SGERD: A Steady-State Genetic Algorithm for Extracting Fuzzy Classification Rules From Data

Eghbal G. Mansoori, Mansoor J. Zolghadri, and Seraj D. Katebi

***Abstract*—This paper considers the automatic design of fuzzy-rule-based classification systems from labeled data. The performance of classifiers and the interpretability of generated rules are of major importance in these systems. In past research, some genetic-based algorithms have been used for the rule learning process. These genetic fuzzy systems have utilized different approaches to encode rules. In this paper, we have proposed a novel steady-state genetic algorithm to extract a compact set of good fuzzy rules from numerical data (SGERD). The selection mechanism of this algorithm is nonrandom, and only the best individuals can survive. Our approach is very simple and fast, and can be applied to high-dimensional problems with numerical attributes. To select the rules having high generalization capabilities, our algorithm makes use of some rule- and data-dependent parameters. We have also proposed an enhancing function that modifies the rule evaluation measures in order to assess the candidate rules more effectively before their selection. Experiments on some well-known data sets are performed to show the performance of SGERD.**

***Index Terms*—Data mining, fuzzy-rule-based classification system, fuzzy rule learning, steady-state genetic algorithm.**

I. Introduction

A FUZZY-RULE-BASED classification system is a special case of fuzzy modeling, where the output of the system is crisp and discrete. The fuzzy models present two main advantages: first, they permit working with imprecise data and also provide a comfortable way to naturally represent the missing values, and second, the acquired knowledge with these models may be more human-understandable. In designing fuzzy-rule-based systems, two conflicting objectives are addressed: the accuracy of systems and the interpretability of fuzzy rules. In practice, one of these objectives prevails over the other. Nevertheless, a new tendency to look for a good balance between the interpretability and the accuracy is increasing recently. In this research, improving the interpretability of rule-based systems is a central issue, while the accuracy and compaction of the obtained rules are also receiving attention [1]. Some studies have discussed the tradeoff between these two conflicting objectives [2]–[4].

Many approaches have been proposed for generating and learning fuzzy classification rules from numerical data. These include simple heuristic procedures [5], [6], neuro-fuzzy techniques [7], [8], clustering methods [9], and genetic algorithms [10]–[12]. Additionally, some approaches combining mentioned objectives are also reported, for example, applying linguistic constraints to fuzzy modeling [13], data reduction via evolutionary stratified instance selection [14], and using genetic algorithms for iteratively developing fuzzy classifiers [15]. Multi-objective genetic algorithms are also capable of capturing a set of nondominated solutions while including both the accuracy and the interpretability and compactness [4], [16]–[19].

Genetic algorithms (GAs) are search algorithms that use operations found in natural evolution to guide, seeking the search space. GAs have been theoretically and empirically proven to provide robust search capabilities in complex spaces, offering a valid approach to problems requiring an efficient and effective searching. The basic idea is to maintain a population of chromosomes (representing candidate solutions to the concrete problem being solved) that evolves over time through a process of competition and controlled variation. Two distinct approaches in which GAs have been applied to learning processes are the Pittsburgh [20], [21] and the Michigan [22], [23] approaches. In the first approach, each individual encodes a complete rule set of classifiers, whereas in the second approach, each individual in the population corresponds to classifier rules that are evolved as a whole. Similar to the Michigan approach is the cooperative–competitive approach, in which the complete population or a subset of it codifies the rule base [24], [25]. The main intention in this approach is to develop a fuzzy-rule-based system that accompanies the cooperation among rules with competition between them [26]. A third alternative, the iterative approach [27], is presented in which each individual represents a single rule.

The iterative approach attempts to reduce the search space for possible solutions. In this approach, as in Michigan, each individual represents a single rule, but contrary to the Pittsburgh approach, only the best individuals are selected as solutions. Therefore, in the iterative model, the GA provides a partial solution to the problem of learning. Some genetic-based iterative approaches, namely genetic-based machine learning (GBML) algorithms, have been used in the literature for generating fuzzy classification rules from data to find the nondominated rule set [28], [29].

A structural learning algorithm in a vague environment (SLAVE) [30], [31] is an iterative fuzzy GBML algorithm that uses the concepts of fuzzy logic and GA. This learning algorithm extracts a set of fuzzy rules from examples. This process is developed through an iterative approach in which a rule is selected each time. SLAVE uses a GA to select the rule that best represents the system. Ishibuchi and Yamamoto [32] have also utilized a GBML algorithm to select a subset of fuzzy rules incorporating the combinatorial effect of rules. However,

Manuscript received October 27, 2006; revised February 27, 2007, April 22, 2007, and May 10, 2007; accepted July 3, 2007.

The authors are with the Department of Computer Science and Engineering, Shiraz University, Shiraz 71935, Iran (e-mail: mansoori@shirazu.ac.ir; zjahromi@shirazu.ac.ir; katebi@shirazu.ac.ir).

Digital Object Identifier 10.1109/TFUZZ.2008.915790

in these approaches, the number of possible rules increases exponentially with the problem dimension. This also occurs for the search space to select the best rule set. As a result, the genetic-based rule selection algorithms require a long CPU time and large memory storage in the case of high-dimensional problems [33]. To overcome these difficulties, we have proposed a steady-state algorithm for extracting fuzzy classification rules from data (SGERD), a steady-state genetic algorithm to extract a compact set of readable fuzzy rules from numerical data in much lower computational efforts.

SGERD is generational and population-based, where its generations are finite and bounded to the problem dimension. Individual selection in this algorithm is nonrandom, and only the best ones can survive. Each parent produces a finite number of offspring through reproduction, whereas the crossover and mutation operators are very specific, and a few crossovers might be replaced by mutations. The fitness function used in SGERD is based on a rule evaluation criterion, very determinant in featuring the best rules among all candidates. Because of its importance, we have also proposed an enhancing function that modifies the initial rule evaluation measures in order to conduct the selection of better rules.

The rule selection mechanism in SGERD induces competition among rules by only considering the quality of approximation performed by each rule. However, to consider the cooperation among rules in order to increase the generalization power of the classifier, we have proposed a heuristic approach that selects only more cooperative rules among the final population for a rule base.

The rest of this paper is organized as follows. In Section II, the general design of fuzzy-rule-based classification systems from labeled data is explained. Section III considers the rule evaluation criteria. In Section IV, the tradeoff between general and specific rules is discussed. Using SGERD to generate fuzzy classification rules is explained in Section V. In Section VI, we present the experimental results. Section VII concludes the paper.

II. General Design of Fuzzy-Rule-Based Classification Systems

Fuzzy if–then rules for a pattern classification problem with n attributes can be written as

$$\text{Rule } R_j\text{: If } x_1 \text{ is } A_{j1} \text{ and } \ldots \text{ and } x_n \text{ is } A_{jn}, \text{ then class } C_j, \quad \text{for } j = 1, 2, \ldots, N \qquad (1)$$

where $X = [x_1, x_2, \ldots, x_n]$ is an n-dimensional pattern vector, A_{ji} $(i = 1, 2, \ldots, n)$ is an antecedent linguistic value, C_j is the consequent class of R_j, and N is the number of fuzzy rules. Generally, for an M-class problem with m labeled patterns $X_p = [x_{p1}, x_{p2}, \ldots, x_{pn}], p = 1, 2, \ldots, m$, the task of the classifier design is to generate a set of N fuzzy rules in the form of (1).

For this purpose, each attribute is first rescaled to unit interval [0, 1] by using a linear transformation that preserves the distribution of training patterns. Then, the pattern space is partitioned into fuzzy subspaces, and each subspace is identified by a fuzzy rule if there are some patterns in that subspace [6]. To do partitioning, usually K suitable membership functions are used to assign K linguistic values to each input attribute. Traditionally, triangular membership functions are used for this purpose, because they are simpler and more human-understandable. Fig. 1 shows these membership functions for four different values of K, where the linguistic labels L_3, L_4, and L_5, for example, can be interpreted as linguistic values *small*, *medium*, and *large*, respectively.

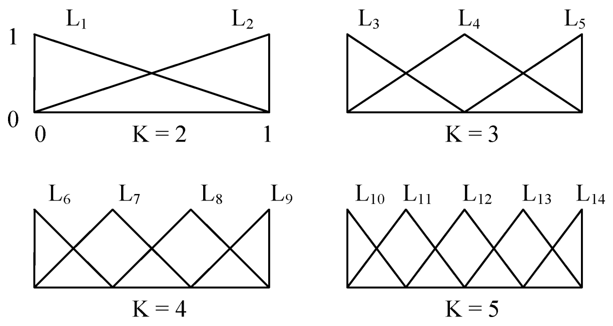


Fig. 1. Different partitioning of each input attribute.

Given an input partitioning of pattern space, one approach is to consider all possible combinations of antecedent linguistic values and generate a fuzzy rule for each combination. However, for high-dimensional problems, this approach can generate too many rules that are impractical to handle. For example, for a data set having n input attributes, K^n fuzzy rules might be generated. An approach for handling this problem is employing some rule evaluation criteria to select a small subset of rules among all candidates [32].

This approach simultaneously considers all membership functions in Fig. 1 for each attribute. That is, one of the 14 fuzzy sets can be used for each attribute when generating a candidate rule. In this case, for an n-dimensional problem, 14^n antecedent combinations should be considered. However, it is not practical to consider such a huge number of antecedent combinations when dealing with high-dimensional problems.

One solution for the aforementioned problem is presented in [17] by adding the fuzzy set *don't care* (with linguistic label L_0) to each attribute. The membership function of this fuzzy set is defined as $\mu_{L_0}(x) = 1$ for all values of x. The trick is not to consider all antecedent combinations (which is now 15^n); instead only short fuzzy rules having a limited number of antecedent conditions are generated as candidate rules. Though the number of candidate rules in this scheme would also be quite large for many problems, we have exploited this idea in Section V to propose a special GBML algorithm that selects a compact set of effective fuzzy rules among all candidates.

The consequent class C_j of fuzzy rule R_j in (1) is determined by training patterns in the corresponding fuzzy subspace. The compatibility grade of the training pattern X_p is defined with the antecedent part $A_j = A_{j1} \times A_{j2} \times \cdots \times A_{jn}$ of rule R_j using the product operator as

$$\mu_j(X_p) = \prod_{i=1}^{n} \mu_{ji}(x_{pi}) \qquad (2)$$

where $\mu_{ji}(\cdot)$ is the membership function of the antecedent fuzzy set A_{ji} and $A_{ji} \in L_0, L_1, \ldots, L_{14}$. In order to select the consequent class of a rule, we have used the heuristic method proposed by Ishibuchi and Nakashima [34], which is based on the confidence of association rules from the field of data mining. A fuzzy classification rule in (1) can be viewed as an association rule of the form $A_j \Rightarrow$ Class C_j, where A_j is a multidimensional fuzzy set representing the antecedent condition of the rule and C_j is its class label. In [34], a measure for evaluating the confidence of a fuzzy association rule is provided as

$$\text{Conf}(A_j \Rightarrow \text{Class } T) = \frac{\sum_{X_p \in \text{Class } T} \mu_j(X_p)}{\sum_{\text{p}=1}^{\text{m}} \mu_j(X_p)}. \quad (3)$$

The consequent class C_j of the fuzzy rule R_j is specified by identifying the class with the maximum confidence obtained by (3). That is, the consequent class C_j is chosen as

$$C_j = \arg\ \max\{\text{Conf}(A_j \Rightarrow \text{Class } T) | T = 1, 2, \ldots, M\}. \quad (4)$$

The most popular fuzzy reasoning method in fuzzy-rule-based classification systems is the reasoning based on a single winner rule [35]. This method is simple and intuitive for human users. Other fuzzy reasoning methods are studied in [35] and [36]. When using the single winner reasoning method, a new pattern $X_t = [x_{t1}, x_{t2}, \ldots, x_{tn}]$ is classified according to the consequent class of winner rule R_w. Indeed, the winner rule has the maximum compatibility grade with X_t among the fired rules. This can be stated as

$$\mu_w(X_t) = \max\{\mu_j(X_t), \quad j = 1, 2, \ldots, N\} \quad (5)$$

where $\mu_j(X_t)$ is the compatibility grade of rule R_j with X_t using (2). Note that the classification of a pattern not covered by any rule in the rule base is rejected. This also occurs for a pattern X_t when two rules with different consequent class have the same value of $\mu(X_t)$ in (5).

Although, in this paper, we have tried to generate a compact set of good fuzzy rules with an acceptable performance, nevertheless, it is possible to adjust the membership functions or to use weighted fuzzy rules to achieve higher classification accuracy. However, modifying the membership function of fuzzy sets will degrade the interpretability of fuzzy rules. Instead, assigning some appropriate weight for rules [37], [38] or finding suitable weighting functions [39] will increase the accuracy of the classifier while preserving the comprehensibility of the fuzzy rules. In this paper, however, we have used fuzzy rules with no weights.

III. Rule Evaluation Criteria

In the field of data mining, the support of association rules often has been used as rule evaluation criteria [40]. The fuzzy version of support s_F for fuzzy classification rule $A_j \Rightarrow$ Class C_j is defined in [41] as

$$s_F(A_j \Rightarrow \text{Class } C_j) = \frac{1}{m} \sum_{X_p \in \text{Class } C_j} \mu_j(X_p) \quad (6)$$

where *m* is the number of given training patterns. The crisp version of support s_C for this rule can be specified as

$$s_C(A_j \Rightarrow \text{Class } C_j) = \frac{\eta_j}{m} \quad (7)$$

where η_j is the number of patterns truly classified by rule R_j as class C_j (i.e., the number of true positives). Multiplying the fuzzy version of support in (6) with the confidence of R_j from (3), we will obtain the rule evaluation measure proposed in [32] as

$$f_{CS}(A_j \Rightarrow \text{Class } C_j) = \frac{\left(\sum_{X_p \in \text{Class } C_j} \mu_j(X_p)\right)^2}{\sum_{\text{p}=1}^{\text{m}} \mu_j(X_p)}. \quad (8)$$

This criterion is used in Section VI for comparison purposes only.

In the SLAVE algorithm, a heuristic rule evaluation criterion based on the support has been used for extracting fuzzy rules from numerical data. Its simple criterion for rule evaluation can be specified as

$$f_C(A_j \Rightarrow \text{Class } C_j) = \eta_j - \bar{\eta}_j \quad (9)$$

where $\bar{\eta}_j$ is the number of patterns incorrectly classified by rule R_j as class C_j (i.e., the number of false positives). This rule evaluation measure can be fuzzified as

$$f_F(A_j \Rightarrow \text{Class } C_j) = \sum_{X_p \in \text{Class } C_j} \mu_j(X_p) - \sum_{X_p \notin \text{Class } C_j} \mu_j(X_p). \quad (10)$$

Since the number of training patterns *m* in (6) and (7) is equal for all fuzzy rules, it can be unconsidered. So, (10) can be obtained from the support of positive and negative patterns. This criterion is used as a basis in this paper for evaluating rules. In continuation, a proposed enhancing function is presented that modifies the measure in (10) to distinguish more effectively the better candidate rules.

Two main subspaces can be identified by each fuzzy rule: the covering subspace and the decision subspace. While any pattern in the covering subspace will cause the rule to be fired, only those in the decision subspace would be classified by it. The covering subspace of a rule has some overlaps with its adjacent rules, such that any pattern in this subspace will also fire the adjacent rules. However, in a rule generation phase, it is impossible to exactly identify the decision subspace of a rule since it depends on the decision area of adjacent rules. To do this approximately, we have introduced a threshold τ_j for rule R_j, and the patterns having compatibility grade above it are supposed to be in the decision subspace of R_j. This threshold will depend on the length of R_j (i.e., the number of antecedent variables). Since all membership functions in Fig. 1 are 0.5-complete and the compatibility grades in (2) are computed using the product operator, we have defined $\tau_j = 0.5^{k_j}$, where k_j is the length of rule R_j.

Because of the importance of positive training patterns in the decision subspace of a rule, we have proposed a piecewise linear function to enhance the compatibility grade of these

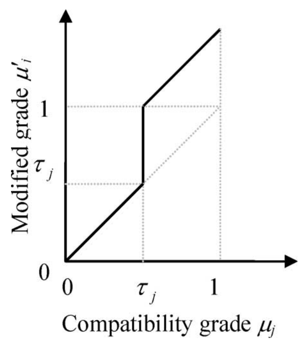


Fig. 2. Input–output mapping of enhancing function.

patterns. In [39], we have presented a weighting function in this manner that improves the performance of fuzzy classification systems. Using this enhancing function, the compatibility grade of each positive pattern will be shifted above 1. Fig. 2 shows the input–output mapping of this function.

Applying the enhancing function to rule R_j, the compatibility grade of positive patterns in its covering subspace will be modified as

$$\mu'_j(X_p) = \begin{cases} \mu_j(X_p), & \text{if } \mu_j(X_p) < \tau_j \\ \mu_j(X_p) + 1 - \tau_j, & \text{if } \mu_j(X_p) \geq \tau_j \end{cases} \tag{11}$$

and therefore, the rule evaluation measure in (10) can be modified by (11) as

$$\begin{aligned} f'_F(A_j \Rightarrow \text{Class } C_j) &= \sum_{X_p \in \text{Class } C_j} \mu'_j(X_p) - \sum_{X_p \notin \text{Class } C_j} \mu'_j(X_p) \\ &= \sum_{\substack{X_p \in \text{Class } C_j \\ \mu_j(X_p) \geq \tau_j}} \{\mu_j(X_p) + 1 - \tau_j\} + \sum_{\substack{X_p \in \text{Class } C_j \\ \mu_j(X_p) < \tau_j}} \mu_j(X_p) \\ &\quad - \sum_{X_p \notin \text{Class } C_j} \mu_j(X_p) \\ &= \sum_{\substack{X_p \in \text{Class } C_j \\ \mu_j(X_p) \geq \tau_j}} \mu_j(X_p) + \gamma_j(1 - \tau_j) + \sum_{\substack{X_p \in \text{Class } C_j \\ \mu_j(X_p) < \tau_j}} \mu_j(X_p) \\ &\quad - \sum_{X_p \notin \text{Class } C_j} \mu_j(X_p) \\ &= \gamma_j(1 - \tau_j) + \sum_{X_p \in \text{Class } C_j} \mu_j(X_p) - \sum_{X_p \notin \text{Class } C_j} \mu_j(X_p) \end{aligned} \tag{12}$$

or equivalently,

$$f'_F(A_j \Rightarrow \text{Class } C_j) = f_F(A_j \Rightarrow \text{Class } C_j) + \gamma_j(1 - \tau_j) \tag{13}$$

where γ_j is the number of positive patterns in the decision subspace of R_j (i.e., patterns having compatibility grade above τ_j).

It is notable that the positive patterns can be identified only after determining the consequent class of R_j by using (4). Therefore, modifying the compatibility grade of positive patterns in (11) should not affect the confidence of R_j in (3), and therefore, its consequent class in (4).

TABLE I
Attribute Covering Degree of Linguistic Labels

Linguistic label, A	Attribute covering degree, $\xi(A)$
L_1, L_2, L_4	1 (≈0.99)
L_7, L_8	2/3 (0.67)
$L_3, L_5, L_{11}, L_{12}, L_{13}$	1/2 (0.50)
L_6, L_9	1/3 (0.33)
L_{10}, L_{14}	1/4 (0.25)

IV. General Versus Specific Rules

The most apparent drawback of fuzzy-rule-based classification systems obtained from labeled data is overfitting. This often occurs when the generated fuzzy rules are very specific (i.e., very data-dependent). These rules have high classification accuracy on training data, but their performance descends considerably when applying to test data, revealing a lack of generality. This is because the specific rules have more antecedent variables, and therefore, more restricted fuzzy subspaces. So, more rules must be generated to cover the whole space of the problem. Consequently, the overfitting occurs when many specific fuzzy rules with high classification accuracy on training data are generated to cover the problem space.

To prevent overfitting, a few general rules having large fuzzy subspaces are needed. But for some data sets with high class-overlapping patterns, the generated fuzzy rules must be specific enough to achieve an acceptable performance. Therefore, there is a tradeoff between the specificity and the generality of rules, and balancing them is an important challenge.

To overcome this difficulty, we have included some heuristics in computing the fitness function of SGERD. In this scheme, the evaluation measure of some longer rules (e.g., rules having antecedent variables more than $n/2$) is modified by two valuable factors. The first factor is rule-dependent and takes higher values for more general rules to appreciate the generality. This factor indeed measures the rule subspace size ρ_j of rule R_j. The second factor, on the other hand, is data-dependent and would show the interclass dependency of data set. This factor (namely, class overlapping rate χ_j) is used to emphasize the specificity of rules.

To evaluate the rule subspace size ρ_j, we have assigned an attribute covering degree ξ to each of the membership functions in Fig. 1. This degree is used to illustrate how much each membership function is covering the domain of the input attribute [42]. Table I shows this value for each of the linguistic labels in Fig. 1.

Since the values of all covering degrees are below 1, accumulating some of them in a rule will result in much lower values. So, the rule subspace size ρ_j for rule R_j can be specified as

$$\rho_j = \prod_{i=1}^{k_j} \xi(A_{ji}) \tag{14}$$

where k_j is the length of R_j and A_{ji} is an antecedent linguistic label of this rule. Obviously, this factor takes higher values for general rules than the specific ones with more antecedent variables. So, it should directly affect the evaluation measure of R_j in (17).

The class-overlapping rate, as mentioned before, depends on the distribution of distinct classes in training data. To compute this factor approximately, we have utilized the *K*-NN approach to detect lonely patterns (i.e., patterns surrounded by irrelevant ones) in training data. Firstly, the Euclidian distances between all pairs of patterns are computed (preferably offline). Then for each pattern, the class of its five nearest neighbors (5-NN) is investigated, and it is marked as a lonely (noisy or outlier) pattern if its four or five neighbors are from a different class. At last, the rate of unmarked patterns is considered a class-overlapping factor. Saying formally, this factor for class *C* can be formulated as

$$1 - \frac{o(C)}{\upsilon(C)} \tag{15}$$

where $\upsilon(C)$ is the number of examples from class *C* in training data while *o*(*C*) of them are outliers. From (15), it is clear that, when training patterns of different classes are possibly well separated (i.e., *o*(*C*) is small), the class overlapping rate will take a higher value (near 1). On the contrary, a lower value will be assigned to it when mentioned patterns are highly mixed together (i.e., *o*(*C*) is large). But we have introduced this factor to emphasize the specificity of rules, so its inverse is used in (17) to affect the rule evaluation measures.

Incorporating k_j in (15), we can customize the class overlapping rate χ_j for rule R_j as

$$\chi_j = \left(1 - \frac{o(C_j)}{\upsilon(C_j)}\right)^{k_j} \tag{16}$$

where C_j is the consequent class of R_j.

Including these factors in rule evaluation measures, the fitness function of SGERD for rule R_j can be specified as

$$\text{fitness } (R_j) = \begin{cases} f_j, & \text{if } k_j \le \dfrac{n}{2} \\ \dfrac{\rho_j}{\chi_j} f_j, & \text{if } k_j > \dfrac{n}{2} \end{cases} \tag{17}$$

where f_j is one of the rule evaluation measures in (8), (10), or (13).

V. Using SGERD to Generate Fuzzy Rules

The proposed SGERD is similar to Genitor [43], the first steady-state genetic algorithm and cross-generational elitist selection, heterogenous recombination and cataclysmic mutation (CHC) [44] that monotonically collects the best individuals found so far. Unlike Genitor, in which two parents produce one offspring at a time, in the reproduction of SGERD, a finite number of offspring are generated. On the other hand, in SGERD, similar to Genitor and CHC, offspring do not replace parents definitely, but rather the least fit member of the population. To borrow the terminology used by the evolution strategy community, SGERD, like Genitor and CHC, is a ($\mu + \lambda$) strategy, while the canonical GA is a (μ, λ) strategy [43]. Indeed, after recombination, some of the best unique individuals are drawn from the population of parents and offspring to create the next generation.

A. SGERD Characteristics

SGERD is generational and population-based. Some of its characteristics are problem-dependent and must be specified accordingly. Generally, for an *n*-dimensional problem with *M* classes and *m* labeled patterns, the task of SGERD is to generate possibly a prespecified number of *Q* rules per class (i.e., the best ones) in the final population, though only a few of these $M \times Q$ rules would be used in the rule base.

Before explaining the algorithm, we have considered some overall characteristics of SGERD as follows:

1) *Chromosomes:* Each individual (i.e., chromosome) in the population is a fuzzy classification rule having *n* antecedent variables and a consequent class in $\{1, 2, \ldots, M\}$. Each antecedent variable (i.e., gene) takes one fuzzy set from $\{L_0, L_1, \ldots, L_{14}\}$ in Fig. 1 as its allele. Since the fuzzy set L_0 is *don't care* and its membership value is always 1, the antecedent variables using L_0 have no effect on the fuzzy subspace of the rule, so we have called such variables inactive. Regardless of elitism, in each generation of SGERD, only one of these inactive variables will be activated (i.e., L_0 will be replaced by an appropriate label from $\{L_1, L_2, \ldots, L_{14}\}$). Accounting for both active and inactive genes, the length of each chromosome is fixed to *n*. However, for rule R_j, the actual length is k_j, the number of its active variables.
2) *Initial population:* The initial population is selected from all fuzzy rules having only one active antecedent variable. Since each variable can take one of the 14 fuzzy sets, the number of these candidate rules will be 14 $\times$ *n* at most. Obtaining their consequent class, these rules can be grouped into *M* classes. Choosing possibly the best *Q* of them per class, the initial main population will contain $M \times Q$ rules at most. The rest of these candidates (i.e., at most $M \times Q$ of their best ones) are set aside as an auxiliary population that will be utilized in the mutation process. Obviously, all members of the auxiliary population have only one active variable in their antecedents.
3) *Fitness function:* The rule evaluation metrics in (8), (10), and (13), as modified in (17), are the fitness function for measuring the candidate rules.
4) *Stop criteria:* As mentioned before, in each generation of SGERD, only one of the inactive variables in the antecedent part of each rule will be activated (i.e., each parent rule R_j generates, as offspring, longer rules). Since each rule might have up to *n* active variables, the number of generations will be *n* at most. However, each parent is allowed to produce only self-fitter offspring. So, generating a new population will be stopped if no new offspring fitter than their parents can be produced. Thus, the number of generations will be less than *n* in practice.
5) *Reproduction:* To generate offspring through reproduction, each parent rule R_j (with consequent class C_j) exploits the participation of another parent from C_j. This

second parent R_p is only utilized to determine which inactive antecedent variable in R_j should be activated in this generation. The selection of R_p is done through SGERD-specific crossover or mutation operations. In this way, R_p is selected randomly among rules of class C_j in the current population (this is supposed as crossover operation since both parents R_j and R_p are chosen from the main population). However, it is possible that R_j be selected again for R_p in this process. In this case, R_p would be selected randomly from the auxiliary population instead (doing mutation on R_p instead of crossover). After identifying R_p, one of its active antecedent variables is selected randomly (e.g., x_i). If x_i is inactive in R_j, then the reproduction on R_j can take place by activating x_i (i.e., replacing L_0 with one fuzzy set from $L_1, L_2, \ldots, L_{14}$) and generating all possible offspring (at most 14), provided the offspring be fitter than R_j. These offspring will use x_i as active beside other active antecedents in R_j. However, if x_i has been active in R_j before reproduction, then no reproduction can occurr on it. In this case, R_j can survive in the next generation through elitism. The consequent class of some offspring might differ from their parents' because each offspring is a more specific fuzzy rule (with more active antecedent variables) than its parent, and so, its fuzzy subspace is smaller. This might also prevent an offspring from being produced if no patterns exist in the new subspace.

B. SGERD Algorithm

In this section, we have discussed in detail the algorithm of SGERD that generates as a final population, a prespecified number of Q rules per class ($R = M \times Q$ rules in total at most).

Algorithm: SGERD.

Inputs: m labeled patterns of an n-dimensional M-class problem and Q.

Outputs: Possibly $R = M \times Q$ fuzzy classification rules.

1) $i = 1$ (i: generation number).
2) Generate all fuzzy rules having only one active antecedent variable (at most $C = 14 \times n$ candidate rules would be generated).
3) Determine the consequent class of each candidate rule using (4).
4) Divide the candidate rules into M groups according to their consequent class.
5) Rank, in descending order of their fitness values, the candidate rules in each group.
6) Choose the best Q rules from each class (i.e., possibly $R = M \times Q$ rules in total) as the population in the ith generation. In the first generation only, choose the second best R rules as the auxiliary population and put away for mutation.
7) Increment i, **if** $i > n$, **goto** step 11.
8) Use all individuals in the previous generation (i.e., R rules) as parents and do reproduction (i.e., crossover, mutation, or elitism) on them. That is, for each parent rule, generate as offspring all fuzzy rules having one more active antecedent variable than its parent, provided each new offspring be fitter than its parent. In this case, the number of offspring will totally be $R \times 14$ at most.
9) **If** no offspring fitter than the parents is produced in step 8, **goto** step 11.
10) Consider both parents and offspring in step 8 as candidate rules (at most $C = R + R \times 14$ rules in total) and **goto** step 3.
11) Use $R = M \times Q$ rules (obtained in step 6 for the ith generation) as the final population and **stop**. The actual length of these rules is i or less.

TABLE II
STATISTICS OF DATA SETS USED FOR SGERD EVALUATION

Data set	Number of attributes (n)	Number of classes (M)	Number of samples (m)
Iris	4	3	150
Wine	13	3	178
Glass	9	6	214
Cancer (Breast)	9	2	684
Pima (Diabetes)	8	2	768
Image	19	7	210
Sonar	60	2	208
Segment	19	7	2310
Yeast	8	10	1484
Vowel	10	11	990
Page (Blocks)	10	5	5473

As mentioned before, the rule selection scheme in SGERD only considers the evaluation measure of each rule to select the best ones through competition. However, incorporating the cooperation among rules will increase the generalization power of the classifier. To attain this intention, we have proposed a heuristic approach that extracts more cooperative rules from the final population.

This approach is capable of selecting a compact rule base while presenting the best possible cooperation among rules, and thereby, the best possible behavior. In this approach, using their fitness, first the best rule for each class (i.e., M rules) is selected. Then, among the remaining, the second best rules are investigated by classifying the training patterns, and only those are selected that improve the classification accuracy. This algorithm continues for classes that add more rules to the rule base. Hence, the rule base of the designed classification system would have N ($\leq R$) fuzzy rules in practice. This heuristic approach will work better with imbalanced data sets where some classes would need more rules than others.

VI. Experimental Results

In this section, the performance of SGERD in fuzzy rule-based classification systems is examined. We have used 11 data sets with numerical attributes available from the University of California, Irvine machine learning repository [45]. Table II shows the specifications of these data sets.

To illustrate graphically the operation of SGERD in extracting fuzzy rules, some data sets having two attributes were needed. For this purpose, the Fisher interclass separability criterion [15] is used to rank the features of Iris and Wine data sets. Then, two highest ranked features, $\{x_4, x_3\}$ for Iris and $\{x_{13}, x_{12}\}$ for Wine, are selected and are denoted by Iris2f and Wine2f, respectively.

Applying SGERD on Iris2f, in the first generation, we obtain the following best rules, where R_1, R_2, and R_3 are the first best rules per class and R_4, R_5, and R_6 are the second best ones. Evidently, there are much more rules in the first generation not shown here.

R_1: If x_4 is L_3, then class 1.
R_2: If x_4 is L_{12}, then class 2.
R_3: If x_4 is L_5, then class 3.
R_4: If x_3 is L_3, then class 1.
R_5: If x_3 is L_{12}, then class 2.
R_6: If x_3 is L_{13}, then class 3.

The antecedent variable x_4 in rules R_1, R_2, and R_3 is active while x_3 is inactive. The inverse situation has occurred for rules R_4, R_5, and R_6. In the reproduction phase, the inactive variables can be activated through crossover or mutation, provided the newly produced rules be fitter than their parents.

Considering rule R_1 as the first parent for example, the second parent can be R_4 or any other rule from class 1 in the main population. If the randomly selected second parent be R_1 again, the crossover operation cannot occurr in this case. Instead, the mutation operation must be used to choose the second parent through random selection of a rule from class 1 in the auxiliary population. Nevertheless, in this example, rule R_4 is chosen as the second parent. The only active variable in R_4 is x_3, whereas this variable in R_1 is inactive. So, the reproduction (i.e., crossover) can take place on R_1 by activating x_3. In this case, R_1 can produce up to 14 rules as offspring in the form of "If x_4 is L_3 and x_3 is L_i, then class 1, for $i = 1, \ldots, 14$." Accounting for the fitness value of these offspring, only four of them (i.e., L_1, L_3, L_6, and L_{10}) are produced indeed. Thus, the rule R_1 produces four offspring for the next generation, while the fittest one is "If x_4 is L_3 and x_3 is L_1, then class 1." The same procedure occurs for rules R_2, R_3, and R_4. The fittest offspring of these rules are next shown as the best rules in the second generation. However, rules R_5 and R_6 in the second generation are not the offspring of R_5 and R_6 in the first generation at all. Rule R_5, for example, is the fittest offspring of either "If x_4 is L_4, then class 2" or "If x_3 is L_4, then class 2." This is because the offspring of these rules have been fitter than the offspring of R_5 in the first generation (i.e., "If x_3 is L_{12}, then class 2").

R_1: If x_4 is L_3 and x_3 is L_1, then class 1.
R_2: If x_4 is L_{12} and x_3 is L_4, then class 2.
R_3: If x_4 is L_5 and x_3 is L_2, then class 3.
R_4: If x_4 is L_1 and x_3 is L_3, then class 1.
R_5: If x_4 is L_4 and x_3 is L_4, then class 2.
R_6: If x_4 is L_2 and x_3 is L_5, then class 3.

Fig. 3 compares the decision area and classification accuracy of first three rules for Iris2f in two generations.

By repeating this procedure for Wine2f, the six best rules in first generation are as follows:

R_1: If x_{13} is L_8, then class 1.
R_2: If x_{13} is L_3, then class 2.
R_3: If x_{12} is L_3, then class 3.
R_4: If x_{13} is L_2, then class 1.

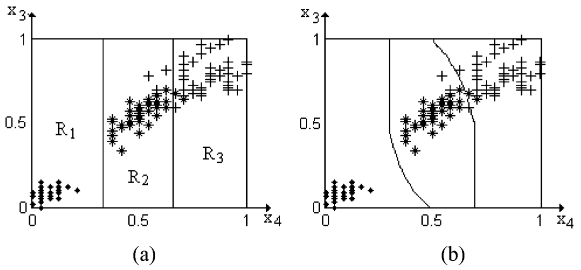


Fig. 3. Performance of applying SGERD on Iris2f. (a) 1st generation; 7 missed. (b) 2nd generation; 6 missed.

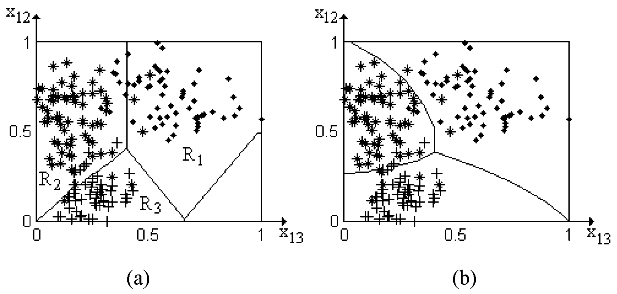


Fig. 4. Performance of applying SGERD on Wine2f. (a) 1st generation; 23 missed. (b) 2nd generation; 18 missed.

R_5: If x_{13} is L_6, then class 2.
R_6: If x_{12} is L_6, then class 3.

In the second generation, rules R_2, R_3, and R_6 are replaced by their fitter offspring, while R_1 and R_4 are offspring of R_4 and R_1 in the first generation, respectively. Moreover, rule R_5 is replaced by offspring of a new parent.

R_1: If x_{13} is L_2 and x_{12} is L_2, then class 1.
R_2: If x_{13} is L_3 and x_{12} is L_4, then class 2.
R_3: If x_{13} is L_1 and x_{12} is L_3, then class 3.
R_4: If x_{13} is L_8 and x_{12} is L_2, then class 1.
R_5: If x_{13} is L_1 and x_{12} is L_4, then class 2.
R_6: If x_{13} is L_1 and x_{12} is L_6, then class 3.

The effect of the new rule R_1 on the classification accuracy is illustrated in Fig. 4.

To illustrate the scalability of SGERD, especially the effect of parameter Q on its performance, we have run SGERD for three different values of Q and have computed some criteria for each data set. For this purpose, all patterns of each data set are used as training examples for constructing the rule base and also for testing the classifier. The fitness function of SGERD in these experiments has been the measure in (13) as modified in (17). The computed criteria are: 1) number of generations SGERD has been run to produce the candidate rules; 2) total number of candidate rules that have been generated; 3) CPU time in seconds elapsed to run SGERD and also to extract the rule base from the final population; 4) number of rules in the rule base; 5) average length of rules in the rule base; and 6) the classification error rate obtained by applying the rule base to training data. Table III summarizes the results.

As Table III shows, the number of generations produced by SGERD is much less than *n* for high-dimensional problems

TABLE III
PERFORMANCE OF SGERD FOR DIFFERENT VALUES OF Q

Data set	Q	No. of generations	No. of candidates	CPU time	No. of rules	Length of rules	Error rate
	10	4	185	1	6	1.83	1.33
Iris	20	4	502	1	5	2.00	2.67
	30	4	658	3	6	2.50	2.00
	10	5	512	2	7	3.29	1.12
Wine	20	7	997	4	8	3.00	0.56
	30	8	1701	6	8	3.38	1.69
	10	7	841	4	13	4.08	33.18
Glass	20	8	1893	7	12	3.83	31.78
	30	8	3205	14	13	3.92	26.17
	10	5	340	2	4	2.25	2.92
Cancer	20	6	626	3	5	2.60	2.63
	30	6	897	4	5	2.20	2.34
	10	6	329	2	6	2.83	25.78
Pima	20	8	646	6	9	6.67	24.48
	30	8	748	5	8	3.00	23.96
	10	7	1138	5	7	4.00	13.33
Image	20	9	2283	10	10	4.40	12.86
	30	10	4393	20	7	5.00	12.38
	10	8	1286	8	5	4.60	20.19
Sonar	20	8	1895	13	7	5.14	16.83
	30	11	2660	19	7	5.57	15.87
	10	6	1084	16	10	4.20	17.58
Segment	20	9	2879	45	13	5.23	13.16
	30	9	4835	81	16	5.50	14.68
	10	6	333	7	17	2.41	45.01
Yeast	20	7	1580	25	19	3.16	46.56
	30	7	4491	68	24	5.38	39.56
	10	6	2411	32	30	3.70	41.01
Vowel	20	9	5126	115	40	4.10	32.83
	30	6	7957	230	45	3.93	29.39
	10	6	622	22	14	3.43	7.66
Page	20	6	1293	32	10	3.50	8.33
	30	6	1922	46	9	3.78	8.79

(e.g., Image, Segment, and Sonar data sets). Indeed, SGERD has produced 11 generations at most. As mentioned before, in new generations, more antecedent variables of rules are activated and so the fuzzy subspace of offspring becomes smaller than that of their parents. Consequently, producing fitter offspring becomes less attainable. Considering the number of generations, number of candidate rules, and the CPU time altogether, they justify that SGERD is fast enough to be applied to both high-dimensional and large problems. From the size of the rule base and the average length of rules, we conclude that our approach gathers a small set of simple and readable rules in the rule base to produce an interpretable system.

From the classification error rate and the CPU time in Table III, it is clear that the value of Q is determinant in the majority of data sets. In practice, we have tried to assign a suitable value for Q heuristically. For this purpose, we have used one half of the initial candidate rules ($14 \times n$ rules at most) as the main population and the other half as the auxiliary population. For either high-dimensional or large problems, however, this value has been restricted to 20 in this paper. Saying formally,

$$Q = \min\left(\frac{14 \times n}{2 \times M}, 20\right). \tag{18}$$

To examine the generalization power of fuzzy-rule-based classification systems designed by SGERD, we have used the ten-fold cross-validation (10-CV) testing method. In this method, each data set is randomly divided into ten subsets of the same size without considering the class membership of each pattern. Nine subsets are used as training patterns for generating the rule base. The tenth subset is used as test patterns for evaluating the system. The same training and testing procedure is also performed nine times after exchanging the role of each subset so that all subsets are used as test patterns once. Since the error rate on test patterns depends on the initial division of the data set, the 10 CV is iterated five times by using different divisions of the data set, and the average classification rate is reported as the performance of the classifier. Table IV illustrates the simulation results where all numbers are average values, and the error rate also includes the rejection rate. As mentioned before, the measures in (8), (10), and (13) are modified in (17), and used as the fitness function.

In this table, two results are reported for each data set. The first result is obtained by using only one rule per class (i.e., the best M rules among $R = M \times Q$ in final population). To obtain the second result, the next best rules affecting the classification accuracy of training data are also included (i.e., the best N rules among R rules as discussed in Section V).

Comparing the results of the first and the second evaluation criteria in Table IV prove that SGERD is very sensitive to the rule evaluation method. It is obvious that the rule evaluation measure in (10), which is based on the difference of positive and negative supports, works better than the product of the confidence and the support in all data sets except "Cancer" and "Page." Also, as seen in this table, the evaluation measure in (13), which is the modified version of (10), has better performance on all data sets except for "Iris," "Pima," and "Sonar." Although it works a little worse for these three data sets, its performance improvement for others is noticeable, especially for large data sets. Indeed, this evaluation measure is more suitable for multiclass data sets having unequal distribution of classes. However, using criterion in (10) generates simpler rules than (13), as the average length of rules point out. Since the measure in (13) pays more attention to positive patterns, it relies on more specific rules with more antecedent variables, so the length of rules increases. Considering this point, we conclude that the rule base obtained from (10) is more interpretable.

Since the C4.5 algorithm [46] is one of the most well known and frequently used methods for designing nonfuzzy-rule-based systems, we have implemented it to compare with SGERD in detail. Table V summarizes these comparisons made on six data sets. In this table, all results are average values obtained from five iterations of the 10-CV testing method except the CPU time, which is the running time for only once.

The results in this table clarify that SGERD outperforms C4.5 in total. The generalization ability of SGERD is much better than that of C4.5, as the classification error rates show. Regarding much fewer and simpler rules proves the interpretability of the rule base constructed by SGERD. Nevertheless, these high performances are achieved with much lower computational efforts. However, C4.5 is not a GBML approach (it is just a decision tree learning method). It does not search the best rules among a population of rules; rather, it only tries to make the best decision tree for training patterns in a greedy manner. Thus, it is not fair

TABLE IV
COMPARING THE PERFORMANCE OF SGERD FOR THREE RULE EVALUATION MEASURES

Data set	Evaluation measure in (8)			Evaluation measure in (10)			Evaluation measure in (13)		
	Error rate	No. of rules	Length of rules	Error rate	No. of rules	Length of rules	Error rate	No. of rules	Length of rules
Iris	4.93	3.00	1.00	4.67	3.00	1.00	6.27	3.00	2.07
	3.07	3.96	1.01	3.07	4.00	1.01	3.60	4.30	1.95
Wine	10.95	3.00	1.00	11.80	3.00	1.00	10.34	3.00	3.61
	5.83	6.38	1.31	4.84	7.12	1.39	3.81	6.14	3.56
Glass	60.13	5.88	1.79	52.92	5.96	1.96	47.37	5.92	4.46
	54.31	7.50	1.71	37.10	11.52	1.85	36.62	10.22	4.24
Cancer	6.36	2.00	1.63	6.64	2.00	1.64	5.50	2.00	2.95
	3.48	4.64	1.52	3.71	5.38	1.40	2.98	3.96	2.31
Pima	32.42	2.00	1.00	26.70	2.00	1.00	31.17	2.00	7.48
	30.73	3.96	1.03	25.36	6.12	1.42	26.92	7.76	7.18
Image	31.90	7.00	1.78	21.43	7.00	2.01	16.95	7.00	4.39
	22.00	11.30	1.88	16.48	11.44	2.18	13.90	9.28	4.56
Sonar	47.88	2.00	1.00	27.78	2.00	1.00	33.38	2.00	5.49
	36.93	8.90	1.04	22.80	4.92	1.14	24.80	5.96	5.17
Segment	32.61	7.00	1.93	22.36	7.00	1.99	17.01	7.00	4.20
	25.32	16.30	2.13	16.30	15.44	2.25	15.06	12.52	4.32
Yeast	68.44	8.88	1.83	64.52	9.86	2.95	52.36	9.50	5.41
	68.34	9.38	1.79	50.16	22.36	2.85	43.47	21.50	5.50
Vowel	65.88	11.00	1.44	67.19	11.00	2.75	55.60	11.00	3.98
	56.57	27.22	1.80	50.32	30.00	3.04	41.47	33.78	3.88
Page	9.31	5.00	1.32	9.75	5.00	1.45	9.48	5.00	3.79
	8.90	5.96	1.27	9.58	7.18	1.51	8.62	9.26	3.67

TABLE V
PERFORMANCE OF SGERD FOR DIFFERENT VALUES OF Q

Data set	Error rate		No. of rules		Len. of rules		CPU time	
	C4.5	SGERD	C4.5	SGERD	C4.5	SGERD	C4.5	SGERD
Iris	5.07	3.60	7.34	4.30	3.44	1.94	6	5
Wine	5.40	3.81	7.52	6.14	3.01	3.56	26	33
Glass	32.84	36.62	45.50	10.22	6.75	4.24	54	26
Cancer	4.60	2.98	24.40	3.96	5.77	2.31	127	26
Pima	27.79	26.92	54.74	7.76	7.15	7.17	178	46
Sonar	27.90	24.80	21.96	5.96	5.29	5.17	223	106

TABLE VI
COMPARING THE ERROR RATE OF SGERD WITH SOME CLASSIFICATION APPROACHES

Classification approach		Data set					
Reference	Method	Glass	Wine	Cancer	Sonar	Iris	Pima
Quinlan [46]	C4.5	32.50	-	5.26	25.60	4.80	25.40
Elomaa [47]	C4.5	**27.70**	5.60	5.30	25.10	6.60	25.70
Sanchez [48]	GP+SA	42.10	-	4.65	-	-	-
SLAVE [31]	GBML	-	6.20	-	-	4.28	-
Ishibuchi [32]	GBML	40.38	6.90	3.25	24.06	-	-
Abonyi [49]	DT+Fuzzy	33.97	8.78	3.18	-	3.89	26.95
Guan [50]	Nonfuzzy+GA	-	8.33	4.70	-	4.40	-
Ishibuchi [29]	Hybrid GBML	34.63	4.94	3.32	23.70	5.33	**24.17**
SGERD	GBML	36.62	**3.81**	**2.98**	**22.80**	**3.07**	25.36

to compare our GA-based approach with C4.5, though SGERD outperforms it.

For comparing the performance of SGERD with other approaches (especially GBML ones) in the literature, some reported results are included in Table VI. The average error rates in this table are obtained by using the 10-CV testing method where the best result in each column is highlighted by boldface.

In this table, we have shown some reported results from [46] where a modified version of the C4.5 algorithm (Rel. 8) is proposed to appropriately handle continuous attributes. Elomaa and Rousu [47] have proposed an optimal discretization method of continuous attributes into multiple intervals and have examined the performance of six variants of the C4.5 algorithm. In Table VI, we have included the best results of [47] obtained by using the optimal multisplitting strategy. We have also reported some results from Sanchez *et al.* [48] that combine genetic programming (GP) operators with simulated annealing (SA) to search the best rules. Gonzalez and Prez [31] proposed some modifications of their original SLAVE learning algorithm, including new genetic operators for reducing the learning time and improving the interpretability of rules. The best results of [31] are shown in Table VI. Although the data set division in SLAVE has been 70% for training and 30% for test instead of 90–10% used in SGERD, the performance of SGERD is good enough to compensate this deficiency. Ishibuchi and Yamamoto [32] have utilized a GA to select a subset of fuzzy rules generated by some heuristic rule selection criteria to take into account the combinatorial effect of rules. Its best results are also cited. We have also cited the best results of a decision-tree-based fuzzy classifier in [49] that are obtained from the 5-CV testing method. In that paper, an initial fuzzy classifier constructed from a decision tree is reduced and optimized in three phases using GA. However, the obtained membership functions are not linguistic and so not readable. Table VI also includes the best results of a GA-based nonfuzzy classifier that uses a class decomposition approach [50]. At last, we have reported the best results of a

TABLE VII
COMPARING THE NUMBER OF RULES EXAMINED BY GA-BASED APPROACHES

Classification approach	Population size	Number of generations	Number of examined rules
Ishibuchi [32]	600	5000	3 000 000
Ishibuchi [29]	10×200	1000	2 000 000
SLAVE [31]	20	1000	20 000
Guan [50]	100	200	20 000
SGERD (Glass)	6×20	8	1893
SGERD (Sonar)	2×20	8	1895

hybrid algorithm that combines two fuzzy GBML approaches (i.e., Michigan and Pittsburgh) for designing fuzzy-rule-based classification systems [29]. This algorithm, however, produces fuzzy rules having weights. The use of weighted rules in fuzzy classification systems would result in a better performance. Evidently, assigning appropriate weights to the rules obtained by SGERD will achieve higher classification accuracy.

As Table VI shows, the generalization ability of SGERD is considerable and the classification error rate in the majority of data sets is the least except for "Glass" and "Pima," where they are comparable to other GBML approaches. At last, to compare the computational efforts of GA-based approaches in Table VI, we have cited the approximate number of rules examined by each method in Table VII.

The number of examined rules for approaches other than SGERD is obtained by multiplying the population size and the number of generations. For SGERD, however, the number of total offspring produced in whole generations is counted.

VII. CONCLUSION

In this paper, we examined the performance of SGERD in extracting fuzzy IF–THEN rules from numerical data for classification purposes. We showed that, unlike standard GA, the selection mechanism in SGERD is nonrandom, and only the fittest individuals will survive. In its reproduction phase, we saw that each parent employs the cooperation of another individual through crossover or mutation to generate offspring. It was illustrated that almost all characteristics of SGERD were problem-dependent. These included population size, number of generations and number of offspring for each parent, crossover and mutation rates, and a class-overlapping factor used for attaining generality.

Using SGERD, the population size in each generation was limited to the number of classes multiplied by the number of rules per class, which, in the worst case, was 130 for the "Image" data set. The other parameter that highly affected the elapsed CPU time was the number of generations. This parameter was bounded to problem dimensions, though in practice, much fewer rules were produced especially in high-dimensional data sets. Therefore, SGERD was fast enough to be applied to high-dimensional data sets having numerical attributes.

Altogether, the advantages of SGERD can be outlined as follows:

1) Its algorithm is simple and intuitive.
2) It generates a few general rules that are short, accurate, and interpretable.
3) Its population size and number of generations are small, so it saves memory space.
4) It is fast and can be applied to high-dimensional problems.

However, SGERD suffers from a drawback. Since it selects the rules of each class independent of other classes, it possibly cannot utilize the cooperation among rules when producing offspring. Moreover, selecting only the best rules might intensify this drawback, though in the phase of rule base construction from the final population, this cooperation is taken into account. The future version of SGERD must obviate this drawback.

We also showed that the rule evaluation criteria highly impress the performance of SGERD. Therefore, developing the proposed criteria in this paper or offering new evaluation measures might assist SGERD in extracting more efficient fuzzy rules from numerical data.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their valuable comments.

REFERENCES

[1] R. Mikut, J. Jäkel, and L. Gröll, "Interpretability issues in data-based learning of fuzzy systems," *Fuzzy Sets Syst.*, vol. 150, pp. 179–197, 2005.
[2] R. Alcalá, J. Alcalá-Fdez, J. Casillas, O. Cordón, and F. Herrera, "Hybrid learning models to get the interpretability-accuracy trade-off in fuzzy modeling," *Soft Comput.*, vol. 10, pp. 717–734, 2006.
[3] J. Casillas, O. Cordon, F. Herrera, and L. Magdalena, *Accuracy Improvements in Linguistic Fuzzy Modeling*. New York: Springer-Verlag, 2003.
[4] H. Ishibuchi and Y. Nojima, "Analysis of interpretability-accuracy trade-off of fuzzy systems by multi-objective fuzzy genetics-based machine learning," *Int. J. Approx. Reason.*, vol. 44, no. 1, pp. 4–31, 2007.
[5] S. Abe and M. S. Lan, "A method for fuzzy rules extraction directly from numerical data and its application to pattern classification," *IEEE Trans. Fuzzy Syst.*, vol. 3, no. 1, pp. 18–28, Feb. 1995.
[6] H. Ishibuchi, K. Nozaki, and H. Tanaka, "Distributed representation of fuzzy rules and its application to pattern classification," *Fuzzy Sets Syst.*, vol. 52, no. 1, pp. 21–32, 1992.
[7] D. Chakraborty and N. R. Pal, "A neuro-fuzzy scheme for simultaneous feature selection and fuzzy rule-based classification," *IEEE Trans. Neural Netw.*, vol. 15, no. 1, pp. 110–123, Jan. 2004.
[8] S. Mitra and Y. Hayashi, "Neuro-fuzzy rule generation: Survey in soft computing framework," *IEEE Trans. Neural Netw.*, vol. 11, no. 3, pp. 748–768, May 2000.
[9] S. Abe and R. Thawonmas, "A fuzzy classifier with ellipsoidal regions," *IEEE Trans. Fuzzy Syst.*, vol. 5, no. 3, pp. 358–368, Aug. 1997.
[10] R. Alcalá, J. Alcalá-Fdez, F. Herrera, and J. Otero, "Genetic learning of accurate and compact fuzzy rule based systems based on the 2-tuples linguistic representation," *Int. J. Approx. Reason.*, vol. 44, pp. 45–64, 2007.
[11] O. Cordón, F. Gomide, F. Herrera, F. Hoffmann, and L. Magdalena, "Ten years of genetic fuzzy systems: Current framework and new trends," *Fuzzy Sets Syst.*, vol. 41, pp. 5–31, 2004.
[12] H. Ishibuchi, T. Nakashima, and T. Murata, "Three-objective genetics-based machine learning for linguistic rule extraction," *Inf. Sci.*, vol. 136, no. 1–4, pp. 109–133, 2001.
[13] J. V. de Oliveira, "Semantic constraints for membership function optimization," *IEEE Trans. Fuzzy Syst.*, vol. 19, no. 1, pp. 128–138, Jan. 1999.
[14] J. R. Cano, F. Herrera, and M. Lozano, "Evolutionary stratified training set selection for extracting classification rules with trade off precision-interpretability," *Data Knowl. Eng.*, vol. 60, pp. 90–108, 2007.
[15] J. A. Roubos, M. Setnes, and J. Abonyi, "Learning fuzzy classification rules from labeled data," *IEEE Trans. Fuzzy Syst.*, vol. 8, no. 5, pp. 509–522, May 2001.

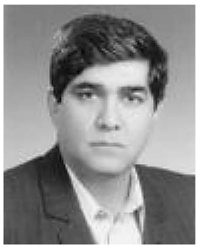
[16] H. Ishibuchi and Y. Nojima, "Evolutionary multiobjective optimization for the design of fuzzy rule-based ensemble classifiers," *Int. J. Hybrid Intell. Syst.*, vol. 3, no. 3, pp. 129–145, 2006.
[17] H. Ishibuchi and T. Yamamoto, "Fuzzy rule selection by multi-objective genetic local search algorithms and rule evaluation measures in data mining," *Fuzzy Sets Syst.*, vol. 141, no. 1, pp. 59–88, 2004.
[18] C. Setzkorn and R. C. Paton, "On the use of multi-objective evolutionary algorithms for the induction of fuzzy classification rule systems," *BioSystems*, vol. 81, pp. 101–112, 2005.
[19] H. Wang, S. Kwong, Y. Jin, W. Wei, and K. F. Man, "Multiobjective hierarchical genetic algorithm for interpretable fuzzy rule-based knowledge extraction," *Fuzzy Sets Syst.*, vol. 149, pp. 149–186, 2005.
[20] K. A. De Jong, W. M. Spears, and D. F. Gordon, "Using genetic algorithms for concept learning," *Mach. Learn.*, vol. 13, pp. 161–188, 1993.
[21] S. F. Smith, "A learning system based on genetic algorithms," Ph.D. dissertation, Dept. Comput. Sci., Univ. Pittsburgh, Pittsburgh, PA, 1980.
[22] J. H. Holland, *Adaptation in Natural and Artificial Systems*. Ann Arbor, MI: Univ. Michigan Press, 1975.
[23] T. Kovacs, *Strength or Accuracy: Credit Assignment in Learning Classifier Systems*. New York: Springer-Verlag, 2004.
[24] D. P. Greene and S. F. Smith, "Competition-based induction of decision models from examples," *Mach. Learn.*, vol. 3, pp. 229–257, 1993.
[25] M. L. Wong and K. S. Leung, *Data Mining Using Grammar Based Genetic Programming and Applications*. Norwell, MA: Kluwer, 2000.
[26] J. Casillas, B. Carso, and L. Bull, "Fuzzy XCS: A Michigan genetic fuzzy system," *IEEE Trans. Fuzzy Syst.*, vol. 15, no. 4, pp. 536–550, Aug. 2007.
[27] A. Gonzalez and F. Herrera, "Multi-stage genetic fuzzy systems based on the iterative rule learning approach," *Mathware Soft Comput.*, vol. 4, pp. 233–249, 1997.
[28] F. Herrera, "Genetic fuzzy systems: Status, critical considerations and future directions," *Int. J. Comput. Intell. Res.*, vol. 1, no. 1, pp. 59–67, 2005.
[29] H. Ishibuchi, T. Yamamoto, and T. Nakashima, "Hybridization of fuzzy GBML approaches for pattern classification problems," *IEEE Trans. Syst., Man, Cybern. B*, vol. 35, no. 2, pp. 359–365, Apr. 2005.
[30] L. Castillo, A. Gonzalez, and R. Perez, "Including a simplicity criterion in the selection of the best rule in a genetic fuzzy learning algorithm," *Fuzzy Sets Syst.*, vol. 120, no. 2, pp. 309–321, 2001.
[31] A. Gonzalez and R. Perez, "SLAVE: A genetic learning system based on an iterative approach," *IEEE Trans. Fuzzy Syst.*, vol. 7, no. 2, pp. 176–191, Apr. 1999.
[32] H. Ishibuchi and T. Yamamoto, "Comparison of heuristic criteria for fuzzy rule selection in classification problems," *Fuzzy Opt. Dec. Making*, vol. 3, no. 2, pp. 119–139, 2004.
[33] O. Cordon, F. Herrera, F. Hoffman, and L. Magdalena, *Genetic Fuzzy Systems*. Singapore: World Scientific, 2001.
[34] H. Ishibuchi and T. Nakashima, "Effect of rule weights in fuzzy rule-based classification systems," *IEEE Trans. Fuzzy Syst.*, vol. 9, no. 4, pp. 506–515, Aug. 2001.
[35] H. Ishibuchi, T. Nakashima, and T. Morisawa, "Voting in fuzzy rule-based systems for pattern classification problems," *Fuzzy Sets Syst.*, vol. 103, no. 2, pp. 223–238, 1999.
[36] O. Cordon, M. J. del Jesus, and F. Herrera, "A proposal on reasoning methods in fuzzy rule-based classification systems," *Int. J. Approx. Reason.*, vol. 20, pp. 21–45, 1999.
[37] H. Ishibuchi and T. Yamamoto, "Rule weight specification in fuzzy rule-based classification systems," *IEEE Trans. Fuzzy Syst.*, vol. 13, no. 4, pp. 428–435, Aug. 2005.
[38] M. J. Zolghadri and E. G. Mansoori, "Weighting fuzzy classification rules using receiver operating characteristics (ROC) analysis," *Inf. Sci.*, vol. 177, no. 11, pp. 2296–2307, 2007.
[39] E. G. Mansoori, M. J. Zolghadri, and S. D. Katebi, "A weighting function for improving fuzzy classification systems performance," *Fuzzy Sets Syst.*, vol. 158, no. 5, pp. 583–591, 2007.
[40] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A. I. Verkamo, "Fast discovery of association rules," in *Advances in Knowledge Discovery and Data Mining*, U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, Eds. Menlo Park, CA: AAAI Press, 1996.
[41] T.-P Hong, C.-S. Kuo, and S.-C. Chi, "Trade-off between computation time and number of rules for fuzzy mining from quantitative data," *Int. J. Uncertainty, Fuzziness Knowl.-Based Syst.*, vol. 9, no. 5, pp. 587–604, 2001.
[42] H. Ishibuchi, T. Murata, and I. B. Turksen, "Single-objective and two-objective genetic algorithms for selecting linguistic rules for pattern classification problems," *Fuzzy Sets Syst.*, vol. 89, no. 2, pp. 135–150, 1997.
[43] D. Whitley, "A genetic algorithm tutorial," Colorado State Univ., Fort Collins, Rep. CS-93-103, 1993, pp. 29–31.
[44] L. Eshelman, "The CHC adaptive search algorithm," in *Foundations of Genetic Algorithms*, G. Rawlins, Ed. San Mateo, CA: Morgan-Kaufmann, 1991, pp. 256–283.
[45] C. L. Blake and C. J. Merz, *UCI Repository of Machine Learning Databases*. Irvine, CA: Dept. Inf. Comput. Sci., Univ. California, 1998.
[46] J. R. Quinlan, "Improved use of continuous attributes in C4.5," *J. Artif. Intell. Res.*, vol. 4, pp. 77–90, 1996.
[47] T. Elomaa and J. Rousu, "General and efficient multisplitting of numerical attributes," *Mach. Learn.*, vol. 36, pp. 201–244, 1999.
[48] L. Sanchez, I. Couso, and J. A. Corrales, "Combining GP operators with SA search to evolve fuzzy rule-based classifiers," *Inf. Sci.*, vol. 136, no. 1–4, pp. 175–191, 2001.
[49] J. Abonyi, J. A. Roubos, and F. Szeifert, "Data-driven generation of compact, accurate, and linguistically-sound fuzzy classifiers based on a decision-tree initialization," *Int. J. Approx. Reason.*, vol. 32, no. 1, pp. 1–21, 2003.
[50] S. U. Guan and F. Zhu, "Class decomposition for GA-based classifier agents—A pitt approach," *IEEE Trans. Syst., Man, Cybern. B: Cybern.*, vol. 34, no. 1, pp. 381–392, Feb. 2004.

Eghbal G. Mansoori received the B.Sc. degree in computer engineering in 1992, the M.Sc. degree in artificial intelligence in 1996, and the Ph.D. degree in artificial intelligence (bioinformatics) in Jan. 2008 from Shiraz University, Shiraz, Iran.

Since January 2008, he has been with the Department of Computer Science and Engineering, Shiraz University as an Assistant Professor. His current research interests include genetic fuzzy systems, bioinformatics, machine vision, and biological signal processing.

Mansoor J. Zolghadri received the B.Sc. degree in mechanical engineering from Shiraz University, Shiraz, Iran, in 1979, the M.Sc. and Ph.D. degrees in instrumentation and control from the University of Bradford, Bradford, U.K., in 1982 and 1988, respectively.

Since 1988, he has been with the Department of Computer Science and Engineering, Shiraz University, where he is currently an Associate Professor. He is the author or coauthor of more than 40 papers published in various international journals and conference proceedings. His current research interests include pattern recognition, fuzzy systems, information retrieval, and web search.



Seraj D. Katebi received the Graduate (Hons.) degree in computer systems engineering from Coventry University, Coventry, U.K., in 1972, and the M.Sc. and Ph.D. degrees in automatic control from the Control Systems Center, University of Manchester Institute of Science and Technology (UMIST), Manchester, U.K., in 1973 and 1976, respectively.

In 1976, he joined the Department of Computer Science and Engineering, Shiraz University, Shiraz, Iran, where he has been a Full Professor since 1993, and is engaged in teaching undergraduate and graduate courses and conducting research in various aspects of nonlinear control and artificial intelligence (AI). He is the author or coauthor of several papers published in various international journals.