} - c_{t+1}}{2}\right]$$

Figure 7 shows these intervals.

Therefore, we create a population of chromosomes containing C_1 as its first individual and the remaining ones initiated randomly, with each gene being in its respective interval of performance.

Finally, the fitness function is the same one used in the multiselection stage: if the KB is not complete to a τ degree, the function will be equal to 1. We should remember that the hedges and modifiers selected in the previous phase are involved in the computation of this function.

4. EXPERIMENTS

To analyze the performance of our proposal, we have applied the multistage genetic learning process to two well known sample sets: Iris and Pima.

To calculate an error estimation the FRBCS, we use, for both sample bases, random resampling[32] with five random partitions of each sample base on training and test sets, 70% and 30%, respectively. The outcomes shown in every table are means of correct classification percentages and number of rules, for all five training and for all five test sets, respectively.

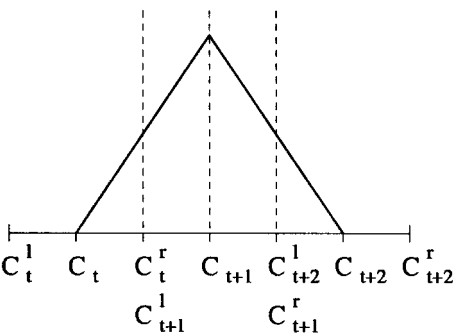

Figure 7. Intervals of performance.

We shall compare our results with those obtained with other algorithms based on learning techniques such as decision tress, C4.5 algorithm,[15] neural networks, LVQ,[17] inductive rules, CN2,[16] and FRBSs, WM-FRLP.[8,2]

The parameters used for the three processes in the experiments are shown in the following:

- *Generation process*: $\omega = 0.05$, $k = 0.1$, $\epsilon = 1.5$.
- *Multiselection process*:
  Number of generations, 500.
  Number of individuals, 61.
  τ, 0.1
  % of rules for the niche radius, 0.025.
  β (power factor), 0.5.
  Number of solutions, 3.
  Cross probability, 0.6.
  Mutation probaiblity, 0.1.
- *Tuning process*:
  Number of generations, 500.
  Number of individuals, 61.
  τ, 0.1.
  a (crossover parameter), 0.35.
  b (mutation parameter), 5.
  Cross probability, 0.6.
  Mutation probability, 0.1.

4.1. Iris Data

The Fisher Iris data are a set of 150 examples of iris flowers with four attributes and three classes (setosa, versicolor, and virginica). Taking into account the characteristics of this example set, we consider it interesting to use, as the initial DB, a fuzzy partition constituted of five triangular fuzzy sets. To show the performance of an FRBCS obtained using the proposed genetic learning process, as well as the influence of the fuzzy rule type and the FRM utilized to structure the knowledge about the problem, the organization of the experiments is explained in the following paragraphs:

1. We will obtain, by means of the *fuzzy rule generation process*, a RB of each of the considered type of rules (a), (b), and (c). Because this learning process stage is independent of the FRM used, the system performance will be computed with the different FRMs studied in Ref. 11. Columns 2–7 of the table in Appendix I show the results with all the FRMs; Table II presents the best results. The table in Appendix I also shows the final number of rules.
2. The postprocessing stages, i.e., the multiselection and tuning stages, which have as inputs the three types of RBs and the FRMs with the best behavior, will be run to obtain (a) a simplified RB, (b) a set of linguistic hedges, and (c) a set of membership function parameters, cooperating with the FRM.

As we mentioned earlier, this optimization proces will be carried out from two points of view: looking for the best set of hedges common to all fuzzy rules in the RBs, as well as the best set of hedges for each rule. Moreover, to analyze

Table II. Iris generation process.

Type (a) RB			Type (b) RB			Type (c) RB		
FRM	Training	Test	FRM	Training	Test	FRM	Training	Test
1	94.487	94.256	1	97.312	95.208	1	97.312	95.208
2	98.580	95.797	2	98.575	96.222	3	97.679	95.734
3	97.469	94.357	5	97.144	95.797	5	97.495	95.734
4	95.304	94.256	4	97.506	95.298	4	97.506	95.208

the bias introduced by the use of the hedges, the multiselection nd tuning processes will be run with the same RBs and FRMs, but in this case, without linguistic hedges. All the results are shown in Appendix II in the following form:

- The rows denoted S_i (with $i \in \{1, 2, 3\}$) correspond to the results obtained by the KB selected in the stage number i of the multiselection process.
- The rows indicated by T_i (with $i \in \{1, 2, 3\}$) show the classification results obtained by the tuned KB that was generated starting from the KB selected in stage i.

The best results are shown in Table III, in which the column denoted S-H (stage−hedges) describes whether the results are obtained in the multiselection stage (MS) or if the results correspond to the tuned FRBCS (T) for the linguistic hedges denoted H0, a *multiselection* or *tuning* process without use of the linguistic modifiers, H1, the postprocessing process with a Hedges I learning process, and H2, the process with a Hedges II learning process.

For the FRMs that produced the best results, we used the following notation:

1. Classical FRM.
2. FRM based on normalized sum.
3. FRM based on weighted normalized sum.
4. FRM based on quasiarithmetic mean ($p = 20$).
5. FRM based on weighted arithmetic mean.
6. FRM based on arithmetic mean.
7. FRM based on sowa or like ($\alpha = 0.7$).
8. FRM based on weighted quasiartihmetic mean ($p = 20$).

Table III. Iris postprocessing process.

Type (a) RB				Type (b) RB				Type (c) RB			
FRM	Training	Test	S-H	FRM	Training	Test	S-H	FRM	Training	Test	S-H
1	99.379	96.246	MS-II	1	99.644	95.657	T-II	1	98.764	95.758	T-II
2	99.649	95.797	T-I	2	99.649	96.710	MS-0	3	98.580	95.797	T-0
3	99.821	96.222	T-I	5	100	96.184	MS-II	5	98.757	95.309	T-0
4	99.438	95.169	MS-I	4	99.293	96.184	MS-I	4	98.401	96.184	T-II

Table IV. Other Iris algorithms.

Algorithm	Training	Test
C4.5	98.38	92.7
CN2	98.92	94.16
LVQ	98.55	95.72

In our experiments we use the word "weighted" (or W) to indicate use of the weighted function g_2 (see Section 2.1.2).

To establish the behavior of the proposed multistage learning process for the FRBCSs design, our results will be compared with those obtained by other learning algorithms such as C4.5,[15] CN2,[16], and LVQ[17] (see Table IV), and WM-FRLP[8,2] with three types of fuzzy if–then rules. The results of the WM-FRLP method with the FRMs that obtain the best classification percentages for the proposed learning method are shown in Table V.

The results indicate that the classification system obtained with our genetic learning method in cooperation with the alternative FRMs has a greater generalization ability, i.e., a higher percentage of success in the classification of test samples than the classification systems obtained by means of other algorithms. In addition, our classifiers present a greater interpretability.

An analysis of the results of our model leads to the following observations:

- The use of linguistic hedges causes an increasing cardinality of the RB (see Appendix II). The linguistic hedge *very* produces a reduction of the example's membership degree, which means that the RB must include a greater number of rules to be able to verify the completeness property.
- Learning a set of linguistic hedges for each fuzzy rule (Hedges II) may result in a KB that is overtuned to the training samples and lead to a loss in the classification system's generalization ability.
- The type (b) fuzzy rules are the most suitable structures, for this method, to represent the knowledge that we can extract from the training sets. The antecedent of a fuzzy rule represents a fuzzy area in the search space, and if the consequent describes information about all the classes, noise could be introduced into the classification process for the other classes due to another antecedent, i.e., another fuzzy area, could better represent the knowledge for those classes.
- The selection mechanism in the genetic stages of the learning process—multiselection and tuning—might mean that, among several individuals with the same percentage of success in the training examples and different success in the tests,

Table V. Iris WM-FRLP algorithm.

Type (a) RB			Type (b) RB			Type (c) RB		
FRM	Training	Test	FRM	Training	Test	FRM	Training	Test
1	90.97	88.25	1	97.31	94.32	1	96.96	94.32
2	97.29	92.88	2	96.43	93.20	3	96.95	93.40
3	98.56	94.38	5	96.23	93.89	5	96.25	92.72
4	91.18	90.34	4	97.31	94.32	4	96.96	94.32

one that has a worse behavior in the testing data may be selected. The reason is that the selection of individuals with the same value for the fitness function is carried out using the position in the ordering of the population.

- We notice that the best results are obtained for an FRBCS composed of a type (b) RB. These results are obtained before the tuning process. As mentioned, the generation process of these types of rules extracts knowledge from the problem in a more accurate way. Therefore, the tuning of the membership function parameters for the FRBCS with a type (b) RB may not be significant.
- On the contrary, for those RBs with structure (c), the tuning process is very important because it allows us to tune the fuzzy area represented by the rule antecedent, so that the multiple consequent introduces the minimum amount of noise in the classification process.
- In type (a) rules, which only show information about the class label that is presented in a greater proportion in that area of the space, the tuning process or the hedge learning for each rule affords a greater precision to the FRBCS.

4.2. Pima Data

Pima is a set of 768 cases involving the diagnosis of diabetes, where eight variables are taken into account, with two possible classes (having or not having the complaint). Considering the characteristics of this sample set, i.e., the number of variables and their domain, among others, we considered interesting to use a fuzzy partition constituted by three triangular fuzzy sets as the initial DB.

The results from the experiments on the Iris sample base have shown that the type (b) RB, as a knowledge representation structure, has the best behavior. That is why the experiments on the Pima sample base are oriented to developing FRBCSs with type (b) RB.

The best results obtained in the different stages of the genetic learning process are shown in Table VI, according to the best results obtained in the generation process by the different FRMs (see columns 8 and 9 in the table in Appendix I). Using the remaining algorithms, the results obtained for this sample base are as shown in Table VII. Again, note that the FRBCS obtained using the multistage genetic learning process has a greater percentage of correct classifications in the testing samples than the classification systems obtained with the C4.5, CN2, and LVQ techniques and the WM-FRLM.

Table VI. Classification results for Pima with a type (b) RB.

Generation Process			Postprocessing Processes		
FRM	Training	Test	Training	Test	S-H
1	74.872	73.668	81.806	74.061	T-0
3	76.161	74.488	83.272	75.811	MS-II
7	74.037	73.872	80.866	75.676	MS-I
8	74.769	73.564	78.144	74.173	MS-0

Table VII. Classification results for Pima with other learning algorithms.

Algorithm	Training	Test
C4.5	96.06	71.4
CN2	85.4	74.5
LVQ	83.68	67.71
WM-FRLP		
FRM 1	85.81	73.23
FRM 3	83.93	72.60
FRM 7	85.88	72.81
FRM 8	85.85	73.23

5. CONCLUSIONS

In this work, we described a genetic learning process for obtaining linguistic FRBCSs that integrates FRMs cooperating with the KBs. Furthermore, this process learns the best set of linguistic hedges for the linguistic variables terms thus maintaining the descriptive capabilities of the classification system.

We analyze the behavior of different types of fuzzy rules and the FRMs in the learning of the FRBCS using our genetic learning process. The most powerful type of fuzzy rule to extract the knowledge is the rule that the consequent must indicate a class and the associated certainty degree. With an FRBCS with this type of rule, the best results are obtained.

As we know, the FRMs can be classified into two categories, i.e., those which use a single rule to classify and those based on a combination of information provided by different rules. Methods in the first category classify only with the best rule, so they consider the rules as intervals and ignore the information provided by overlapped fuzzy subsets. The use of an FRM that considers the information given by all the rules in the FRBS design process increases the generalization ability of the resulting system. Nevertheless, it cannot determine a unique FRM as the best suitable for any type of problem, so it will be necessary to carry out a small study on the different FRMs' behavior to obtain the best FRBCS to solve it.

As future work, we intend to extend the multistage genetic learning process to design a new kind of FRBCS, the approximate one, in which the antecedent part of the fuzzy rule presents an approximate nature[13], and to design FRBCSs with fuzzy rules in a disjunctive normal form where each linguistic variable may have different linguistic valus associated in the same rule.[5,35]

This research was supported by CICYT TIC 96-0778.

References

1. A. Bárdossy and L. Duckstein, *Fuzzy Rule-Based Modeling with Applications to Geophysical, Biological and Engineering Systems*, CRC Press, Boca Raton, FL, 1995.
2. Z. Chi, H. Yan, and T. Pham, *Fuzzy Algorithms with Application to Image Processing and Pattern Recognition*, World Scientific, Singapore, 1996.
3. S. Abe and R. Thawonmas, "A fuzzy classifier with ellipsoidal regions," *IEEE Trans. Fuzzy Systems*, **5**, 358–368 (1997).

4. L. Fu, "Rule generation from neural networks," *IEEE Trans. Systems, Man Cybernetics*, **24**, 1114–1124 (1994).
5. A. González and R. Pérez, "Completeness and consistency conditions for learning fuzzy rules," *Fuzzy Sets Systems*, **96**, 37–51, 1998.
6. H. Ishibuchi, K. Nozaki, and H. Tanaka, "Distributed representation of fuzzy rules and its application to pattern classification," *Fuzzy Sets Systems* **52**, 21–32 (1992).
7. D.P. Mandal, C.A. Murthy, and S.K. Pal, "Formulation of a multivalud recognition system," *IEE Trans. Systems, Man Cybernetics*, **22**, 607–620 (1992).
8. Z. Chi, J. Wu, and H. Yan, "Handwritten numeral recognition using self-organizing maps and fuzzy rules," *Pattern Recognition*, **28**, 59–66 (1995).
9. L.I. Kuncheva, "On the equivalence between fuzzy and statistical classifiers," *Int. J. Uncertainty, Fuzziness Knowledge-Based Systems*, **15**, 245–253 (1996).
10. O. Cordón, M.J. del Jesus, F. Herrera, and E. López, "Selecting fuzzy rule based classification systems with specific reasoning methods using genetic algorithms," *Proc. 7th Int. Fuzzy Systems Assoc. World Congress*, Prague, 1997, pp. 424–429.
11. O. Cordón, M.J. del Jesus, and F. Herrera, *A Proposal on Reasoning Methods in Fuzzy Rule-Based Classification Systems*, Tech. Rep. DECSAI-970126, Dept. Computer Science and Artificial Intelligence, Univ. of Granada, November 1997.
12. H. Ishibuchi, T. Morisawa, and T. Nakashima, "Voting schemes for fuzzy-rule-based classification systems," *Proc. 5th IEEE Int. Conf. on Fuzzy Systems*, 1996, pp. 614–620.
13. O. Cordón, M.J. del Jesus, F. Herrera, and M. Lozano, *MOGUL: A Methodology to Obtain Genetic Fuzzy Rule-Based Systems under the Iterative Rule Learning Approach*, Tech. Rep. DECSAI-980101, Dept. Computer Science and Artificial Intelligence, University of Granada, January 1998.
14. D.E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, Reading, MA, 1989.
15. J.R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann, San Mateo, CA, 1993.
16. P. Clark and T. Niblett, *Learning If Then Rules in Noisy Domains*, Rep. TIRM 86-019, The Turing Institute, Glasgow, 1986.
17. T. Kohonen, *Self-Organizing Maps*, Series in Information Sciences 30, Springer-Verlag, Berlin, 1995.
18. A. González and F. Herrera, "Multi-stage genetic fuzzy systems based on the iterative rule learning approach," *Mathware and Soft Computing*, **4**, 233–249 (1997).
19. A.K. Nath and T.T. Lee, "On the design of a classifier with linguistic variables as inputs," *Fuzzy Sets Systems*, **11**, 265–286 (1983).
20. H. Ishibuchi, K. Nozaki, and H. Tanaka, "Efficient fuzzy partition for pattern space for classification problems," *Fuzzy Sets Systems*, **59**, 295–304 (1983).
21. L.A. Zadeh, "The concept of a linguistic variable and its applications to approximate reasoning, Parts I–III," *Inform. Sci.* **8**, 199–249; 301–357; **9**, 43–80 (1975).
22. S. Mitra and S.K. Pal, "Fuzzy self organization, inferencing, and rule generation," *IEEE Trans. Systems, Man Cybernetics*, **26**, 608–619 (1996).
23. C. Alsina, E. Trillas, and L. Valverde, "On some logical connectives for fuzzy set theory," *J. Math. Anal. Appl.*, **93**, 149–163 (1983).
24. D. Dubois and H. Prade, "A review of fuzzy set aggregation connectives," *Inform. Sci.*, **36**, 85–121 (1985).
25. J.H. Holland and J.S. Reitman, *Cognitive Systems Based on Adaptive Algorithms*, in *Pattern-Directed Inference Systems*, D.A. Waterman and F. Hayes-Roth, Eds., Academic Press, New York, 1978.
26. S.F. Smith, "A learning system based on genetic adaptive algorithms," Ph.D. Thesis, Univ. of Pittsburgh, 1980.
27. O. Cordón, A. González, F. Herrera, and R. Pérez, "Encouraging cooperation in the genetic iterative rule learning approach for qualitative modeling," in *Computing with Words in Information/Intelligent Systems*, J. Kacprzyk and L. Zadeh, Eds., Physical-Verlag, Heidelberg, 1998.

28. T. Back, *Evolutionary Algorithms in Theory and Practice*, Oxford Univ. Press, 1996.
29. O. Cordón and F. Herrera, "A three-stage evolutionary process for learning descriptive and approximative fuzzy logic controller knowledge bases from examples," *Int. J. Approximate Reasoning*, **17**, 369–407 (1997).
30. F. Herrera, M. Lozano, and J.L. Verdegay, "A learning process for fuzzy control rules using genetic algorithms," *Fuzzy Sets Systems* (1998), to appear.
31. D. Beasly, D.R. Bull, and R.R. Martin, "A sequential niche technique for multimodal function optimization," *Evolutionary Computation*, **1**, 101–125 (1993).
32. S.M. Weiss and C.A. Kulikowski, *Computer Systems That Learn*, Morgan Kaufmann, San Mateo, CA, 1991.
33. K. Deb and D.E. Goldberg, "An investigation of niche and species formation in genetic function optimization," *Proc. 3rd Int. Conf. on Genetic Algorithms* (ICGA'89), Hillsdale, 1989, pp. 42–50.
34. F. Herrera and M. Lozano, "Tackling real-coded genetic algorithms: Operators and tools for the behavioural analysis," *Artificial Intelligence Rev.*, **12**, 265–319, 1998.
35. L. Magdalena and F. Monasterio-Huelin, "Fuzzy rules-based controllers that learn by evolving its knowledge base," *Int. J. Approximate Reasoning*, **16**, 335–358 (1997).

APPENDIX I

Classification results of the FRBCSs obtained in the generating process with different FRMs.

	Iris Type (a) RB (61.8 Rules)		Iris Type (b) RB (70 Rules)		Iris Type (c) RB (64.4 Rules)		PIMA Type (b) RB (279.6 Rules)	
FRM Based on	TRA	Test	TRA	Test	TRA	Test	TRA	Test
Classic	95.487	94.256	97.312	95.208	97.312	95.208	74.873	73.668
Normalized sum	94.580	95.797	98.575	96.222	96.750	94.093	76.754	73.960
W. normalized sum	97.469	94.357	98.224	95.797	97.680	95.734	76.161	74.488
Arithmetic mean	92.157	91.779	97.495	95.371	96.566	93.668	71.178	70.057
W. arithmetic mean	94.202	93.280	97.144	95.797	97.496	95.734	71.981	70.780
Sowa or like 0.3	93.835	93.280	97.500	95.270	97.496	95.734	73.060	71.910
W. Sowa or like 0.3	94.575	93.280	97.327	95.208	97.690	95.208	73.652	72.946
Sowa or like 0.5	94.564	93.280	97.495	95.208	97.862	95.208	73.513	73.044
W. Sowa or like 0.5	94.753	93.280	97.323	95.208	97.506	95.208	74.072	73.566
Sowa or like 0.7	94.937	94.256	97.323	95.208	97.506	95.208	74.037	73.872
W. Sowa or like 0.7	94.753	94.256	97.506	95.208	97.506	95.208	94.595	73.461
Sowa or like 0.8	94.937	94.256	97.506	95.208	97.506	95.208	74.490	73.461
W. Sowa or like 0.8	94.753	94.256	97.506	95.208	97.506	95.208	74.664	73.461
Sowa or like 0.9	94.937	94.256	97.506	95.208	97.506	95.208	74.664	73.461
W. Sowa or like 0.9	94.753	94.256	97.506	95.208	97.506	95.208	74.803	73.668
Quasiarithmetic mean 10	94.937	93.281	97.328	95.208	97.506	95.208	74.280	72.946
W. quasiarithmetic mean 10	94.586	94.256	97.506	95.208	97.506	95.208	74.629	73.564
Quasiarithmetic mean 20	95.304	94.256	97.506	95.208	97.506	95.208	74.629	73.564
W. quasiarithmetic mean 20	94.586	94.256	97.506	95.208	97.506	95.208	74.769	73.564
Badd 10	94.758	94.256	97.506	95.208	97.506	95.208	75.115	73.564
W. badd 10	94.937	94.256	97.312	95.208	97.312	95.208	75.011	73.668
Badd 20	95.293	94.256	97.312	95.208	97.312	95.208	75.011	73.668
W. badd 20	95.120	94.256	97.312	95.208	97.312	95.208	74.838	73.769

APPENDIX II

Classification results of the FRBCSs obtained in the postprocessing processes with different FRMs.

Without Hedges				Hedges I				Hedges II			
	Training	Test	NR		Training	Test	NR		Training	Test	NR
1. Iris Type (a) RB and Classic FRM											
S_1	97.663	94.357	30	S_1	99.115	93.769	39.8	S_1	99.379	94.643	44.4
S_2	97.663	93.831	39.4	S_2	99.115	94.256	39.2	S_2	99.379	96.246	49.2
S_3	97.663	94.256	38	S_3	99.115	94.744	39.4	S_3	99.644	93.242	46.6
T_1	99.293	94.367	30	T_1	99.638	95.333	39.8	T_1	98.821	93.405	44.4
T_2	99.293	92.979	39.4	T_2	99.817	94.256	39.2	T_2	99.644	94.256	49.2
T_3	99.465	92.855	38	T_3	99.816	94.907	39.4	T_3	99.644	92.352	46.6
2. Iris Type (a) RB and FRM Based on Weighted Normalized Sum											
S_1	99.465	95.634	41.4	S_1	99.828	94.782	43.4	S_1	100	94.256	50.8
S_2	99.465	95.309	38	S_2	99.638	95.696	40.6	S_2	99.828	95.309	50
S_3	99.465	94.256	38.8	S_3	99.649	95.270	43.8	S_3	100	92.879	51.4
T_1	99.649	94.782	41.4	T_1	99.828	93.769	43.4	T_1	100	93.831	50.8
T_2	99.466	95.208	38	T_2	99.821	96.222	40.6	T_2	100	93.831	50
T_3	99.466	93.242	38.8	T_3	99.649	94.782	43.8	T_3	100	93.042	51.4
3. Iris Type (a) RB and FRM Based on Normalized Sum											
S_1	99.465	95.634	41	S_1	99.649	94.194	43.8	S_1	100	94.194	51.8
S_2	99.465	95.108	40.6	S_2	99.649	95.309	44.6	S_2	100	93.203	53.2
S_3	99.465	94.156	41.6	S_3	99.465	95.332	45	S_3	100	93.730	52.8
T_1	99.465	94.782	41	T_1	99.649	94.357	43.8	T_1	100	93.831	51.8
T_2	99.465	95.634	40.6	T_2	99.649	95.797	44.6	T_2	100	92.716	53.2
T_3	99.466	94.395	41.6	T_3	99.649	95.797	45	T_3	100	92.290	52.8
4. Iris Type (a) RB and FRM Based on Quasiarithmetic Mean ($p = 20$)											
S_1	97.663	93.341	36.4	S_1	99.438	95.169	40.8	S_1	99.816	93.281	48.4
S_2	97.663	93.869	37	S_2	99.477	92.290	39.4	S_2	100	93.831	50.6
S_3	97.663	93.242	37.4	S_3	99.649	94.782	38.8	S_3	99.644	93.241	46.8
T_1	99.293	93.970	36.4	T_1	99.644	94.256	40.8	T_1	100	93.768	48.4
T_2	99.293	94.782	37	T_2	99.293	93.204	39.4	T_2	100	94.907	50.6
T_3	99.993	93.831	37.4	T_3	99.649	94.179	38.8	T_3	100	92.731	46.8
5. Iris Type (b) RB and Classic FRM											
S_1	98.569	94.720	40.6	S_1	99.293	94.357	45.4	S_1	99.644	94.480	47.4
S_2	98.569	93.606	39	S_2	99.477	93.668	44.8	S_2	99.816	91.113	46.2
S_3	98.569	95.208	43.8	S_3	99.649	93.180	44.8	S_3	100	94.031	51.2
T_1	99.114	93.281	40.6	T_1	99.472	93.444	45.4	T_1	99.644	95.657	47.4
T_2	99.114	93.281	39	T_2	99.477	93.343	44.8	T_2	99.817	91.601	46.2
T_3	99.465	94.782	43.8	T_3	99.649	93.281	44.8	T_3	100	94.256	51.2
6. Iris Type (b) RB and FRM Based on Normalized Sum											
S_1	99.466	94.620	45.6	S_1	99.828	93.280	49.8	S_1	99.828	94.744	53.2
S_2	99.649	96.710	48	S_2	99.465	95.309	47.4	S_2	100	93.281	58.6
S_3	99.649	95.208	47.4	S_3	99.655	94.295	48.6	S_3	100	94.156	60.8
T_1	99.466	94.256	45.6	T_1	99.828	92.855	49.8	T_1	99.828	93.730	53.2
T_2	99.649	96.222	48	T_2	99.465	95.309	47.4	T_2	100	94.357	58.6
T_3	99.649	95.208	47.7	T_3	99.649	93.845	48.6	T_3	100	94.256	60.8

Classification results of the FRBCSs obtained in the postprocessing processes with different FRMs. (*Continued*)

Without Hedges				Hedges I				Hedges II			
	Training	Test	NR		Training	Test	NR		Training	Test	NR
7. Iris Type (b) RB and FRM Bassed on Weighted Arithmetic Mean											
S_1	99.470	94.720	43.2	S_1	99.649	94.845	43.6	S_1	100	96.184	49.2
S_2	99.470	94.094	41	S_2	99.821	93.242	44.6	S_2	100	93.954	44.6
S_3	99.643	94.782	40	S_3	99.821	93.730	43.6	S_3	100	93.668	52.6
T_1	99.649	94.357	43.2	T_1	99.649	94.232	43.6	T_1	100	95.208	49.2
T_2	99.828	95.758	41	T_2	99.821	92.917	44.6	T_2	100	93.567	44.6
T_3	99.821	93.869	40	T_3	99.821	93.831	43.6	T_3	100	93.668	52.6
8. Iris Type (b) RB and FRM Based on Quasiarithmetic Mean ($p = 20$)											
S_1	98.764	93.706	38.4	S_1	99.293	94.682	44	S_1	99.649	94.342	52.8
S_2	98.764	94.194	41	S_2	99.293	96.184	44.4	S_2	100	94.605	45.2
S_3	98.764	94.682	43.8	S_3	99.465	93.730	49	S_3	99.649	96.184	48.2
T_1	99.293	92.368	38.4	T_1	99.293	94.245	44	T_1	99.649	94.845	52.8
T_2	99.293	93.706	41	T_2	99.821	92.917	44.4	T_2	100	93.567	45.2
T_3	99.293	94.266	43.8	T_3	99.821	93.831	49	T_3	100	93.668	48.2
9. Iris Type (c) RB and Classic FRM											
S_1	97.851	94.682	42	S_1	98.380	93.768	40.6	S_1	98.563	95.533	49.6
S_2	97.851	93.644	37	S_2	98.397	94.031	40.2	S_2	98.569	95.633	41.4
S_3	97.851	94.682	36.6	S_3	98.558	93.730	41	S_3	98.585	96.184	50.6
T_1	98.218	94.256	42	T_1	98.574	94.782	40.6	T_1	98.746	93.544	49.6
T_2	98.397	94.194	37	T_2	98.397	94.194	40.2	T_2	98.931	95.270	41.4
T_3	98.401	93.281	36.6	T_3	98.472	94.295	41	T_3	98.764	95.758	50.6
10. Iris Type (c) RB and FRM Based on Weighted Normalized Sum											
S_1	98.401	85.208	40.6	S_1	98.942	93.080	41.8	S_1	99.115	93.343	48.4
S_2	98.580	95.634	37.6	S_2	98.580	93.343	41.6	S_2	99.115	93.304	49.4
S_3	98.752	95.633	39.4	S_3	99.108	94.782	38.2	S_3	98.757	94.782	50.2
T_1	98.585	93.304	40.6	T_1	99.115	92.066	41.8	T_1	99.287	93.807	48.4
T_2	98.580	95.797	37.6	T_2	99.115	95.309	41.6	T_2	99.115	93.768	49.4
T_3	98.925	95.309	39.4	T_3	99.108	94.845	38.2	T_3	98.930	94.782	50.2
11. Iris Type (c) RB and FRM Based on Weighted Arithmetic Mean											
S_1	98.401	95.208	39	S_1	98.759	93.304	43.2	S_1	99.287	95.270	47.2
S_2	98.580	94.682	40	S_2	99.114	93.792	39.2	S_2	98.942	94.720	53.4
S_3	98.401	94.093	41.2	S_3	98.752	94.256	42.2	S_3	99.287	94.682	47.8
T_1	98.757	95.309	39	T_1	98.942	93.831	43.2	T_1	99.287	94.782	47.2
T_2	98.764	94.782	40	T_2	99.115	92.252	39.2	T_2	98.942	94.295	53.4
T_3	98.585	94.821	41.2	T_3	98.931	94.357	42.2	T_3	99.287	94.256	47.8
12. Iris Type (c) RB and FRM Based on Quasiarithmetic Mean ($p = 20$)											
S_1	98.046	94.093	38.2	S_1	98.401	94.093	43.4	S_1	98.936	94.744	47.2
S_2	98.046	94.782	39.6	S_2	98.752	92.754	41.6	S_2	98.569	92.615	49.4
S_3	98.046	94.720	35.8	S_3	98.574	93.606	41.8	S_3	98.401	94.581	52
T_1	98.574	93.668	38.2	T_1	98.757	94.682	43.4	T_1	99.287	93.831	47.2
T_2	98.757	94.682	39.6	T_2	98.931	93.730	41.6	T_2	98.920	93.180	49.4
T_3	98.401	94.295	35.8	T_3	98.574	93.180	4.18	T_3	98.401	96.184	52