ELSEVIER

Information Sciences 136 (2001) 109–133

INFORMATION SCIENCES
AN INTERNATIONAL JOURNAL

www.elsevier.com/locate/ins

Three-objective genetics-based machine learning for linguistic rule extraction

Hisao Ishibuchi [a,*], Tomoharu Nakashima [a], Tadahiko Murata [b]

[a] *Department of Industrial Engineering, Osaka Prefecture University, Gakuen-cho 1-1, Sakai, Osaka 599-8531, Japan*

[b] *Department of Industrial and Information Systems Engineering, Ashikaga Institute of Technology, 268 Omae-cho, Ashikaga, Tochigi 326-8558, Japan*

Abstract

This paper shows how a small number of linguistically interpretable fuzzy rules can be extracted from numerical data for high-dimensional pattern classification problems. One difficulty in the handling of high-dimensional problems by fuzzy rule-based systems is the exponential increase in the number of fuzzy rules with the number of input variables. Another difficulty is the deterioration in the comprehensibility of fuzzy rules when they involve many antecedent conditions. Our task is to design comprehensible fuzzy rule-based systems with high classification ability. This task is formulated as a combinatorial optimization problem with three objectives: to maximize the number of correctly classified training patterns, to minimize the number of fuzzy rules, and to minimize the total number of antecedent conditions. We show two genetic-algorithm-based approaches. One is rule selection where a small number of linguistically interpretable fuzzy rules are selected from a large number of prespecified candidate rules. The other is fuzzy genetics-based machine learning where rule sets are evolved by genetic operations. These two approaches search for non-dominated rule sets with respect to the three objectives.

Keywords: Pattern classification; Fuzzy systems; Genetic algorithms; Rule extraction

* Corresponding author. Tel.: +81-722-54-9350; fax: +81-722-54-9915.

E-mail addresses: hisaoi@ie.osakafu-u.ac.jp (H. Ishibuchi), nakashi@ie.osakafu-u.ac.jp (T. Nakashima), murata@ashitech.ac.jp (T. Murata).

0020-0255/01/$ - see front matter
PII: S0020-0255(01)00144-X

1. Introduction

Recently fuzzy rule-based systems have been applied to pattern classification problems [24]. Main characteristic features of fuzzy rule-based systems are their nonlinearity and comprehensibility. Since fuzzy rule-based systems are universal approximators of nonlinear functions [23,37], they can handle highly nonlinear classification problems. Many studies on fuzzy rule-based classification systems tried to improve their classification ability. Some studies used neuro-fuzzy techniques [1,27,29,30], and other studies employed genetic algorithms for designing fuzzy rule-based classification systems [4,12,17,40]. In those studies, emphasis is placed on the classification ability of fuzzy rule-based systems rather than their comprehensibility. That is, error rates on training patterns are minimized by various techniques. Another research direction in fuzzy rule-based systems is to extract interpretable knowledge from numerical data in the form of fuzzy rules [11,20,21,31–34,38,39]. In this research direction, the design of fuzzy rule-based systems is viewed as finding a small number of comprehensible fuzzy rules with clear interpretations. That is, emphasis is placed on the comprehensibility of fuzzy rule-based systems. This research direction is closely related to data mining and knowledge discovery.

Our task in this paper is to design comprehensible fuzzy rule-based systems for high-dimensional pattern classification problems. One difficulty in the handling of high-dimensional problems by fuzzy rule-based systems is the exponential increase in the number of fuzzy rules with the dimensionality of input spaces. While fuzzy rules for two-dimensional problems can be concisely written in the form of fuzzy rule tables, it is impractical to generate all fuzzy rules in the tabular form for high-dimensional problems. A straightforward approach for avoiding the exponential increase in the number of fuzzy rules is the input selection [15] where only a few input variables are selected for designing fuzzy rule-based systems (for general discussions on feature selection, see [6,22,25,26]). While it was shown that very simple rules performed well on some classification problems [7,15], the effectiveness of input selection is problem-dependent. Another approach is the use of multi-dimensional antecedent fuzzy sets [2]. In this approach, antecedent conditions of fuzzy rules are not linguistically described by single-dimensional fuzzy sets on each axis, but defined by multi-dimensional fuzzy sets. Clustering techniques are usually used for identifying multi-dimensional antecedent fuzzy sets, which can be tuned further by learning algorithms. This approach is applicable to high-dimensional problems because the number of fuzzy rules is independent of the number of input variables. From the viewpoint of classification performance, the use of multi-dimensional antecedent fuzzy sets may be the most effective approach to the handling of high-dimensional problems by fuzzy rule-based systems. This approach, however, has an inherent difficulty from the viewpoint of comprehensibility. That is, it is very difficult for users to linguistically in-

terpret multi-dimensional antecedent fuzzy sets. For generating linguistically interpretable fuzzy rules, multi-dimensional antecedent fuzzy sets are projected onto each axis [33,34]. Such projection usually deteriorates the performance of fuzzy rule-based systems. Fuzzy rules generated by the projection have many antecedent conditions in the case of high-dimensional problems. Another approach to the handling of high-dimensional problems is the use of hierarchical fuzzy rule-based systems where a number of low-dimensional fuzzy rule tables are hierarchically connected [8]. Outputs from fuzzy rule tables in lower layers are used as inputs to those in higher layers. Since each fuzzy rule table has only a few input variables, the exponential increase in the number of fuzzy rules is avoided. This approach has the same difficulty as the use of multi-dimensional antecedent fuzzy sets. That is, hierarchical fuzzy rule-based systems are not comprehensible. Especially, it is very difficult for users to interpret fuzzy rule tables with intermediate variables that are not input or output variables of the whole fuzzy rule-based system.

We use the following fuzzy rules for an n-dimensional pattern classification problem with c classes and n continuous attributes:

$$\text{Rule } R_j: \text{If } x_1 \text{ is } A_{j1} \text{ and } \ldots \text{ and } x_n \text{ is } A_{jn} \text{ then Class } C_j \text{ with } CF_j, \quad j = 1, 2, \ldots, N, \tag{1}$$

where R_j is the label of the jth fuzzy rule, $\vec{x} = (x_1, \ldots, x_n)$ is an n-dimensional pattern vector, A_{ji} is a linguistic value such as *small* and *large* for the ith attribute, C_j is a consequent class (i.e., one of the c classes), CF_j is a certainty grade in the unit interval $[0, 1]$, and N is the number of fuzzy rules. Let us assume that we have K_i linguistic values for describing the ith attribute ($i = 1, 2, \ldots, n$). One of those K_i linguistic values is used as the antecedent fuzzy set A_{ji} for the ith attribute in each fuzzy rule. In this case, the total number of possible combinations of antecedent linguistic values is $K_1 \times K_2 \times \cdots \times K_n$. It is a natural idea to examine all the combinations for generating fuzzy rules when our pattern classification problem involves only a few attributes (i.e., when n is small). On the contrary, it is impractical to examine all the combinations when n is large. Due to the exponential increase in the number of fuzzy rules, the comprehensibility of fuzzy rule-based systems is drastically impaired as the dimensionality of the pattern space increases. The increase in the number of antecedent conditions also impairs the comprehensibility of each fuzzy rule. In general, it is much easier for users to intuitively understand short fuzzy rules with only a few antecedent conditions than long rules with many conditions [20].

We construct comprehensible fuzzy rule-based systems for high-dimensional pattern classification problems using a small number of short fuzzy rules with clear linguistic interpretations. Only a few antecedent conditions are specified by linguistic values in our fuzzy rules. The other conditions are viewed as

"*don't care*" conditions. This can be implemented by considering don't care as an additional linguistic value. In this case, each antecedent fuzzy set A_{ji} of our fuzzy rules in (1) is selected from the given K_i linguistic values and don't care. Since don't care conditions can be omitted, fuzzy rules with many don't care conditions are written as short fuzzy rules. For example, in a computer simulation described in Section 6, we obtained the following fuzzy rules for a 13-dimensional pattern classification problem:

$$\text{If } x_7 \text{ is medium and } x_{11} \text{ is medium then Class 1 with } CF_1 = 0.56, \quad (2)$$

$$\text{If } x_{10} \text{ is small then Class 2 with } CF_2 = 0.94, \quad (3)$$

$$\text{If } x_7 \text{ is small then Class 3 with } CF_3 = 0.85. \quad (4)$$

Since we use don't care as an additional linguistic value, the total number of possible combinations of antecedent linguistic values is $(K_1 + 1) \times \cdots \times (K_n + 1)$. Our task in this paper is to design comprehensible fuzzy rule-based systems by searching for a small number of fuzzy rules in the huge search space with those combinations. The difficulty of our task lies in the size of the search space. In the following sections, we show how genetic algorithms can tackle this difficult task.

2. Problem formulation

Let us assume that we have m training patterns $\overrightarrow{x_p} = (x_{p1}, \ldots, x_{pn})$, $p = 1, 2, \ldots, m$ from c classes. For simplicity of explanation, each attribute value is assumed to be a real number in the unit interval $[0, 1]$. This means that the pattern space of our pattern classification problem is the n-dimensional unit hyper-cube $[0, 1]^n$. In computer simulations of this paper, every attribute value was normalized into a real number in the unit interval $[0, 1]$.

We also assume that K_i linguistic values have already been given by users for describing the ith attribute ($i = 1, 2, \ldots, n$). That is, we assume that the membership function of each linguistic value has already been specified by the users based on their domain knowledge and intuition. We do not modify the given membership function because such modification may cause a gap between the modified membership function and the users' understanding of each linguistic value. Any shapes of membership functions can be handled by our approaches. The point is that the membership function of each linguistic value should be consistent with the users' domain knowledge and intuition. In computer simulations of this paper, we used a typical set of linguistic values with triangular membership functions for each attribute. Examples of such linguistic values are shown in Fig. 1. These linguistic values are used in our computer simulations for all attributes. This is just for simplicity of explanation. Our approaches are applicable to arbitrarily given sets of linguistic values.

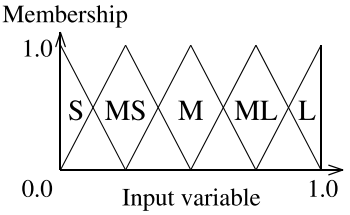

Fig. 1. Typical examples of linguistic values (S: small, MS: medium small, M: medium, ML: medium large, and L: large).

Since the consequent class and the certainty grade of each fuzzy rule in (1) can be easily determined by a heuristic rule generation procedure [18] from the given training patterns (see Appendix A), the design of fuzzy rule-based systems is to determine the number of fuzzy rules and the antecedent part of each rule. This task is to find a small number of combinations of antecedent linguistic values from the huge search space with $(K_1 + 1) \times \cdots \times (K_n + 1)$ combinations.

As we have already mentioned, the comprehensibility of fuzzy rule-based systems is impaired by the increase in the number of fuzzy rules. Thus we try to minimize the number of fuzzy rules. From the viewpoint of the comprehensibility of each fuzzy rule, a large number of antecedent conditions is not desirable. Thus we also try to minimize the number of antecedent conditions. At the same time, we want to design fuzzy rule-based systems with high classification performance. Based on these discussions, we formulate our task of designing comprehensible fuzzy rule-based classification systems as the following three-objective optimization problem:

$$\text{Maximize } f_1(S), \text{ minimize } f_2(S), \text{ and minimize } f_3(S), \tag{5}$$

where $f_1(S)$ is the number of correctly classified training patterns by a rule set S, $f_2(S)$ is the number of fuzzy rules in S, and $f_3(S)$ is the total number of antecedent conditions in S. For example, when the rule set S consists of the three fuzzy rules in (2)–(4), $f_2(S)$ and $f_3(S)$ are calculated as $f_2(S) = 3$ and $f_3(S) = 4$, respectively. The first objective $f_1(S)$ is calculated by classifying all the training patterns by the rule set S. In Appendix B, we show how each training pattern is classified by fuzzy rules in the rule set S. Our task is to find non-dominated rule sets with respect to the three objectives.

Let us briefly explain the concept of non-dominated rule sets for our three-objective optimization problem. A rule set S is said to be dominated by another rule set S^* if all the following inequalities hold:

$$f_1(S) \leqslant f_1(S^*), \quad f_2(S) \geqslant f_2(S^*), \quad f_3(S) \geqslant f_3(S^*), \tag{6}$$

and at least one of the following inequalities holds:

$$f_1(S) < f_1(S^*), \quad f_2(S) > f_2(S^*), \quad f_3(S) > f_3(S^*). \tag{7}$$

The first condition (i.e., all the three inequalities in (6)) means that no objective of S^* is worse than S. The second condition (i.e., one of the three inequalities in (7)) means that at least one objective of S^* is better than S. If there exists no S^* that satisfies both the above two conditions, the rule set S is said to be a non-dominated rule set.

It should be noted that the third objective $f_3(S)$ is not the average number of antecedent conditions in each fuzzy rule but the total number of antecedent conditions in the rule set S. Let $f_{3^*}(S)$ be the average number of antecedent conditions in each fuzzy rule. For example, $f_{3^*}(S)$ is calculated as $f_{3^*}(S) = 1.33$ for the rule set S with R_1, R_2 and R_3 in (2)–(4). Let us construct another rule set S^+ by adding the following fuzzy rule with a single antecedent condition to S:

$$R_4 : \text{If } x_4 \text{ is small then Class 2 with } CF_4 = 0.68. \tag{8}$$

For the rule set S^+ with $R_1 \sim R_4$, $f_{3^*}(S^+)$ is calculated as $f_{3^*}(S^+) = 1.25$. That is, the average number of antecedent conditions is improved by adding R_4 to S while the complexity of the rule set is increased. Even if the added fuzzy rule R_4 does not improve the classification performance of the rule set (i.e., $f_1(S) \geqslant f_1(S^+)$), S^+ is not dominated by S when we use the average number of antecedent conditions as the third objective. This example shows that the average number of antecedent conditions is not an appropriate index for measuring the simplicity of fuzzy rules in the context of multi-objective optimization. Thus we use the total number of antecedent conditions as the third objective $f_3(S)$.

3. Three-objective genetic algorithms

We use three-objective genetic algorithms for finding non-dominated rule sets. Standard single-objective genetic algorithms are also applicable to our problem if the three objectives are integrated into a single scalar fitness function. Before describing three-objective genetic algorithms, we briefly discuss the handling of our problem by single-objective genetic algorithms.

A well-known simple trick for handling multi-objective optimization problems is to combine multiple objectives into a single scalar fitness function using weight parameters as

$$\text{fitness}(S) = w_1 \cdot f_1(S) - w_2 \cdot f_2(S) - w_3 \cdot f_3(S), \tag{9}$$

where w_1, w_2 and w_3 are non-negative real numbers. In (9), the two objectives $f_2(S)$ and $f_3(S)$ to be minimized can be viewed as having negative weights "$-w_2$" and "$-w_3$", respectively. The three weights w_1, w_2 and w_3 in (9) should be specified based on the users' preference in a particular pattern classification problem. It is, however, difficult to assign appropriate values to the three weights.

If the users have a desired goal related to a particular objective, such an objective can be handled as a constraint condition. The constraint condition for each objective can be written as $f_1(S) \geqslant f_1^*$, $f_2(S) \leqslant f_2^*$ and $f_3(S) \leqslant f_3^*$ where f_1^*, f_2^* and f_3^* are the desired goals for $f_1(S)$, $f_2(S)$ and $f_3(S)$, respectively. If desired goals for two objectives are given by the users, our three-objective optimization problem can be rewritten as a single-objective optimization problem with two inequality conditions. For example, when the desired goals f_1^* and f_2^* are given, we have the following single-objective optimization problem:

$$\text{Minimize } f_3(S) \text{ subject to } f_1(S) \geqslant f_1^* \text{ and } f_2(S) \leqslant f_2^*. \tag{10}$$

This problem can be handled as the maximization problem of the following scalar fitness function:

$$\text{fitness}(S) = -w \cdot \max\{0, f_1^* - f_1(S)\} - w \cdot \max\{0, f_2(S) - f_2^*\} - f_3(S), \tag{11}$$

where w is a large positive weight value (e.g., $w = 100$). It should be noted that the first and second terms of (11) are zero when the corresponding inequality conditions hold in (10).

In this paper, we use a multi-objective genetic algorithm for finding non-dominate rule sets of our three-objective optimization problem. Multi-objective genetic algorithms do not require the specification of the weight parameters or the desired goals. In our former study [11], we showed how a two-objective genetic algorithm can be used for obtaining non-dominated rule sets with respect to $f_1(S)$ and $f_2(S)$. In this paper, we use a multi-objective genetic algorithm [9,28], which is based on the scalar fitness function in (9) with random weight values. The weight values w_1, w_2, and w_3 are randomly updated whenever a pair of parent strings are selected. This is one characteristic feature of our multi-objective genetic algorithm. Another characteristic feature is that non-dominated rule sets are stored in a tentative pool separately from the current population. The tentative pool is updated at every generation in order to store only non-dominated rule sets among examined ones. From the tentative pool, N_{elite} rule sets are randomly selected as elite individuals, which are added to a new population. The outline of our three-objective genetic algorithm is written as follows.

Three-objective genetic algorithm

Step 1 *Initialization*. Generate an initial population of N_{set} rule sets where N_{set} is the population size.

Step 2 *Evaluation*. Calculate the values of the three objectives for each rule set in the current population. Then update the tentative pool of non-dominated rule sets.

Step 3 *Selection*. Repeat the following procedures to select $(N_{\text{set}} - N_{\text{elite}})$ pairs of rule sets.

1. Randomly specify the three weight values as

$$w_i = \text{random}_i/(\text{random}_1 + \text{random}_2 + \text{random}_3), \quad i = 1, 2, 3, \tag{12}$$

   where random_i is a non-negative random real number.
2. Calculate the fitness value of each rule set using (9) with the randomly specified weight values. Then select a pair of rule sets based on the fitness value of each rule set. We specify the selection probability of each rule set S in the current population Ψ using the roulette wheel selection with the linear scaling:

$$P(S) = \frac{\text{fitness}(S) - f_{\min}(\Psi)}{\sum_{S \in \Psi}\{\text{fitness}(S) - f_{\min}(\Psi)\}}, \tag{13}$$

   where $f_{\min}(\Psi)$ is the minimum fitness value in the current population Ψ.

Step 4 *Crossover and mutation*. Generate a new rule set from each pair of selected rule sets by crossover and mutation operations. These two operations are used with prespecified probabilities. By the genetic operations, $(N_{\text{set}} - N_{\text{elite}})$ rule sets are generated.

Step 5 *Elitist strategy*. Randomly select N_{elite} non-dominated rule sets from their tentative pool, and add them to the generated $(N_{\text{set}} - N_{\text{elite}})$ rule sets for constructing a new population of the size N_{set}.

Step 6 *Termination test*. If a prespecified stopping condition is satisfied, end the algorithm. Otherwise, return to Step 2.

The choice of crossover and mutation operations in Step 4 depends on the coding of rule sets. They are described in the following sections. We use this three-objective genetic algorithm because it can be easily implemented. This algorithm involves no additional parameters. It uses only standard parameters such as the population size, the crossover probability, the mutation probability, and the number of elite solutions. Other multi-objective genetic algorithms are also applicable to our three-objective optimization problem. For reviews of multi-objective genetic algorithms, see [36,41,42].

4. Rule selection

We have already proposed a two-objective rule selection method [11] for maximizing the classification performance and minimizing the number of fuzzy rules. In the rule selection method, all combinations of antecedent linguistic values were examined to generate candidate rules from which a small number of fuzzy rules were selected by genetic algorithms. This method cannot be directly applied to high-dimensional problems because the number of candidate

rules exponentially increases with the dimensionality of pattern spaces. In this section, we describe how our rule selection method can be extended to the case of high-dimensional problems. We use a prescreening procedure of candidate rules that was introduced for rule selection with a scalar fitness function [10] and two-objective rule selection [14].

4.1. Candidate rule generation

Our trick for prescreening candidate rules is based on the length (i.e., the number of antecedent conditions) of fuzzy rules. While the total number of possible combinations of antecedent linguistic values is huge (i.e., $(K_1 + 1) \times \cdots \times (K_n + 1)$), the number of short fuzzy rules with only a few antecedent conditions is not large. Thus we can generate a tractable number of candidate rules by examining only short fuzzy rules. When we have five linguistic values, the number of fuzzy rules of the length k is calculated as ${}_nC_k \times 5^k$, which is the total number of combinations of selecting k attributes (i.e., ${}_nC_k$) and assigning linguistic values to the selected attributes (i.e., 5^k).

4.2. Genetic operations for rule selection

Let N be the number of generated candidate rules. As in our previous studies [10,11,14], a subset S of the N candidate rules is denoted by a binary string of the length N as $S = s_1 s_2 \ldots s_N$. In this coding, $s_j = 1$ and $s_j = 0$ mean that the jth candidate rule R_j is included in S and excluded from S, respectively. The size of the search space with this coding is 2^N, which is the total number of subsets of the N candidate rules. Each rule set S is evaluated by the fitness function in (9) using randomly specified three weights whenever a pair of parent strings is selected. Since each rule set is represented by a binary string, standard genetic operations are applicable. In our computer simulations, we used the uniform crossover and the biased mutation.

For efficiently searching for small rule sets with high classification ability, we use two domain-specific techniques. One technique is a kind of local search. When the fitness value of a binary string S (i.e., rule set S) is calculated, all the given training patterns are classified by S for calculating the first objective $f_1(S)$. As shown in Appendix B, a single winner rule is responsible for the classification of each training pattern (for other kinds of fuzzy reasoning, see [5]). If a fuzzy rule in S is responsible for the classification of no training pattern, we can remove that rule without causing any deterioration in the first objective because that rule has no influence on the classification of any training pattern. At the same time, the elimination of such a fuzzy rule improves the second objective $f_2(S)$ and the third objective $f_3(S)$. Thus we remove all the fuzzy rules that are not responsible for the classification of any training

pattern. This local search technique is applied to every rule set before its three objectives are evaluated in Step 2 of our three-objective genetic algorithm.

The other technique is to bias the mutation. A larger probability was assigned to the mutation from $s_j = 1$ to 0 than the mutation from $s_j = 0$ to 1. That is, the mutation is biased toward the decrease in the number of fuzzy rules in order to improve the second and third objectives. The biased mutation plays an important role especially when the number of candidate rules is large.

These two techniques are added to the three-objective genetic algorithm in the previous section. A number of randomly generated initial rule sets (i.e., binary strings) are evolved by the genetic operations for finding non-dominated rule sets of our three-objective optimization problem.

5. Fuzzy genetics-based machine learning

The effectiveness of the rule selection method in the previous section strongly depends on the choice of candidate rules (i.e., on the choice of a prescreening procedure). If candidate rules are inappropriately specified, it is impossible for the three-objective genetic algorithm to find good rule sets. In this section, we describe a fuzzy genetics-based machine learning (GBML) algorithm, which does not require any prescreening procedure. Non-dominated rule sets are found from all combinations of antecedent linguistic values. Our fuzzy GBML algorithm is a hybridization of the two approaches in the GBML research: the Michigan approach [3] and the Pittsburgh approach [35]. In the Michigan approach, a single rule is handled as an individual. On the other hand, a rule set is handled as an individual in the Pittsburgh approach. Each approach has its own advantage and disadvantage in the application to the design of fuzzy rule-based systems [13,16]. In this section, we show how these two approaches can be combined into a single hybrid algorithm to search for non-dominated rule sets of our three-objective optimization problem. Hybrid fuzzy GBML algorithms were proposed for single-objective and two-objective problems in [13,14].

5.1. Coding of fuzzy rules and rule sets

Since the consequent class and the certainty grade can be easily determined by a heuristic procedure [18] from the given training patterns as shown in Appendix A, our hybrid algorithm is used for determining the number of fuzzy rules and the antecedent part of each rule. Thus only the antecedent part is coded as a string. The fuzzy rule R_j in (1) is denoted by its n antecedent linguistic values as $R_j = A_{j1}A_{j2}\ldots A_{jn}$. For simplicity of explanation, we use an alphabet with some simple symbols. For example, when the five linguistic values in Fig. 1 and don't care are used as antecedent linguistic values for all the n attributes, they are represented by an alphabet with the following six

symbols. 1: *small*, 2: *medium small*, 3: *medium*, 4: *medium large*, 5: *large*, and #:*don't care*. In this case, the fuzzy rule R_j is denoted by a string of the length n consisting of these six symbols. For example, a string "1#35" denotes the antecedent part "If x_1 is small and x_2 is don't care and x_3 is medium and x_4 is large". A rule set S is denoted by a concatenated string where each substring of the length n denotes a fuzzy rule in S. For example, "1#35 23#4 4#11 1#25 1##1" denotes a rule set with five fuzzy rules for a four-dimensional pattern classification problem.

5.2. Genetic operations in our hybrid algorithm

In our hybrid fuzzy GBML algorithm, a rule set is represented by a concatenated string and handled as an individual. In this sense, our algorithm is based on the Pittsburgh approach. We use the following tricks for improving the search ability of our algorithm to efficiently find non-dominated rule sets.

(a) Hybridization with a Michigan-style fuzzy GBML algorithm.
(b) Heuristic generation of an initial population.
(c) Rule generation from misclassified or rejected training patterns.
(d) Adjustable string length for minimizing the number of fuzzy rules.

The first trick is the hybridization of the Pittsburgh approach and the Michigan approach. In our hybrid algorithm, a Michigan-style algorithm (i.e., fuzzy classifier system [12,17]) is used as a mutation operation of a Pittsburgh-style algorithm. In our former studies [13,16], we examined advantages and disadvantages of these two individual algorithms when they are used for maximizing the classification performance of fuzzy rule-based systems. The main finding in those studies is that each individual algorithm has its own advantages and disadvantages. For example, Michigan-style algorithms have high search ability to find good fuzzy rules in the huge search space while it does not work well for finding good rule sets. This is because the classification performance of a rule set is not used as a fitness function in the Michigan approach. Only the classification performance of each fuzzy rule is used as a fitness function. On the contrary, Pittsburgh-style algorithms work well for finding good rule sets while it does not have high search ability to find good fuzzy rules in the huge search space. The Pittsburgh approach can directly optimize rule sets by maximizing a fitness function while the optimization of rule sets is indirect in the Michigan approach. The aim of the hybridization of these two approaches is to implement the advantages of each approach in a single hybrid algorithm. In our hybrid algorithm, a Michigan-style algorithm [12,17] is used as a kind of mutation for partially modifying each rule set in the current population. The outline of the Michigan-style fuzzy GBML algorithm used in our hybrid algorithm is shown in Appendix C.

Since the search space is huge, the classification performance of randomly generated rule sets is usually very poor [12,17]. Thus we do not randomly

generate initial rule sets. They are directly generated from training patterns in a heuristic manner. For generating N_{rule} fuzzy rules, first we randomly select N_{rule} training patterns. A single fuzzy rule is generated from each training pattern. Let $\overrightarrow{x_p} = (x_{p1}, \ldots, x_{pn})$ be one of the selected training patterns. A fuzzy rule R_j is directly generated from $\overrightarrow{x_p}$ in the following heuristic manner.

Direct rule generation procedure

Step 1. Select the most compatible linguistic value for the ith attribute value x_{pi} of the training pattern $\overrightarrow{x_p}$ from the given K_i linguistic values as the ith antecedent fuzzy set A_{ji} for $i = 1, 2, \ldots, n$. In this step, don't care is not selected as antecedent fuzzy sets because it always has the highest compatibility grade with any attribute values.
Step 2. With a prespecified probability P_{DC}, replace each antecedent fuzzy set A_{ji} with don't care. In our computer simulations, we specified P_{DC} as 0.5. This means that half of the antecedent fuzzy sets were replaced with don't care on the average.
Step 3. Specify the consequent class C_j and the certainty grade CF_j by the heuristic procedure in Appendix A using all the given training patterns.

This trick can be also used for generating new fuzzy rules in the generation update process of the Michigan-style algorithm. When a training pattern $\overrightarrow{x_p} = (x_{p1}, \ldots, x_{pn})$ is misclassified or rejected by the current population (i.e., by the rule set S) in the Michigan-style algorithm, a new rule is directly generated from $\overrightarrow{x_p}$ by the above procedure. That is, some new rules are directly generated from misclassified or rejected training patterns (for details, see Appendix C). Other new rules are generated by genetic operations from existing rules in the current population. Of course, when all the training patterns are correctly classified, all new rules are generated by the genetic operations in the Michigan-style algorithm.

Our objectives are not only to maximize the classification performance of fuzzy rule-based systems but also to minimize their size. Thus the number of fuzzy rules in each rule set should be adjustable. For this purpose, the string length in our hybrid algorithm is not constant. The string length is adjusted by a crossover operation, which generates a new string whose length is different from its parent strings. We use a kind of one-point crossover with different cutoff points illustrated in Fig. 2. In this crossover operation, one of the two offspring is randomly chosen as a new string.

5.3. Hybrid algorithm

Our hybrid algorithm can be written as follows in the framework of the three-objective genetic algorithm in Section 3.

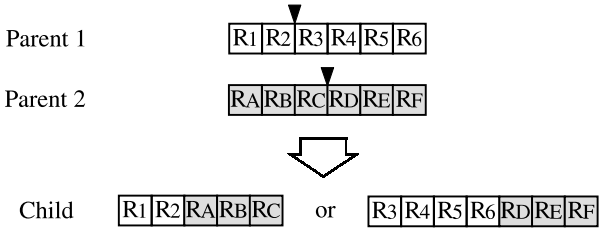

Fig. 2. One-point crossover with different cutoff points.

Hybrid fuzzy GBML algorithm

Step 1 *Initialization*. Generate an initial population consisting of N_{set} rule sets by the direct rule generation method.
Step 2 *Evaluation*. Calculate the values of the three objectives for each rule set in the current population. Then update the tentative pool of non-dominated rule sets.
Step 3 *Selection*. Select $(N_{set} - N_{elite})$ pairs of rule sets from the current population as in the three-objective genetic algorithm.
Step 4 *Crossover and mutation*. Generate a new rule set from each pair of selected rule sets using the one-point crossover in Fig. 2. The crossover operation is applied to each pair with a prespecified crossover probability. When the crossover operation is not applied, one of the parent strings is randomly chosen as an offspring. By the crossover operation, $(N_{set} - N_{elite})$ rule sets are generated. Then partially modify each of the generated rule sets using the Michigan-style algorithm in Appendix C. This algorithm is applied to each rule set with a prespecified probability.
Step 5 *Elitist strategy*. Randomly select N_{elite} non-dominated rule sets from their tentative pool, and add the selected elite rule sets to the generated $(N_{set} - N_{elite})$ rule sets in Step 4.
Step 6 *Termination test*. If a prespecified stopping condition is satisfied, end the algorithm. Otherwise, return to Step 2.

6. Computer simulations

6.1. Data sets and conditions of computer simulations

We applied the rule selection method and the hybrid algorithm to commonly used data sets in the literature: iris data, wine data, and glass data. All the data sets are available from the UC Irvine machine learning database.

Since we had no domain knowledge on each data set, we used the five linguistic values with the triangular membership functions in Fig. 1 for every

attribute of each data set. Our two algorithms (i.e., the rule selection method and the hybrid algorithm) were executed under the framework of the three-objective genetic algorithm in Section 3 using the following parameter values. The population size was 50 rule sets, the crossover probability was 0.9, the number of elite solutions was five, and each algorithm was terminated at the 1000th generation.

For the rule selection method in Section 4, we used biased mutation probabilities: 0.001 for the mutation from $s_j = 0$ to 1 and 0.1 for the mutation from $s_j = 1$ to 0. Randomly generated 50 initial rule sets were evolved in the rule selection method. On the other hand, a single iteration of the Michigan-style algorithm in Appendix C was applied to each rule set as a kind of mutation with the probability 0.9 in our hybrid algorithm. In the Michigan-style part, the crossover probability was 0.9, and the mutation probability was 0.1. Our hybrid algorithm searched for non-dominated rule sets by the evolution from initial rule sets with 20 fuzzy rules.

Each algorithm was applied to each data set 20 times. A set of non-dominated solutions stored in their tentative pool was obtained as final solutions of each trial when it was terminated at the 1000th generation. Those solutions are non-dominated among rule sets examined in the execution of each trial with respect to our three objectives: the number of correctly classified training patterns, the number of fuzzy rules, and the total number of antecedent conditions. From 20 trials, we obtained 20 sets of non-dominated solutions. For concisely summarizing simulation results, we merged them into a single solution set and compared solutions with each other. In such comparison, some solutions were dominated by other solutions obtained from different trials. All solutions that were dominated by other solutions from different trials were removed from the enlarged solution set. The refined solution set is reported as simulation results by each algorithm for each data set in this section. Since the classification performance is measured by the number of correctly classified training patterns in our three-objective optimization problem, we report the value of this objective of each non-dominated solution together with the other two objective values. All the available training data were used in our computer simulations and the classification performance on those training data is reported in this section. For the tradeoff between the generalization ability of fuzzy rule-based systems and the number of fuzzy rules, see our former study on two-objective rule selection [19] where the classification performance on test data was evaluated by the leaving-one-out (LV-1) procedure and the 10-fold cross-validation (10-CV) procedure in computer simulations.

6.2. Simulation results on iris data

The iris data set is a three-class pattern classification problem with four attributes and 150 patterns. We use the iris data set for illustrating three-

Table 1
Non-dominated rule sets obtained by the rule selection for iris data

Number of rules	0	1	2	3	4	5
Total length	0	1	2	3	4	7
Average length	0	1.00	1.00	1.00	1.00	1.40
Number of patterns	0	50	100	142	146	147
Rate (%)	0	33.3	66.7	94.7	97.3	98.0

objective rule selection while it is not actually a high-dimensional pattern classification problem. Fuzzy rules of the following type are used for the iris data set with four attributes:

$$\text{Rule } R_j : \text{If } x_1 \text{ is } A_{j1} \text{ and } \ldots \text{ and } x_4 \text{ is } A_{j4} \text{ then Class } C_j \text{ with } CF_j. \tag{14}$$

Since the iris data set includes only four attributes, we can examine all the $(5+1)^4 = 1296$ combinations of antecedent linguistic values for generating candidate fuzzy rules. By examining those combinations, we generated 587 fuzzy rules from the given 150 training patterns. Some fuzzy rules could not be generated because no training patterns are compatible with those rules. All the generated 587 fuzzy rules were used as candidate rules. In our rule selection method, each rule set was represented by a binary string of the length 587. For obtaining non-dominated rule sets, we applied the three-objective rule selection method to the 587 candidate rules 20 times using different initial populations. From the 20 trials, we found six non-dominated rule sets in Table 1. In this table, the three objectives (i.e., the number of correctly classified training patterns $f_1(S)$, the number of fuzzy rules $f_2(S)$, and the total number of antecedent conditions $f_3(S)$) are shown in the rows labeled as "number of patterns", "number of rules", and "total length", respectively. As mentioned in Section 4, the length of a fuzzy rule means the number of its antecedent conditions. Thus the third row labeled as "average length" shows the average number of antecedent conditions of each fuzzy rule. The last row labeled as "Rate (%)" shows the classification rate on training patterns. Since the iris data set is a three-class pattern classification problem, at least three fuzzy rules are necessary for designing fuzzy rule-based systems with high classification ability. In this sense, Table 1 includes three rule sets that are not practically useful. In Table 1, we can observe a tradeoff between the classification performance and the size of rule sets.

6.3. Simulation results on wine data

The wine data set is a three-class pattern classification problem with 13 attributes and 178 patterns. It is impractical to generate candidate rules by

examining all the $(5+1)^{13}$ combinations of antecedent linguistic values. Candidate rules were generated by examining only short fuzzy rules of the length 2 or less. Using this prescreening procedure, we generated 1834 candidate rules. The three-objective rule selection method was applied to the 1834 rules 20 times using different initial populations. We obtained 17 non-dominated rule sets from the 20 trials. In Table 2, we show 12 non-dominated rule sets with high classification rates. The other non-dominated rule sets have low classification rates because their size is too small. Simulation results in Table 2 show that compact rule sets with a small number of short fuzzy rules were found by the rule selection method. For example, we can design a fuzzy rule-based system with a 91.6% classification rate using the three fuzzy rules in (2)–(4) of Section 1. In Table 2, we can observe a tradeoff between the classification performance and the size of rule sets.

When good rule sets are obtained by the rule selection method, we do not have to use the hybrid fuzzy GBML algorithm. In practice, our hybrid fuzzy GBML algorithm may be employed only when the rule selection method does not work well. Our hybrid algorithm, however, was applied to the wine data set just for demonstrating its search ability. In the application of our hybrid algorithm to the wine data set, each fuzzy rule R_j is represented by its 13 antecedent linguistic values as $R_j = A_{j1}A_{j2}\ldots A_{j13}$. A rule set is represented by a concatenated string where each substring of the length 13 represents a fuzzy rule. Since no prescreening procedure of fuzzy rules is used, the search space in our hybrid algorithm consists of $(5+1)^{13}$ combinations of antecedent linguistic values.

From 20 trials of our hybrid algorithms, we obtained 18 non-dominated rule sets. Among them, 13 rule sets have more than 90% classification rates. We compared those 13 rule sets with the 12 rule sets in Table 2. Seven out of the 13 rule sets by the hybrid algorithm have the same objective values (e.g., (3, 3, 161) and (3, 4, 163)) as those in Table 2 by the rule selection method. The other six non-dominated rule sets are shown in Table 3. One rule set in Table 3 with the

Table 2
Non-dominated rule sets obtained by the rule selection for wine data

Number of rules	3	3	3	4	4	4
Total length	3	4	5	4	5	6
Average length	1.00	1.33	1.67	1.00	1.25	1.50
Number of patterns	161	163	164	169	170	171
Rate (%)	90.4	91.6	92.1	94.9	95.5	96.1
Number of rules	5	5	5	6	7	9
Total length	5	6	7	9	11	17
Average length	1.00	1.20	1.40	1.50	1.57	1.89
Number of patterns	172	173	175	176	177	178
Rate (%)	96.6	97.2	98.3	98.9	99.4	100

Table 3
Non-dominated rule sets obtained by the hybrid algorithm for wine data

Number of rules	3	5	6	6	7	8
Total length	5	7	6	10	8	15
Average length	1.67	1.40	1.00	1.67	1.14	1.88
Number of patterns	165	174	174	175	176	177
Rate (%)	92.7	97.8	97.8	98.3	98.9	99.4

This table does not include non-dominated rule sets that have the same objective values as those in Table 2.

three objective values (3, 5, 165) dominates the corresponding rule set (3, 5, 164) in Table 2. Two rule sets (6, 6, 174) and (7, 8, 176) in Table 3 are not dominated by any rule sets in Table 2. Each of the other three rule sets in Table 3 is dominated by at least one rule set in Table 2. From this comparison between Table 2 and the 13 non-dominated rule sets by the hybrid algorithm, we can conclude that our two algorithms found non-dominated rule sets of almost the same quality. If we take into account the huge search space in the hybrid algorithm, we can see that it has high search ability to find good rule sets.

Our two algorithms use antecedent linguistic values with fixed membership functions. Thus the quality of obtained non-dominated solutions strongly depends on a fuzzy partition. Since our aim is to generate linguistically interpretable fuzzy rules from numerical data, we assume that linguistic values are given. We do not modify their membership functions. For demonstrating the dependency of the quality of non-dominated rule sets on the choice of a fuzzy partition, we applied the rule selection method to the wine data using three linguistic values and don't care. That is, we assumed that each of the 13 attributes of the wine data was uniformly partitioned into three triangular membership functions in a similar manner to Fig. 1. We obtained 14 non-dominated rule sets from 20 trials. Seven rule sets with high classification rates are shown in Table 4. From the comparison between Tables 2 and 4, we can see that better rule sets with higher classification rates and fewer linguistic rules were obtained in the case of three linguistic values. It should be noted that our goal is not to maximize the classification performance but to generate linguistically interpretable linguistic rules. Thus the choice of a fuzzy partition should be done according to domain knowledge and intuition of users.

Table 4
Non-dominated rule sets obtained by the rule selection using three linguistic values

Number of rules	3	3	3	4	4	5	6
Total length	3	4	6	5	6	8	9
Average length	1.00	1.33	2.00	1.25	1.50	1.60	1.50
Number of patterns	164	171	173	174	176	177	178
Rate (%)	92.1	96.1	97.2	97.8	98.9	99.4	100

6.4. Simulation results on glass data

In the previous computer simulations, we have already shown that the rule selection method can find a small number of short fuzzy rules with high classification performance for designing comprehensible fuzzy rule-based systems. In this subsection, we examine the rule selection method and the hybrid algorithm through computer simulations on glass data. The glass data set is a six-class pattern classification problem with nine attributes and 214 patterns. The glass data set is a difficult classification problem with large overlaps between different classes in the pattern space. So it may be difficult to design compact fuzzy rule-based systems with high classification performance by a small number of short fuzzy rules.

For applying the rule selection method to the glass data, we generated candidate fuzzy rules of the length 2 or less as in the case of the wine data. The number of the generated candidate rules was 740. The three-objective rule selection method was applied to the 740 candidate rules. This computer simulation was iterated 20 times using different initial populations. From the 20 trials, we obtained 23 non-dominated rule sets. Among them, nine rule sets are shown in Table 5. Fig. 3 shows 20 non-dominated rule sets with more than 50% classification rates. From Table 5 and Fig. 3, we can see that fuzzy rule-based

Table 5
Non-dominated rule sets obtained by the rule selection for glass data

Number of rules	3	4	4	4	5	6	8	10	10
Total length	3	4	5	6	6	9	14	17	18
Average length	1.00	1.00	1.25	1.50	1.20	1.50	1.75	1.70	1.80
Number of patterns	127	134	140	142	143	148	153	155	157
Rate (%)	59.3	62.6	65.4	66.4	66.8	69.2	71.5	72.4	73.4

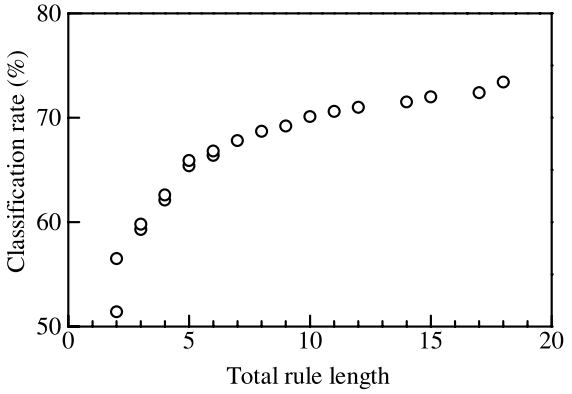

Fig. 3. Projection of non-dominated rule sets obtained by the rule selection on a two-dimensional objective space.

systems with high classification rates could not found. We also examined the case of the three linguistic values and don't care. We could not obtain high classification rates in this case, either.

Since we could not design good classification systems by fuzzy rules of the length 2 or less, we also generated fuzzy rules of the length 3 as candidate rules. The number of generated candidate rules of the length 3 was 5085. We applied the three-objective genetic algorithm to the rule selection problem with the 5825 candidate rules (i.e., the 740 rules of the length 2 or less and the 5085 rules of the length 3). Since the rule selection problem involved many candidate rules, large memory storage and long computation time were required. From 10 trials, we obtained 25 non-dominated solutions. Table 6 shows nine non-dominated rule sets for comparing them with Table 5. From the comparison between Tables 5 and 6, we can see that large rule sets with high classification rates were found by increasing the number of candidate rules. We can also see that the increase in the number of candidate rules led to slight deterioration in classification rates of compact rule sets (e.g., 59.3% by three fuzzy rules in Table 5 and 57.5% in Table 6). This is because the three-objective rule selection method could not efficiently find good rule sets in the enlarged search space. That is, the increase in the number of candidate rules from 740 to 5825 had a bad effect on the search for non-dominated rule sets. Fig. 4 shows 20

Table 6
Non-dominated rule sets obtained by rule selection with 5825 candidate rules

Number of rules	3	4	4	4	5	6	8	14	18
Total length	3	4	5	6	7	9	16	37	49
Average length	1.00	1.00	1.25	1.50	1.40	1.50	2.00	2.64	2.72
Number of patterns	123	134	136	142	143	148	155	162	164
Rate (%)	57.5	62.6	63.6	66.4	66.8	69.2	72.4	75.7	76.6

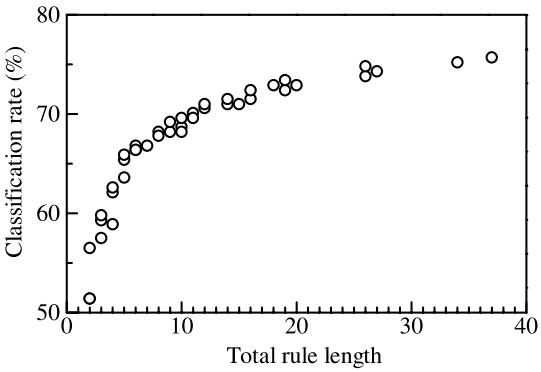

Fig. 4. Projection of non-dominated rule sets obtained by the rule selection with 5825 candidate rules on a two-dimensional objective space.

Table 7
Non-dominated rule sets obtained by the hybrid algorithm for glass data

Number of rules	3	4	4	4	5	6	8	8	10
Total length	3	4	5	6	9	10	14	19	20
Average length	1.00	1.00	1.25	1.50	1.80	1.67	1.75	2.38	2.00
Number of patterns	127	134	140	142	146	149	152	155	156
Rate (%)	59.3	62.6	65.4	66.4	68.2	69.6	71.0	72.4	72.9

non-dominated rule sets with more than 50% classification rates and the total rule length of less than 40. From this figure, we can see that high classification rates were realized by large rule sets.

We also applied the hybrid algorithm to the glass data set. From 20 trials, we obtained 25 non-dominated rule sets. Table 7 shows nine dominated rule sets for comparing them with Tables 5 and 6. From the comparison between these tables, we can see that non-dominated solutions in Table 7 by the hybrid algorithm are almost the same as those in Table 5. That is, there is no deterioration in the search ability due to the large search space, which was observed in Table 6. In Table 7, some rule sets include long fuzzy rules because there is no restriction on the rule length in the hybrid algorithm.

7. Concluding remarks

In this paper, we described how comprehensible fuzzy rule-based classification systems can be designed from numerical data. Emphasis was placed on the comprehensibility of fuzzy rule-based systems rather than their classification performance. Thus our fuzzy rules were generated using given linguistic values for each attribute without modifying their membership functions. That is, we assumed that a fuzzy partition had already been given. Even in this situation, rule generation is a difficult task in the case of high-dimensional pattern classification problems. First we formulated our task of designing comprehensible fuzzy rule-based systems as a three-objective optimization problem. The three objectives are to maximize the classification performance, to minimize the number of fuzzy rules, and to minimize the total length of fuzzy rules (i.e., the total number of antecedent conditions). Next we showed how the rule selection method in our former study could be extended to our three-objective optimization problem for high-dimensional pattern classification problems. Then we proposed a hybrid fuzzy GBML algorithm for finding non-dominated rule sets of our three-objective optimization problem. Finally we showed through computer simulations that a small number of linguistically interpretable fuzzy rules were found by our two algorithms for designing

comprehensible fuzzy rule-based classification systems for high-dimensional pattern classification problems.

Appendix A. Heuristic rule generation procedure

A heuristic rule generation procedure [18] for determining the consequent class and the certainty grade of the fuzzy rule R_j in (1) can be written as follows:

Step 1. Calculate the compatibility grade $\mu_j(\overrightarrow{x_p})$ of each training pattern $\overrightarrow{x_p} = (x_{p1}, \ldots, x_{pn})$ with the fuzzy rule R_j as

$$\mu_{R_j}(\overrightarrow{x_p}) = \mu_{j1}(x_{p1}) \times \cdots \times \mu_{jn}(x_{pn}), \quad p = 1, 2, \ldots, m, \tag{A.1}$$

where $\mu_{ji}(\cdot)$ is the membership function of the antecedent fuzzy set A_{ji}.

Step 2. For each class, calculate the sum of the compatibility grades of the training patterns with the fuzzy rule R_j:

$$\beta_{\text{Class } h}(R_j) = \sum_{\overrightarrow{x_p} \in \text{Class } h} \mu_{R_j}(\overrightarrow{x_p}), \quad h = 1, 2, \ldots, c. \tag{A.2}$$

Step 3. Find Class C_j that has the maximum value of $\beta_{\text{Class } h}(R_j)$:

$$\beta_{\text{Class } C_j} = \max\{\beta_{\text{Class } 1}(R_j), \ldots, \beta_{\text{Class } c}(R_j)\}. \tag{A.3}$$

If the consequent class (i.e., Class C_j) of the fuzzy rule R_j cannot be uniquely determined, we do not generate the fuzzy rule R_j. For example, if $\beta_{\text{Class } h}(R_j) = 1$ for all classes, we cannot determine C_j.

Step 4. Specify the certainty grade CF_j as follows:

$$CF_j = \{\beta_{\text{Class } C_j}(R_j) - \bar{\beta}\} \Big/ \sum_{h=1}^{c} \beta_{\text{Class } h}(R_j), \tag{A.4}$$

where

$$\bar{\beta} = \sum_{\substack{h=1 \\ h \neq C_j}}^{c} \beta_{\text{Class } h}(R_j)/(c-1). \tag{A.5}$$

This heuristic rule generation procedure has the following characteristic features:

1. The consequent class and the certainty grade of each fuzzy rule are locally determined by compatible training patterns with its antecedent part. The determination of a fuzzy rule is independent of the other rules.
2. The certainty grade of each fuzzy rule assumes a real number in the unit interval $[0, 1]$.
3. When all the compatible training patterns are from the same class, the certainty grade is its maximum value 1.

4. When all classes have almost the same sum of the compatibility grades in Step 2, the certainty grade is nearly equal to its minimum value 0.

Appendix B. Fuzzy reasoning method

Let S be the set of fuzzy rules in our fuzzy rule-based classification system. We use a fuzzy reasoning method based on a single winner rule [18]. The winner rule R_{j*} for a new pattern $\overrightarrow{x_p} = (x_{p1}, \ldots, x_{pn})$ is determined as follows:

$$\mu_{R_{j*}}(\overrightarrow{x_p}) \cdot CF_{j*} = \max\{\mu_{R_j}(\overrightarrow{x_p}) \cdot CF_j | R_j \in S\}. \tag{B.1}$$

The new pattern $\overrightarrow{x_p}$ is classified by the winner rule R_{j*}. That is, $\overrightarrow{x_p}$ is assigned to the consequent class of the winner rule. If multiple fuzzy rules with different consequent classes have the same maximum value in (B.1), the classification of the new pattern is rejected.

Appendix C. Fuzzy classifier system

In this paper, we use a single iteration of a Michigan-style fuzzy GBML algorithm (i.e., fuzzy classifier system [12,17]) as a mutation operation of our hybrid algorithm. Let S be a rule set to which our Michigan-style algorithm is applied. Our Michigan-style algorithm replaces $\alpha \cdot |S|$ fuzzy rules in S where $|S|$ is the number of fuzzy rules in S. In our computer simulations, α was specified as $\alpha = 0.3$. When $\alpha \cdot |S|$ is not an integer, the minimum integer that is not smaller than $\alpha \cdot |S|$ is used as the number of replaced fuzzy rules. Let m_{error} be the total number of misclassified and rejected training patterns by the current rule set S. When m_{error} is larger than $\alpha \cdot |S|/2$, half of new rules are generated directly from misclassified or rejected training patterns. On the other hand, when m_{error} is not larger than $\alpha \cdot |S|/2$, m_{error} new rules are generated directly from the m_{error} misclassified or rejected training patterns. In both cases, the other new rules are generated from existing rules in the current population (i.e., rule set S) by genetic operations. The outline of our Michigan-style algorithm used in the hybrid algorithm can be written as follows:

Step 1 *Evaluation*. Evaluate the fitness value of each fuzzy rule by classifying all the given training patterns using the rule set S. The fitness value of each fuzzy rule is defined by the number of training patterns correctly classified by that rule.

Step 2 *Generation of new fuzzy rules*. Generate $\alpha \cdot |S|$ fuzzy rules. At least half of new rules are generated from the existing rules in the current rule set S by genetic operations. First the roulette wheel scheme with the linear scaling is used for selecting pairs of parent rules. From each pair, two new rules are

generated by the uniform crossover. Then the mutation operation is applied to each antecedent fuzzy set of the generated rules. The crossover and mutation operations are used with prespecified probabilities. The other new rules are generated using the direct rule generation procedure in Section 5.2 from misclassified or rejected training patterns. All the newly generated fuzzy rules are added to the rule set S.

Step 3 *Replacement*. Evaluate the fitness value of each fuzzy rule in the enlarged current rule set S by classifying all the given training patterns using S. The worst $\alpha \cdot |S|$ fuzzy rules with the smallest fitness values are removed from S.

References

[1] S. Abe, M.-S. Lan, R. Thawonmas, Tuning of a fuzzy classifier derived from data, International Journal of Approximate Reasoning 14 (1996) 1–24.
[2] S. Abe, R. Thawonmas, A fuzzy classifier with ellipsoidal regions, IEEE Transactions on Fuzzy Systems 5 (1997) 358–368.
[3] L.B. Booker, D.E. Goldberg, J.H. Holland, Classifier systems and genetic algorithms, Artificial Intelligence 40 (1989) 235–282.
[4] O. Cordon, M.J. del Jesus, F. Herrera, Genetic learning of fuzzy rule-based classification systems cooperating with fuzzy reasoning methods, International Journal of Intelligent Systems 13 (1998) 1025–1053.
[5] O. Cordon, M.J. del Jesus, F. Herrera, A proposal on reasoning methods in fuzzy rule-based classification systems, International Journal of Approximate Reasoning 20 (1999) 21–45.
[6] M. Dash, H. Liu, Feature selection for classification, Intelligent Data Analysis, vol. 1, 1997, electronic journal: http://www-east.elsevier.com/ida/.
[7] R.C. Holte, Very simple classification rules perform well on most commonly used dataset, Machine Learning 11 (1993) 63–91.
[8] R. Holve, Rule generation for hierarchical fuzzy systems, in: Proceedings of 16th NAFIPS Annual Meeting, 1997, pp. 444–449.
[9] H. Ishibuchi, T. Murata, A multi-objective genetic local search algorithm and its application to flowshop scheduling, IEEE Transactions on Systems, Man, and Cybernetics – Part C: Applications and Reviews 28 (1998) 392–403.
[10] H. Ishibuchi, T. Murata, T. Nakashima, Linguistic rule extraction from numerical data for high-dimensional classification problems, International Journal of Advanced Computational Intelligence 3 (1999) 386–393.
[11] H. Ishibuchi, T. Murata, I.B. Turksen, Single-objective and two-objective genetic algorithms for selecting linguistic rules for pattern classification problems, Fuzzy Sets and Systems 89 (1997) 135–149.
[12] H. Ishibuchi, T. Nakashima, Improving the performance of fuzzy classifier systems for pattern classification problems with continuous attributes, IEEE Transactions on Industrial Electronics 46 (1999) 1057–1068.
[13] H. Ishibuchi, T. Nakashima, T. Kuroda, A hybrid fuzzy GBML algorithm for designing compact fuzzy rule-based classification systems, in: Proceedings of 9th IEEE International Conference on Fuzzy Systems, 2000, pp. 706–711.
[14] H. Ishibuchi, T. Nakashima, T. Kuroda, Minimizing the number of fuzzy rules by fuzzy genetics-based machine learning for pattern classification problems, in: Proceedings of the 8th

Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, 2000, pp. 96–103.
[15] H. Ishibuchi, T. Nakashima, T. Morisawa, Simple fuzzy rule-based classification systems perform well on commonly used real-world data sets, in: Proceedings of 16th Annual Meeting of the North American Fuzzy Information Processing Society, 1997, pp. 251–256.
[16] H. Ishibuchi, T. Nakashima, T. Murata, Genetic-algorithm-based approaches to the design of fuzzy systems for multi-dimensional pattern classification problems, in: Proceedings of 3rd IEEE International Conference on Evolutionary Computation, 1996, pp. 229–234.
[17] H. Ishibuchi, T. Nakashima, T. Murata, Performance evaluation of fuzzy classifier systems for multi-dimensional pattern classification problems, IEEE Transactions on Systems, Man, and Cybernetics – Part B: Cybernetics 29 (1999) 601–618.
[18] H. Ishibuchi, K. Nozaki, H. Tanaka, Distributed representation of fuzzy rules and its application to pattern classification, Fuzzy Sets and Systems 52 (1992) 21–32.
[19] H. Ishibuchi, T. Sotani, T. Murata, Tradeoff between the performance of fuzzy rule-based classification systems and the number of fuzzy if-then rules, in: Proceedings of 18th International Conference of the North American Fuzzy Information Processing Society – NAFIPS'99, 1999, pp. 125–129.
[20] Y. Jin, Fuzzy modeling of high-dimensional systems: complexity reduction and interpretability improvement, IEEE Transactions on Fuzzy Systems 8 (2000) 212–221.
[21] Y. Jin, W. von Seelen, B. Sendhoff, On generating FC3 fuzzy rule systems from data using evolution strategies, IEEE Transactions on Systems, Man and Cybernetics – Part B: Cybernetics 29 (1999) 829–845.
[22] R. Kohavi, G.H. John, Wrappers for feature subset selection, Artificial Intelligence 97 (1997) 273–324.
[23] B. Kosko, Fuzzy systems as universal approximators, in: Proceedings of 1st IEEE International Conference on Fuzzy Systems, 1992, pp. 1153–1162.
[24] L.I. Kuncheva, Fuzzy Classifier Design, Physica-Verlag, Heidelberg, 2000.
[25] H. Liu, H. Motoda, Feature Selection for Knowledge Discovery and Data Mining, Kluwer Academic Publishers, Boston, 1998.
[26] H. Liu, H. Motoda (Eds.), Feature Extraction, Construction and Selection: A Data Mining Perspective, Kluwer Academic Publishers, Boston, 1998.
[27] S. Mitra, R.K. De, S.K. Pal, Knowledge-based fuzzy MLP for classification and rule generation, IEEE Transactions on Neural Networks 8 (1997) 1338–1350.
[28] T. Murata, H. Ishibuchi, MOGA: Multi-objective genetic algorithms, in: Proceedings of 2nd IEEE International Conference on Evolutionary Computation, 1995, pp. 289–294.
[29] D. Nauck, R. Kruse, A neuro-fuzzy method to learn fuzzy classification rules from data, Fuzzy Sets and Systems 89 (1997) 277–288.
[30] K. Nozaki, H. Ishibuchi, H. Tanaka, Adaptive fuzzy rule-based classification systems, IEEE Transactions on Fuzzy Systems 4 (1996) 238–250.
[31] V. de Oliveira, Semantic constraints for membership function optimization, IEEE Transactions on Systems, Man, and Cybernetics – Part A: Systems and Humans 29 (1999) 128–138.
[32] W. Pedrycz, V. de Oliveira, Optimization of fuzzy models, IEEE Transactions on Systems, Man, and Cybernetics – Part B: Cybernetics 26 (1996) 627–637.
[33] M. Setnes, R. Babuska, U. Kaymak, H.R. van Nauta Lemke, Similarity measures in fuzzy rule base simplification, IEEE Transactions on Systems, Man, and Cybernetics – Part B: Cybernetics 28 (1998) 376–386.
[34] M. Setnes, R. Babuska, B. Verbruggen, Rule-based modeling: Precision and transparency, IEEE Transactions on Systems, Man, and Cybernetics – Part C: Applications and Reviews 28 (1998) 165–169.
[35] S.F. Smith, A learning system based on genetic algorithms, Ph.D. Dissertation, University of Pittsburgh, Pittsburgh, PA, 1980.

[36] D.A. van Veldhuizen, G.B. Lamont, Multiobjective evolutionary algorithms: Analyzing the state-of-the-art, Evolutionary Computation 8 (2000) 125–147.
[37] L.-X. Wang, Fuzzy systems are universal approximators, in: Proceedings of 1st IEEE International Conference on Fuzzy Systems, 1992, pp. 1163–1170.
[38] J. Yen, L. Wang, Simplifying fuzzy rule-based models using orthogonal transformation methods, IEEE Transactions on Systems, Man, and Cybernetics – Part B: Cybernetics 29 (1999) 13–24.
[39] J. Yen, L. Wang, W. Gillespie, Improving the interpretability of TSK fuzzy models by combining global learning and local learning, IEEE Transactions on Fuzzy Systems 6 (1998) 530–537.
[40] Y. Yuan, H. Zhuang, A genetic algorithm for generating fuzzy classification rules, Fuzzy Sets and Systems 84 (1996) 1–19.
[41] E. Zitzler, K. Deb, L. Thiele, Comparison of multiobjective evolutionary algorithms: Empirical results, Evolutionary Computation 8 (2000) 173–195.
[42] E. Zitzler, L. Thiele, Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach, IEEE Transactions on Evolutionary Computation 3 (1999) 257–271.