Multiobjective Optimization in Linguistic Rule Extraction from Numerical Data

Hisao Ishibuchi [1], Tomoharu Nakashima [1], and Tadahiko Murata [2]

[1] Department of Industrial Engineering, Osaka Prefecture University,
1-1 Gakuen-cho, Sakai, Osaka 599-8531, Japan
{hisaoi, nakashi}@ie.osakafu-u.ac.jp
[2] Department of Industrial and Information Systems Engineering,
Ashikaga Institute of Technology,
268 Omae-cho, Ashikaga 326-8558, Japan
murata@ashitech.ac.jp

Abstract. We formulate linguistic rule extraction as a three-objective combinatorial optimization problem. Three objectives are to maximize the performance of an extracted rule set, to minimize the number of extracted rules, and to minimize the total length of extracted rules. The second and third objectives are related to comprehensibility of the extracted rule set. We describe and compare two genetic-algorithm-based approaches for finding non-dominated rule sets with respect to the three objectives of our linguistic rule extraction problem. One approach is rule selection where a small number of linguistic rules are selected from prespecified candidate rules. The other is genetics-based machine learning where rule sets are evolved by generating new rules from existing ones using genetic operations.

1 Introduction

As multi-layer feedforward neural networks, fuzzy rule-based systems are universal approximators of nonlinear functions [20, 29]. Both of them have many application fields such as control, forecast, modeling and classification. One advantage of fuzzy rule-based systems over neural networks is the transparency of input-output relations realized by fuzzy rules. While neural networks are handled as a black-box, fuzzy rules are usually written in a linguistically interpretable form such as "If x_1 is *large* and x_2 is *small* then *y* is *large*" and "If x_1 is *medium* and x_2 is *large* then Class 2." We refer to linguistically interpretable fuzzy rules as linguistic rules. Approximation ability and interpretability make fuzzy rule-based systems a practically useful modeling tool.

Fuzzy rules are obtained from domain experts as linguistic knowledge. Since linguistic knowledge is incomplete and inaccurate, automated rule generation methods from numerical data have been proposed [27, 30]. Automated tuning methods of fuzzy rule-based systems have been also proposed for improving their

E. Zitzler et al. (Eds.): EMO 2001, LNCS 1993, pp. 588-602, 2001.

performance [1, 3, 5, 16, 19]. Some methods [3, 19] used genetic algorithms, and others [1, 5, 16] were similar to the learning of neural networks. See [21] for various techniques of fuzzy rule-based system design. Many of these rule generation and tuning methods tried to maximize the performance of fuzzy rule-based systems.

Recently, some researchers [17, 18, 22-26, 31, 32] tried to improve interpretability of fuzzy rule-based systems. For example, interpretability of membership functions was discussed in [22, 23]. The number of fuzzy rules was decreased in [24-26, 31]. Jin [17] pointed out the following four factors closely related to interpretability of fuzzy rule-based systems.

(a) Distinguishability of a fuzzy partition. Membership functions should be clearly distinguishable from each other so that a linguistic term can be assigned to each membership function.
(b) Consistency of fuzzy rules. Fuzzy rules in a fuzzy rule-based system should not be strongly contradictory to each other.
(c) The number of fuzzy rules. It is easy to examine a small number of fuzzy rules while the examination of many rules is a cumbersome task.
(d) The number of conditions in the antecedent part (i.e., if-part). It is not easy to understand a fuzzy rule with many antecedent conditions.

Among these four factors, distinguishability was included in a cost function in regularized learning of Jin [17].

In this paper, we formulate linguistic rule extraction as a combinatorial optimization problem with three objectives: to maximize the performance of an extracted rule set, to minimize the number of extracted rules, and to minimize the total length of extracted rules. That is, the last two factors (c) and (d) are considered. Since we use prespecified linguistic terms with fixed membership functions, we do not have to consider the distinguishability of a fuzzy partition. Inconsistency of fuzzy rules is resolved by assigning a certainty grade to each linguistic rule. For an *n*-dimensional pattern classification problem, we try to extract linguistic rules of the following form:

$$\text{Rule } R_j: \text{If } x_1 \text{ is } A_{j1} \text{ and } \ldots \text{ and } x_n \text{ is } A_{jn} \text{ then Class } C_j \text{ with } CF_j, \quad (1)$$

where R_j is the label of the *j*-th linguistic rule, $\mathbf{x} = (x_1, \ldots, x_n)$ is an *n*-dimensional pattern vector, A_{ji} is a linguistic value (e.g., *small* and *large*) for the *i*-th attribute, C_j is a consequent class, and CF_j is a certainty grade in the unit interval [0, 1].

For finding non-dominated rule sets of our three-objective linguistic rule selection problem for pattern classification, we examine two schemes: a rule selection method by genetic algorithms and a genetics-based machine learning (GBML) method. These two schemes are compared with each other through computer simulations on a high-dimensional pattern classification problem. We also discuss linguistic rule extraction from numerical data for function approximation, which is formulated as a three-objective problem as in the case of pattern classification.

2 Formulation of Linguistic Rule Extraction

In this section, we formulate linguistic rule extraction from numerical data as a three-objective combinatorial optimization problem for pattern classification. Our basic idea is to simultaneously maximize classification ability and interpretability of extracted rule sets.

2.1 Assumptions

We assume that *m* training patterns (i.e., labeled patterns) are given as numerical data for an *n*-dimensional *c*-class pattern classification problem. We denote those training patterns as $\mathbf{x} = (x_{p1},...,x_{pn})$, $p = 1,2,...,m$. For simplicity of explanation, each attribute value x_{pi} is assumed to be a real number in the unit interval [0, 1], i.e., $x_{pi} \in [0, 1]$. This means that the pattern space of our pattern classification problem is the *n*-dimensional unit hypercube $[0, 1]^n$. In computer simulations of this paper, all attribute values are normalized into real numbers in the unit interval [0, 1].

We also assume that a set of linguistic values is given for describing each attribute. That is, we assume that a fuzzy partition of the pattern space $[0, 1]^n$ is given. For simplicity of explanation, we use five linguistic values in Fig. 1 for all the *n* attributes. Of course, our approaches described in this paper are applicable to more general cases where a different set of linguistic values is given to each attribute. In such a general case, membership functions are not necessary to be triangular. They are specified according to domain knowledge and intuition of human experts.

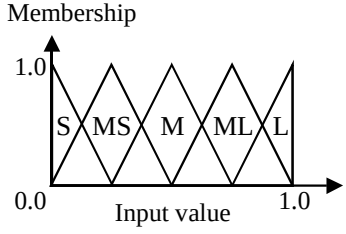

Fig. 1. Membership functions of five linguistic values (S: *small*, MS: *medium small*, M: *medium*, ML: *medium large*, and L: *large*).

As Jin [17] pointed out, it is not easy to understand fuzzy rules with many antecedent conditions. Thus we use "DC: *don't care*" as an additional linguistic value in the antecedent part of our linguistic rule in (1). Linguistic rules with many "*don't care*" conditions can be concisely written even for high-dimensional pattern classification problems. The following are some examples of such linguistic rules for a 13-dimensional wine classification problem used in Section 5.

R_1: If x_7 is *medium* and x_{11} is *medium* then Class 1 with 0.56. (2)

R_2: If x_{10} is *small* then Class 2 with 0.94. (3)

R_3: If x_7 is *small* then Class 3 with 0.85. (4)

These linguistic rules are very simple and easily understood. In the above linguistic rules, "*don't care*" conditions are omitted.

When we use the five linguistic values in Fig. 1 and "DC: *don't care*" as antecedent linguistic values in our linguistic rule in (1), we have $(5+1)^n$ combinations of antecedent linguistic values as follows:

$$\text{Rule } R_j: \text{If } x_1 \text{ is } \left\{\begin{matrix}\text{S}\\ \text{MS}\\ \text{M}\\ \text{ML}\\ \text{L}\\ \text{DC}\end{matrix}\right\} \text{ and ... and } x_n \text{ is } \left\{\begin{matrix}\text{S}\\ \text{MS}\\ \text{M}\\ \text{ML}\\ \text{L}\\ \text{DC}\end{matrix}\right\} \text{ then Class } C_j \text{ with } CF_j. \quad (5)$$

As shown in Appendix, the consequent class C_j and the certainty grade CF_j of each linguistic rule can be easily specified from training patterns when its antecedent conditions are given. Thus our linguistic rule extraction problem can be viewed as finding a small number of combinations of antecedent linguistic values among the above $(5+1)^n$ combinations. The total number of possible rule sets is 2^N where $N=(5+1)^n$. It is not easy to examine all the possible rule sets even for a three-dimensional pattern classification problem. In this case, the total number of possible rule sets is $2^{6\times6\times6} \cong 1.05\times10^{65}$. In the case of high-dimensional problems, we can examine only a tiny portion of possible rule sets because the search space is huge.

2.2 Three-Objective Combinatorial Optimization Problem

Let S_{ALL} be the set of $(5+1)^n$ linguistic rules for our *n*-dimensional pattern classification problem. They correspond to the $(5+1)^n$ combinations of the five linguistic values in Fig. 1 and "*don't care*". Our linguistic rule extraction problem is to find a small number of simple linguistic rules with high classification ability from the rule set S_{ALL}, i.e., to find a compact and high-performance rule set. Because there is a tradeoff between compactness and performance, we try to find non-dominated rule sets with respect to conflicting criteria.

We measure the classification performance of a rule set S ($S \subset S_{\text{ALL}}$) by the number of correctly classified training patterns. Compactness of *S* is measured by two criteria: the number of linguistic rules in *S* and the total number of antecedent conditions in *S*. Of course, "*don't care*" conditions are not counted among antecedent conditions. The number of antecedent conditions in a linguistic rule is referred to as its length. Thus the total number of antecedent conditions is the same as the total

length of linguistic rules. Based on these discussions, our linguistic rule extraction problem is formulated as follows:

$$\text{Maximize } f_1(S), \text{ minimize } f_2(S), \text{ and minimize } f_3(S), \tag{6}$$

where

$f_1(S)$: The number of correctly classified training patterns by *S*,

$f_2(S)$: The number of linguistic rules in *S*,

$f_3(S)$: The total length of linguistic rules in *S*.

For example, when a rule set *S* consists of the three linguistic rules R_1, R_2 and R_3 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 *m* training patterns by the rule set *S*. In Appendix, we show how each pattern is classified by linguistic rules (see [2] for various fuzzy reasoning methods for pattern classification).

The third objective $f_3(S)$ is not the average length of extracted rules but the total length. Let $f_{3*}(S)$ be the average length. 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. Let us construct another rule set S^+ by adding a linguistic rule R_4 of the length one to *S*. For the new rule set S^+ with $R_1 \sim R_4$, $f_{3*}(S^+)$ is calculated as $f_{3*}(S^+) = 1.25$. That is, the average length is improved by adding R_4 to the rule set *S* while the complexity of the rule set is increased. Even if the added linguistic rule R_4 does not improve the classification performance of the rule set (i.e., $f_1(S) = f_1(S^+)$), the new rule set S^+ is not dominated by the rule set *S* when we use the average length as the third objective. This simple example shows that the average length is not an appropriate objective for measuring the simplicity of extracted linguistic rules in the framework of multi-objective optimization. Thus we use the total length as the third objective $f_3(S)$. In [7], the average length was used for rule selection. Since three objectives were combined into a scalar fitness function in [7] for obtaining a single optimal rule set, the above-mentioned difficulty of the average length can be ignored. The difficulty of the average length is crucial only when this objective is used in the framework of multi-objective optimization for obtaining non-dominated rule sets.

2.3 Simple Numerical Example

Let us consider a simple numerical example in Fig. 2 (a) where 121 training patterns are given in the unit square $[0, 1] \times [0, 1]$. If we use a standard grid-type fuzzy partition in Fig. 2 (b), we can generate 5×5 linguistic rules with no "*don't care*" conditions. All the given training patterns are correctly classified by those 25 linguistic rules. On the other hand, a much simpler rule set can be extracted if we consider linguistic rules with "*don't care*" conditions.

As shown in Subsection 2.1, the total number of possible combinations of antecedent linguistic values is 36 for the two-dimensional pattern classification

problem when we use "*don't care*" as an additional linguistic value. By examining subsets of those 36 linguistic rules, we can find that the following three linguistic rules correctly classify all the given training patterns:

R_A : If x_1 is *medium* then Class 2 with 0.75. (7)

R_B : If x_2 is *large* then Class 2 with 0.84 . (8)

R_C : (x_1, x_2) is Class 1 with 0.19. (9)

The last linguistic rule R_C has no linguistic condition (i.e., it has two "*don't care*" conditions). Since R_C has a small certainty grade (i.e., 0.19), this rule is used for pattern classification only when the other two linguistic rules R_A and R_B are not applicable. In this manner, inconsistency is resolved through certainty grades.

All the non-dominated rule sets of our linguistic rule extraction problem for this numerical example are shown in Table 1. Since the numerical example in Fig. 2 is very simple, we can find all non-dominated solutions by examining all rule sets with one, two or three linguistic rules. Usually we can not find all non-dominated solutions by such an enumeration method especially for high-dimensional pattern classification problems. In the rest of this paper, we explain how genetic algorithms can be used for finding non-dominated solutions of our three-objective linguistic rule extraction problem.

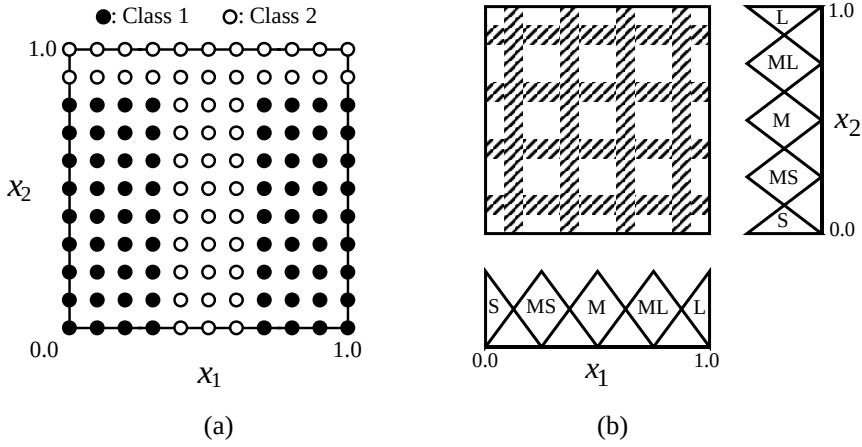

Fig. 2. A numerical example and a grid-type fuzzy partition.

Table 1. All the non-dominated solutions for the numerical example in Fig. 1.

Rule set: S	# of patterns: $f_1(S)$	# of rules: $f_2(S)$	Total length: $f_3(S)$
{}	0 (0%)	0	0
{ R_C }	72 (60%)	1	0
{ R_A, R_C }	105 (87%)	2	1
{ R_A, R_B, R_C }	121 (100%)	3	2

3 Rule Selection

3.1 Basic Idea of Rule Selection

We have already proposed a GA-based rule selection method [7, 15] where multiple objectives were combined into a scalar fitness function for applying standard single-objective genetic algorithms. We have also proposed two-objective genetic algorithms for finding non-dominated rule sets with respect to the classification performance and the number of linguistic rules [8]. In this paper, the total rule length is added to the two-objective rule selection method in [8].

It is not difficult to extend our former two-objective rule selection method to the case of three objectives. Let N be the number of linguistic rules that can be generated from given training patterns. Those N linguistic rules are used as candidate rules in rule selection. In low-dimensional pattern classification problems, all the $(5+1)^n$ linguistic rules in (5) are considered as candidate rules. All linguistic rules, however, are not always generated (e.g., when training patterns are not evenly distributed in the entire pattern space). In high-dimensional pattern classification problems, the number of candidate rules should be much smaller than $(5+1)^n$ because the string length is the same as the number of candidate rules in our rule selection method.

Any subset S of N candidate rules can be represented by a binary string of the length N as $S = s_1 s_2 \cdots s_N$. The inclusion and exclusion of the j-th candidate rule are represented by $s_j = 1$ and $s_j = 0$, respectively. Since every feasible solution of our problem is represented by a binary string, we can use various multiobjective genetic algorithms [28, 33, 34] for finding its non-dominated solutions.

3.2 Candidate Rule Prescreening

In high-dimensional pattern classification problems, we can not handle all the $(5+1)^n$ linguistic rules in (5) as candidate rules. Thus a prescreening procedure of candidate rules is necessary for applying our rule selection method to high-dimensional problems. A simple trick is to examine only short linguistic rules with a few antecedent linguistic conditions [7]. This trick, which has also a good effect on the third objective of our rule extraction problem, can significantly decrease the number of candidate rules. Even when the total number of linguistic rules is huge, the number of short rules is not so large. For example, the number of linguistic rules of the length one with a single antecedent condition is calculated as ${}_{13}C_1 \times 5 = 65$ for a 13-dimensional problem such as wine data. In our computer simulation on the wine data, we examine one rule of the length zero, 65 rules of the length one, and ${}_{13}C_2 \times 5 \times 5 = 1950$ rules of the length two for generating candidate rules in our rule selection method. In [7], the certainty grade of each linguistic rule was also used in addition to the length for prescreening candidate rules.

3.3 Domain-Specific Heuristic Procedures

The performance of genetic algorithms for rule selection can be improved by incorporating domain-specific heuristic procedures. One procedure is for eliminating unnecessary rules. As shown in Appendix, we use a fuzzy reasoning method based on a single winner rule in the classification phase. This means that each pattern is classified by a single winner rule in a rule set. If a linguistic rule is not used as a winner rule for any training pattern, that rule can be removed from the rule set with no deterioration of its classification performance. This elimination improves the second and third objectives of our linguistic rule extraction problem. Our rule elimination procedure removes such a linguistic rule before each rule set is evaluated in three-objective genetic algorithms.

Another trick is to bias the mutation probability. Even when we use an appropriate prescreening procedure, usually the number of selected rules is much smaller than that of candidate rules. That is, binary strings should consist of a small number of 1's and a large number of 0's. The standard mutation tends to increase the number of 1's when binary strings have much more 0's than 1's. For efficiently searching for binary strings with a small number of 1's, we use biased mutation probabilities where the mutation probability from 1 to 0 is much larger than that from 0 to 1.

4 Genetics-Based Machine Learning

4.1 Basic Idea of Genetics-Based Machine Learning

The quality of non-dominated rule sets obtained by rule selection strongly depends on the choice of a prescreening procedure. While some studies [4, 12] showed high performance of short rules with only a few antecedent conditions, this is not always the case. Some pattern classification problems may need long rules as well as short rules. In this case, the search among $(5+1)^n$ linguistic rules is necessary for finding good rule sets. Genetics-based machine learning (GBML) algorithms are promising tools for finding non-dominated rule sets in the huge search space.

We have already proposed Michigan-style GBML algorithms for generating linguistic rules for high-dimensional pattern classification problems [9, 14]. In our Michigan-style algorithm, a single linguistic rule was coded as a string. A population with a fixed number of linguistic rules was evolved by genetic operations for finding good linguistic rules. Since our objectives in this paper are not only classification performance but also the number of linguistic rules and the total rule length, the number of linguistic rules should not be fixed. It is, however, difficult to directly optimize the number of linguistic rules in the framework of Michigan-style algorithms because a fitness value is assigned to each linguistic rule. Thus we use a Pittsburgh-style algorithm with variable sting length for finding non-dominated rule sets.

4.2 Genetic Operations

As in our Michigan-style algorithms in [9, 14], each linguistic rule is coded by its antecedent linguistic values as "$A_{j1}A_{j2}\cdots A_{jn}$". Since the consequent class and the certainty grade are easily specified by training patterns, they are not coded as a part of the string. A rule set is denoted by a concatenated string. Each substring of the concatenated string corresponds to a single linguistic rule.

A new rule set (i.e., a new string) is generated by crossover and mutation. In our computer simulation, we use a kind of one-point crossover shown in Fig. 3, which changes the number of linguistic rules in each rule set. This crossover operation randomly selects a different cutoff point for each parent to form an offspring. A mutation operation randomly replaces each element (i.e., each antecedent linguistic value) of the string with another linguistic value. Elimination of existing rules and addition of new rules can be also used as mutation operations. Such mutation operations change the number of linguistic rules in each string.

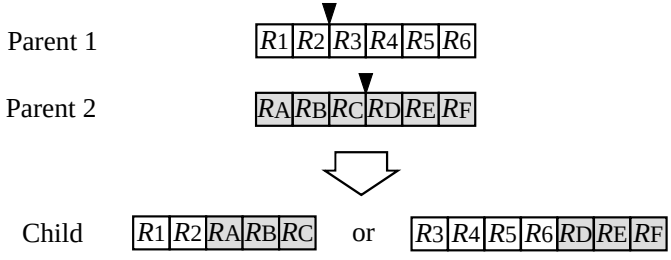

Fig. 3 Crossover operation.

4.3 Hybridization with Michigan-Style Algorithm

The search ability of Pittsburgh-style algorithms to find good linguistic rules is somewhat inferior to that of Michigan-style algorithms when they are applied to high-dimensional pattern classification problems [11, 13]. Michigan-style algorithms, however, can not directly optimize rule sets because a fitness value is assigned to each linguistic rule (not to each rule set). On the other hand, Pittsburgh approach can directly optimize rule sets. A natural idea for utilizing advantages of these two kinds of GBML approaches is to combine them into a single hybrid algorithm [10, 11]. A Michigan-style algorithm, which is used as a mutation operation in our Pittsburgh-style algorithm, partially modifies each rule set by generating new rules. By this hybridization, search ability of our Pittsburgh-style algorithm to efficiently find good linguistic rules is significantly improved. In our computer simulation, a single iteration of a Michigan-style algorithm was applied with a prespecified probability to every rule set generated by the genetic operations of our Pittsburgh-style algorithm.

5 Computer Simulations

5.1 Test Problem and Simulation Conditions

We applied the rule selection method and the hybrid GBML algorithm to wine data (available from UC Irvine Database: http://kdd.ics.uci.edu/) for finding non-dominated rule sets of our linguistic rule extraction problem. These two algorithms were implemented in the framework of three-objective genetic algorithms. The three objectives were combined into the following fitness function, which was used in a roulette wheel selection with a linear scaling.

$$fitness(S) = w_1 \cdot f_1(S) - w_2 \cdot f_2(S) - w_3 \cdot f_3(S). \quad (10)$$

For finding various non-dominated solutions, weights w_1, w_2 and w_3 were not fixed but randomly updated whenever a pair of parent strings were selected as in our two-objective genetic algorithm in [8]. Non-dominated solutions were separately stored from the current population. Some of the stored non-dominated solutions were added to the current population for maintaining its diversity and quality.

Both the rule selection method and the hybrid GBML algorithm were executed under the following parameter specifications:

Population size: 50 rule sets.
Stopping conditions: 1000 generations.

That is, a population of 50 rule sets was evolved until the 1000th generation in both algorithms. We applied each algorithm to wine data 20 times. In the following subsections, we report non-dominated solutions obtained from those 20 trials.

5.2 Simulation Results by Rule Selection

Since wine data involve 13 attributes, the number of possible combinations of antecedent linguistic values is $(5+1)^{13} \cong 1.3 \times 10^{10}$ (i.e., about 13 billion). It is impossible to apply our rule selection method to candidate rules generated from all those combinations. We examined only short linguistic rules of the length two or less. By this prescreening procedure, 1834 linguistic rules were generated as candidate rules. Thus each rule set was represented by a binary string of the length 1834. The task of our rule selection method is to find non-dominated rule sets from those candidate rules. From 20 trials, we obtained 17 non-dominated rule sets. Due to the space limitation, we show 12 rule sets with high classification rates. Too small rule sets are not shown in this table (e.g., those with only two rules). From Table 2, we can see that our rule selection method found various non-dominated rule sets. Some are very compacts, and others have high classification rates. We can observe a tradeoff between the classification performance and the compactness of rule sets.

Table 2. Non-dominated rule sets obtained by the GA-based rule selection method.

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

5.3 Simulation Results by Genetics-Based Machine Learning

The search space in the hybrid GBML algorithm for wine data consists of $(5+1)^{13}$ linguistic rules. This is much larger than that of the rule selection method where non-dominated rule sets were selected from 1834 candidate rules. From 20 trials, we obtained 18 non-dominated rule sets. Twelve rule sets with high classification rates are shown in Table 3. From the comparison between Table 2 and Table 3, we can see that similar results were obtained by the two algorithms while the search space in the hybrid GBML algorithm was terribly large. Some non-dominate rule sets obtained by the hybrid GBML algorithm include linguistic rules of the length three, which were not considered in the rule selection method.

Table 3. Non-dominated rule sets obtained by the hybrid GBML algorithm.

# of rules	3	3	4	4	4	5	5	5	6	6	7	8
Total length	4	5	4	5	6	5	6	7	6	10	8	15
Average length	1.33	1.67	1.00	1.25	1.50	1.00	1.20	1.40	1.00	1.67	1.14	1.88
# of patterns	163	165	169	170	171	172	173	174	174	175	176	177
Classification rate	91.6	92.7	94.9	95.5	96.1	96.6	97.2	97.8	97.8	98.3	98.9	99.4

6 Linguistic Rule Extraction for Function Approximation

Our three-objective linguistic rule extraction problem can be easily modified for the application to function approximation. For an *n*-input and single-output function approximation problem, we use linguistic rules of the following type:

$$\text{Rule } R_j: \text{If } x_1 \text{ is } A_{j1} \text{ and } ... \text{ and } x_n \text{ is } A_{jn} \text{ then } y \text{ is } B_j, \tag{11}$$

where $\mathbf{x} = (x_1,...,x_n)$ is an input vector, A_{ji} is an antecedent linguistic value, y is an output variable, and B_j is a consequent linguistic value. Since "*don't care*" is used only in the antecedent part, the total number of possible combinations of antecedent and consequent linguistic values is $(5+1)^n \times 5$. In the rule selection scheme, some of those combinations are examined for generating candidate rules. In

the GBML scheme, each linguistic rule is denoted by its antecedent and consequent linguistic values as "$A_{j1} \cdots A_{jn} B_j$". A rule set is denoted by a concatenated string.

Let us assume that m input-output pairs $(\mathbf{x}_p, y_p)$, $p = 1,2,...,m$ are given as numerical data where $\mathbf{x}_p = (x_{p1},...,x_{pn})$ is an input vector and y_p is the corresponding output value. The performance of a rule set S can be measured by some cost function $f_1(S)$ such as the total squared error or the total absolute error:

$$f_1(S) = \sum_{p=1}^{m} (\hat{y}_S(\mathbf{x}_p) - y_p)^2 / 2 \quad \text{or} \quad f_1(S) = \sum_{p=1}^{m} | \hat{y}_S(\mathbf{x}_p) - y_p |, \tag{12}$$

where $\hat{y}_S(\mathbf{x}_p)$ is the estimated output by the rule set S (see [6]).

Linguistic rule extraction for function approximation can be formulated as the following three-objective combinatorial optimization problem:

$$\text{Minimize } f_1(S), f_2(S), \text{ and } f_3(S), \tag{13}$$

where $f_2(S)$ and $f_3(S)$ are the number of linguistic rules in S and the total rule length in S, respectively, as in the case of pattern classification. Our task is to find non-dominated rule sets of the three-objective linguistic rule selection problem.

For this task, we can use the rule selection scheme and the GBML scheme described for pattern classification. The following points should be considered when we apply these schemes to function approximation:

(a) The first objective is an error measure such as (12). The magnitude of output values should be taken into account when we use the weighted averaging technique for combining the three objectives into a scalar fitness function.

(b) We should use a fuzzy reasoning method that can handle a rule set of linguistic rules with different specificity levels because our linguistic rules involves an arbitrary number of "*don't care*" conditions. Inconsistency among linguistic rules may be resolved by such a fuzzy reasoning method (see [6]).

(c) Since the consequent part of each linguistic rule is not uniquely specified, its consequent linguistic value is coded together with its antecedent linguistic values in the GBML scheme. This modification increases the search space from 6^n to 5×6^n. Usually an appropriate consequent value for each linguistic rule can be limited to a few alternatives. This can be used for decreasing the search space.

7 Conclusion

We first formulated linguistic rule extraction from numerical data for pattern classification as a three-objective combinatorial optimization problem. Next we explained how non-dominated rule sets can be obtained from candidate rules by genetic algorithms. In the rule selection method, a prescreening procedure of

candidate rules is necessary when it is applied to high-dimensional pattern classification problems. Then we explained how non-dominated rule sets can be found by a genetics-based machine learning algorithm in a huge search space with all possible linguistic rules. These two schemes were applied to wine data with 13 attributes. Simulation results showed that compact rule sets with high classification performance were found. Finally we modified our linguistic rule extraction problem for applying it to function approximation.

Acknowledgement. This study was partially supported by Saneyoshi Scholarship Foundation.

Appendix: Rule Generation and Pattern Classification

The consequent class C_j and the certainty grade CF_j of our linguistic rule in (1) can be determined by the following heuristic procedure (for example, see [15]):

Step 1) Calculate the compatibility grade $\mu_{R_j}(\mathbf{x}_p)$ of each training pattern $\mathbf{x}_p = (x_{p1}, ..., x_{pn})$ with the linguistic rule R_j by the product operation as

$$\mu_{R_j}(\mathbf{x}_p) = \mu_{j1}(x_{p1}) \times \ ... \ \times \mu_{jn}(x_{pn}), \ p = 1,2,...,m, \tag{A1}$$

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

Step 2) For each class, calculate the total compatibility grade of the training patterns with the linguistic rule R_j:

$$\beta_{\text{Class } h}(R_j) = \sum_{\mathbf{x}_p \in \text{Class } h} \mu_{R_j}(\mathbf{x}_p), \quad h = 1,2,...,c. \tag{A2}$$

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

$$\beta_{\text{Class } C_j}(R_j) = \text{Max}\{\beta_{\text{Class } 1}(R_j), ..., \beta_{\text{Class } c}(R_j)\}. \tag{A3}$$

If the consequent class can not be uniquely determined, we do not extract the linguistic rule R_j. For example, if $\beta_{\text{Class } h}(R_j) = 0$ for all classes, we do not generate R_j.

Step 4) Specify the certainty grade CF_j as follows:

$$CF_j = \{\beta_{\text{Class } C_j}(R_j) - \overline{\beta}\} / \sum_{h=1}^{c} \beta_{\text{Class } h}(R_j), \tag{A4}$$

where

$$\overline{\beta} = \sum_{\substack{h=1 \\ h \neq C_j}}^{c} \beta_{\text{Class } h}(R_j)/(c-1). \tag{A5}$$

Let S be a set of generated linguistic rules. A new pattern $\mathbf{x}_p = (x_{p1},...,x_{pn})$ is classified by S using a fuzzy reasoning method based on a single winner rule [15]. The winner rule R_{j^*} for $\mathbf{x}_p = (x_{p1}, ..., x_{pn})$ is determined as follows:

$$\mu_{R_{j^*}}(\mathbf{x}_p) \cdot CF_{j^*} = \max\{\mu_{R_j}(\mathbf{x}_p) \cdot CF_j \mid R_j \in S\}. \quad \text{(A6)}$$

The new pattern $\mathbf{x}_p$ is classified by the winner rule R_{j^*}. That is, $\mathbf{x}_p$ is assigned to the consequent class of R_{j^*}. If multiple linguistic rules with different consequent classes have the same maximum value in (A6), the classification of $\mathbf{x}_p$ is rejected.

References

1. Abe, S., Lan, M.-S., Thawonmas, R.: Tuning of a Fuzzy Classifier Derived from Data, *Int. J. Approximate Reasoning* 14 (1996) 1-24.
2. Cordon, O., del Jesus, M. J., Herrera, F.: A Proposal on Reasoning Methods in Fuzzy Rule-Based Classification Systems, *Int. J. Approximate Reasoning* 20 (1999) 21-45.
3. Herrera, F., Lozano, M., Verdegay, J. L.: Tuning Fuzzy Logic Controllers by Genetic Algorithms, *Int. J. Approximate Reasoning* 12 (1995) 299-315, 1995.
4. Holte, R. C.: Very Simple Classification Rules Perform Well on Most Commonly Used Dataset, *Machine Learning* 11 (1993) 63-91.
5. Horikawa, S., Furuhashi, T., Uchikawa, Y.: On Fuzzy Modeling Using Fuzzy Neural Networks with the Back-Propagation Algorithm, *IEEE Trans. on Neural Networks* 3 (1992) 801-806.
6. Ishibuchi, H.: Fuzzy Reasoning Method in Fuzzy Rule-based Systems with General and Specific Rules for Function Approximation, *Proc. of 8th IEEE Int. Conference on Fuzzy Systems* (1999) 198-203.
7. Ishibuchi, H., Murata, T., Nakashima, T.: Linguistic Rule Extraction from Numerical Data for High-Dimensional Classification Problems, *Int. J. Advanced Computational Intelligence* 3 (1999) 386-393.
8. Ishibuchi, H., Murata, T., Turksen, I. B.: Single-Objective and Two-Objective Genetic Algorithms for Selecting Linguistic Rules for Pattern Classification Problems, *Fuzzy Sets and Systems* 89 (1997) 135-149.
9. Ishibuchi, H., Nakashima, T.: Improving the Performance of Fuzzy Classifier Systems for Pattern Classification Problems with Continuous Attributes, *IEEE Trans. on Industrial Electronics* 46 (1999) 1057-1068.
10. Ishibuchi, H., Nakashima, T.: Linguistic Rule Extraction by Genetics-Based Machine Learning, *Proc. of Genetic and Evolutionary Computation Conference* (2000) 195-202.
11. Ishibuchi, H., Nakashima, T., Kuroda, T.: A Hybrid Fuzzy GBML Algorithm for Designing Compact Fuzzy Rule-Based Classification Systems, *Proc. of 9th IEEE International Conference on Fuzzy Systems* (2000) 706-711.
12. Ishibuchi, H., Nakashima, T., Morisawa, T.: Simple Fuzzy Rule-Based Classification Systems Perform well on Commonly Used Real-World Data Sets, *Proc. of 16th Annual Meeting of the North American Fuzzy Information Processing Society* (1997) 251-256.
13. Ishibuchi, H., Nakashima, T., Murata, T.: Genetic-Algorithm-Based Approaches to the Design of Fuzzy Systems for Multi-Dimensional Pattern Classification Problems, *Proc. of 3rd IEEE Int. Conference on Evolutionary Computation* (1996) 229-234.
14. Ishibuchi, H., Nakashima, T., Murata, T.: Performance Evaluation of Fuzzy Classifier Systems for Multi-Dimensional Pattern Classification Problems, *IEEE Trans. on Systems, Man, and Cybernetics - Part B: Cybernetics* 29 (1999) 601-618.

15. Ishibuchi, H., Nozaki, K., Yamamoto, N., Tanaka, H.: Selecting Fuzzy If-Then Rules for Classification Problems using Genetic Algorithms, *IEEE Trans. on Fuzzy Systems* 3 (1995) 260-270.
16. Jang, J. -S. R.: ANFIS: Adaptive-Network-Based Fuzzy Inference System, *IEEE Trans. on Systems, Man, and Cybernetics* 23 (1993) 665-685.
17. Jin, Y.: Fuzzy Modeling of High-Dimensional Systems: Complexity Reduction and Interpretability Improvement, *IEEE Transactions on Fuzzy Systems* 8 (2000) 212-221.
18. Jin, Y., von Seelen, W., Sendhoff, B.: On Generating FC^3 Fuzzy Rule Systems from Data using Evolution Strategies, *IEEE Transactions on Systems, Man and Cybernetics - Part B: Cybernetics* 29 (1999) 829-845.
19. Karr, C. L., Gentry, E. J.: Fuzzy Control of pH using Genetic Algorithms, *IEEE Trans. on Fuzzy Systems* 1 (1993) 46-53.
20. Kosko, B.: Fuzzy Systems as Universal Approximators, *Proc. of 1st IEEE Int. Conference on Fuzzy Systems* (1992) 1153-1162.
21. Leondes, C. T. (ed.): *Fuzzy Theory Systems: Techniques and Applications*, Academic Press, San Diego (1999).
22. de Oliveira, V.: Semantic Constraints for Membership Function Optimization, *IEEE Trans. on Systems, Man, and Cybernetics - Part A: Systems and Humans* 29 (1999) 128-138.
23. Pedrycz, W., de Oliveira, V.: Optimization of Fuzzy Models, *IEEE Trans. on Systems, Man, and Cybernetics - Part B: Cybernetics* 26 (1996) 627-637.
24. Setnes, M., Babuska, R., Kaymak, U., van Nauta Lemke, H. R.: Similarity Measures in Fuzzy Rule Base Simplification, *IEEE Trans. on Systems, Man, and Cybernetics - Part B: Cybernetics* 28 (1998) 376-366.
25. Setnes, M., Babuska, R., Verbruggen, B.: Rule-Based Modeling: Precision and Transparency, *IEEE Trans. on Systems, Man, and Cybernetics - Part C: Applications and Reviews* 28 (1998) 165-169.
26. Setnes, M., Roubos, H.: GA-Fuzzy Modeling and Classification: Complexity and Performance, *IEEE Trans. on Fuzzy Systems* 8 (2000) 509-522.
27. Takagi, T., Sugeno, M.: Fuzzy Identification of Systems and Its Applications to Modeling and Control, *IEEE Trans. on Systems, Man, and Cybernetics* 15 (1985) 116-132.
28. van Veldhuizen, D. A., Lamont, G. B.: Multiobjective Evolutionary Algorithms: Analyzing the State-of-the-Art, *Evolutionary Computation* 8 (2000) 125-147.
29. Wang, L. -X.: Fuzzy Systems are Universal Approximators, *Proc. of 1st IEEE Int. Conference on Fuzzy Systems* (1992) 1163-1170.
30. Wang, L. X., Mendel, J. M.: Generating Fuzzy Rules by Learning from Examples, *IEEE Trans. on Systems, Man, and Cybernetics* 22 (1992) 1414-1427.
31. Yen, J., Wang, L.: Simplifying Fuzzy Rule-Based Models using Orthogonal Transformation Methods, *IEEE Trans. on Systems, Man, and Cybernetics - Part B: Cybernetics* 29 (1999) 13-24.
32. Yen, J., Wang, L., Gillespie, W.: Improving the Interpretability of TSK Fuzzy Models by Combining Global Learning and Local Learning, *IEEE Trans. on Fuzzy Systems* 6 (1998) 530-537.
33. Zitzler, E., Deb, K., Thiele, L.: Comparison of Multiobjective Evolutionary Algorithms: Empirical Results, *Evolutionary Computation* 8 (2000) 173-195.
34. Zitzler, E., Thiele, L.: Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach, *IEEE Transactions on Evolutionary Computation* 3 (1999) 257-271.