1999 IEEE International Fuzzy Systems Conference Proceedings
August 22-25, 1999, Seoul, Korea

Genetic-Algorithm-Based Approach to Linguistic Approximation of Nonlinear Functions with Many Input Variables

Hisao Ishibuchi and Tomoharu Nakashima
Department of Industrial Engineering, Osaka Prefecture University
Gakuen-cho 1-1, Sakai, Osaka 599-8531, Japan
Phone: +81-722-54-9351 FAX: +81-722-54-9915
{hisaoi, nakashi}@ie.osakafu-u.ac.jp
http://www.ie.osakafu-u.ac.jp/student/ci_lab/ci_lab_e/index.html

Abstract

In this paper, we propose a genetic-algorithm-based approach for extracting a small number of fuzzy if-then rules with clear linguistic meanings from numerical input-output data. The goal of our fuzzy rule extraction is to linguistically describe the input-output relation of a nonlinear function with many input variables in a human understandable manner. In other words, the goal is to construct a comprehensible fuzzy rule-based system from numerical input-output data. The comprehensibility of a fuzzy rule-based system is evaluated by three criteria: linguistic interpretability of fuzzy if-then rules, simplicity of fuzzy if-then rules, and compactness of a fuzzy rule-based system. That is, a comprehensible fuzzy rule-based system consists of a small number of simple fuzzy if-then rules with clear linguistic meanings. In order to cover a multi-dimensional input space by a small number of fuzzy if-then rules, we use general rules with many "*don't care*" conditions in the antecedent part. Such general rules are also preferable from a viewpoint of the simplicity of fuzzy if-then rules. Since nonlinear functions can not be always approximated by only general rules, some specific rules with many linguistic conditions may be also required in many cases. Thus our fuzzy rule-based system is a mixture of general and specific fuzzy if-then rules. In this paper, we first illustrate the necessity of general rules with many "*don't care*" conditions when we try to construct compact fuzzy rule-based systems for high-dimensional problems without the exponential increase in the number of fuzzy if-then rules. Next we demonstrate that a standard fuzzy reasoning method sometimes leads to counter-intuitive results when some specific rules are included in other general rules. Then we illustrate a fuzzy reasoning method for realizing default hierarchies of fuzzy if-then rules. The default hierarchies mean that specific rules have priority over general rules when output values are inferred by fuzzy if-then rules. Finally we show how genetic algorithms can be utilized for generating a small number of fuzzy if-then rules from numerical input-output data.

1. Introduction

Fuzzy systems based on fuzzy if-then rules have been successfully applied to various control problems [1,2]. In those applications, fuzzy rule-based systems are used as approximators of nonlinear functions. It was shown that they have high capability to approximate nonlinear functions [3,4]. Their main advantage over black-box type approximators such as neural networks is the high comprehensibility of fuzzy if-then rules. When a fuzzy rule-based system is used for approximately realizing a nonlinear function with two input variables, fuzzy if-then rules are usually written in a tabular form. In this case, we can linguistically understand the fuzzy rule-based system and imagine the 3-D shape of its input-output relation. This is because the antecedent and consequent parts of each fuzzy if-then rule are specified by linguistic values such as "*small*" and "*large*". In Fig. 1, we show an example of a fuzzy rule table for approximately realizing a nonlinear function. The two-dimensional input space $[0, 1]^2$ in Fig. 1 is partitioned into 25 fuzzy subspaces by five linguistic values on each input variable (S: *small*, MS: *medium small*, M: *medium*, ML: *medium large*, and L: *large*). Fig. 1 shows 25 fuzzy if-then rules such as "If x_1 is *small* and x_2 is *small* then y is *large*". From the 25 fuzzy if-then rules in Fig. 1, we can imagine the 3-D shape of the input-output relation of the fuzzy rule-based system. The imagined shape may be something like Fig. 2.

The main difficulty in applying fuzzy rule-based systems to high-dimensional problems is the exponential increase in the number of fuzzy if-then rules with the dimensionality of input spaces. For example, if we use five linguistic values for each input variable as in Fig. 1, the size of fuzzy rule tables is 5^n for an n-dimensional problem (e.g., $5^n = 25$ for $n = 2$ and $5^n = 3125$ for $n = 5$).

Such exponential increase deteriorates the comprehensibility of fuzzy rule-based systems. That is, it is impractical for human users to manually examine thousands of fuzzy if-then rules. One approach to the handling of high-dimensional problems by fuzzy if-then rules is the use of multi-dimensional antecedent fuzzy sets whose membership functions are directly defined on the input space. In this approach, the antecedent part of each fuzzy if-then rule is not defined by a combination of linguistic values but directly specified by a single multi-dimensional fuzzy set. The use of multi-dimensional antecedent fuzzy sets can drastically decrease the number of fuzzy if-then rules for high-dimensional problems. It also increases the flexibility of fuzzy if-then

0-7803-5406-0/99/$10.00

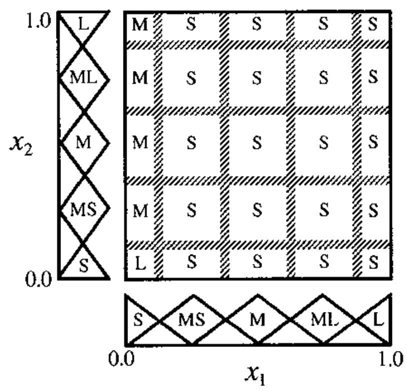


Figure 1: Tabular form representation of 25 fuzzy rules.

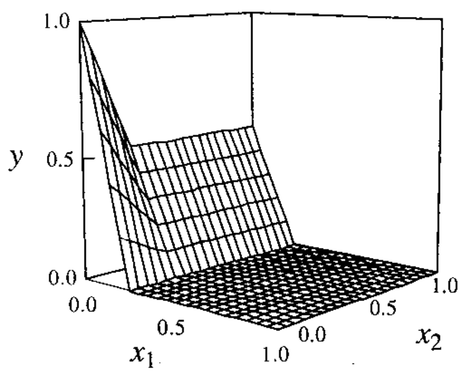


Figure 2: Input-output relation realized by the fuzzy rules.

rules. One drawback of this approach is that the linguistic interpretation of each fuzzy if-then rule is not always easy. Another approach is the use of a hierarchical structure where several subsystems are hierarchically combined into a multi-layer fuzzy rule-based system. Each subsystem is a fuzzy rule-based system with a few inputs. The main advantage of this approach over standard (i.e., plain) fuzzy rule-based systems is that the number of input variables to each subsystem is small. This means that the number of fuzzy if-then rules is also small. From a viewpoint of the comprehensibility of fuzzy rule-based systems, this approach inherently involves difficulties in interpreting intermediate and output subsystems whose inputs are supplied by lower subsystems. That is, it is very difficult for human users to understand (or interpret) each fuzzy if-then rule in such subsystems because their inputs have no clear physical meanings.

Fuzzy if-then rules in this paper are used for linguistically describing the input-output relation of a nonlinear function with many input variables in a human understandable manner. In other words, our goal is to construct a comprehensible fuzzy rule-based system from numerical input-output data. For avoiding the exponential increase in the number of fuzzy if-then rules, we use general rules with many *"don't care"* conditions in the antecedent part. Since specific rules with many antecedent conditions may be also required for function approximation, our fuzzy rule-based system is a mixture of general and specific fuzzy if-then rules. In this paper, we propose a framework of genetic-algorithm-based rule extraction from numerical input-output data for linguistically describing a nonlinear function with many input variables in a human understandable manner.

2. Fuzzy Reasoning with General Fuzzy If-Then Rules

2.1. Fuzzy If-Then Rules for High-Dimensional Problems

For approximately realizing a nonlinear function with n input variables, we use fuzzy if-then rules of the following form:

$$\text{Rule } R_j\text{: If } x_1 \text{ is } A_{j1} \text{ and } \cdots \text{ and } x_n \text{ is } A_{jn} \text{ then } y \text{ is } B_j,\ \ j = 1, 2, \cdots N, \tag{1}$$

where j is a rule index, A_{ji}'s are antecedent fuzzy sets with linguistic labels such as *small* and *large* (i.e., A_{ji}'s are linguistic values), B_j is a consequent linguistic value, N is the total number of fuzzy if-then rules, $\mathbf{x}$ is an n-dimensional input vector $\mathbf{x} = (x_1, \cdots, x_n)$, and y is an output variable. Examples of fuzzy if-then rules with typical linguistic values are shown in Fig. 1. In this paper, we assume that the input space of the nonlinear function is the n-dimensional unit hyper-cube $[0, 1]^n$. We also assume that the output space is the unit interval $[0, 1]$. Thus our problem is to approximately realize the nonlinear mapping from $[0, 1]^n$ to $[0, 1]$. We use the five linguistic values in Fig. 1 for all the input and output variables. In this case, the size of a fuzzy rule table (i.e., the number of fuzzy if-then rules) is calculated for an n-dimensional problem as shown in Table 1.

From Table 1, we can see that a fuzzy rule-based system is not comprehensible for human users when the number of input variables is more than two. Even in the case of $n = 3$, it is a troublesome task for human users to manually examine each of 125 fuzzy if-then rules. The number of fuzzy if-then rules can be drastically decreased if we use only general fuzzy if-then rules with many *"don't care"* conditions [5]. Let us define the length of a fuzzy if-then rule by the number of antecedent conditions excluding *"don't care"*. For example, the length of the following fuzzy if-then rule is two:

$$\text{If } x_1 \text{ is } \textit{small} \text{ and } x_2 \text{ is } \textit{don't care} \text{ and } x_3 \text{ is } \textit{don't care} \text{ and } x_4 \text{ is } \textit{don't care} \text{ and } x_5 \text{ is } \textit{large} \text{ then } y \text{ is } \textit{large}. \tag{2}$$

This fuzzy if-then rule can be rewritten by omitting the *"don't care"* conditions as "If x_1 is *small* and x_5 is *large* then y is *large*" with the two antecedent conditions. If we use only general fuzzy if-then rules whose lengths are two or less, the number of fuzzy if-then rules is calculated for an n-dimensional problem as shown in Table 2. For example, in the case of $n = 2$, there are 25 rules of the length 2, 10 rules of the length 1, and a single rule of the length 0. The fuzzy

Table 1: The number of fuzzy if-then rules for an n-dimensional problem.

$n=1$	$n=2$	$n=3$	$n=4$	$n=5$	$n=6$	$n=7$	$n=8$	$n=9$	$n=10$
5	25	125	625	3125	15625	78125	390625	1953125	9765625

Table 2: The number of general fuzzy if-then rules for an n-dimensional problem.

$n=1$	$n=2$	$n=3$	$n=4$	$n=5$	$n=6$	$n=7$	$n=8$	$n=9$	$n=10$
6	36	91	171	276	406	561	741	946	1176

if-then rule of the length 0, which has no antecedent condition, is written in the form "y is B_j". From the comparison between Table 1 and Table 2, we can see that the number of general fuzzy if-then rules is very small if compared with the total number of fuzzy if-then rules of the length n. Since a general fuzzy if-then rule has only a few antecedent conditions, it can cover a large area of the input space. This means that the entire input space can be covered by a small number of general fuzzy if-then rules. Since nonlinear functions can not be always approximated by only general rules, some specific rules with many antecedent conditions may be also necessary in many cases. Thus our fuzzy rule-based system is a mixture of general and specific fuzzy if-then rules.

2.2. Simplified Fuzzy Reasoning

One of the most frequently used fuzzy reasoning methods for function approximation problems is the simplified fuzzy reasoning method [6]. In the simplified fuzzy reasoning method, the estimated value $\hat{y}$ for the input vector $\mathbf{x} = (x_1, \cdots, x_n)$ is calculated from the N fuzzy if-then rules in (1) as follows:

$$\hat{y} = \hat{y}(\mathbf{x}) = \sum_{j=1}^{N} b_j \cdot \mu_j(\mathbf{x}) \Big/ \sum_{j=1}^{N} \mu_j(\mathbf{x}), \qquad (3)$$

where b_j is a representative real number (i.e., modal value) of the consequent fuzzy set B_j, and $\mu_j(\mathbf{x})$ is the compatibility grade of the input vector $\mathbf{x}$ with the j-th fuzzy if-then rule R_j. In computer simulations of this paper, we use the center of the triangular membership function of each consequent fuzzy set B_j as its representative real number b_j. The compatibility grade $\mu_j(\mathbf{x})$ is often defined by the following product operation:

$$\mu_j(\mathbf{x}) = \mu_{j1}(x_1) \times \mu_{j2}(x_2) \times \cdots \times \mu_{jn}(x_n), \qquad (4)$$

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

Let us illustrate the simplified fuzzy reasoning method by some numerical examples. The first example does not include "*don't care*" conditions. We applied the simplified fuzzy reasoning method to the 25 fuzzy if-then rules in Fig. 1. The shape of the estimated nonlinear function $\hat{y} = \hat{y}(\mathbf{x})$ is depicted in Fig. 2. From the comparison between Fig. 1 and Fig. 2, we can see that the estimated nonlinear function $\hat{y} = \hat{y}(\mathbf{x})$ coincides with our intuitive understanding of the 25 fuzzy if-then rules in Fig. 1.

Next we consider the following mixture of general and specific fuzzy if-then rules:

R_A: y is *small*,
R_B: If x_1 is *small* then y is *medium*,
R_C: If x_1 is *small* and x_2 is *small* then y is *large*.

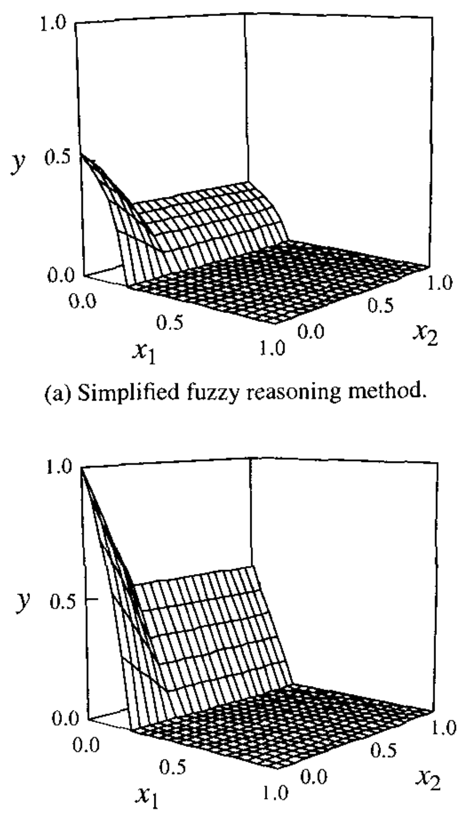
Let us try to imagine the 3-D shape of the nonlinear function realized by these three fuzzy if-then rules. The imagined 3-D shape may be something like Fig. 2. If we use these three fuzzy if-then rules in the following hierarchical manner, they are almost the same as the 25 fuzzy if-then rules in Fig. 1.

If x_1 is *small* and x_2 is *small* then y is *large*
else {if x_1 is *small* then y is *medium*
else y is *small*}. (5)

In this hierarchical structure, the most specific rule R_C has priority over the other rules R_A and R_B in inferring the output value y. If the antecedent conditions of R_C are not satisfied by the input vector, the next rule R_B has priority over R_A. The most general rule R_A is used only when the input vector is not compatible with the other rules. It seems that we usually use our knowledge in such a hierarchical manner when we have general rules together with specific (or exceptional) rules. That is, a specific rule is usually used with priority over a general rule when the antecedent conditions of both rules are satisfied. Such a default hierarchy of nonfuzzy rules was discussed in Holland et al. [7]. The priority of specific rules has been discussed in the AI community (see, for example, Poole [8]). We applied the simplified fuzzy reasoning method to the three fuzzy if-then rules R_A, R_B and R_C. The shape of the estimated nonlinear function $\hat{y} = \hat{y}(\mathbf{x})$ is depicted in Fig. 3 (a). Unfortunately the shape in Fig. 3 (a) does not coincide with our intuitive understanding of the three fuzzy if-then rules.

2.3. Fuzzy Reasoning for Default Hierarchies

We have illustrated that the simplified fuzzy reasoning method can not handle the default hierarchy of fuzzy if-then



(a) Simplified fuzzy reasoning method.

(b) Our fuzzy reasoning method.

Figure 3: Shape of the estimated nonlinear function $\hat{y} = \hat{y}(\mathbf{x})$.

rules in the previous section. As a result, estimated nonlinear functions do not always coincide with our intuition. For obtaining intuitively acceptable results, we modify the simplified fuzzy reasoning method as follows:

$$\hat{y} = \hat{y}(\mathbf{x}) = \sum_{j=1}^{N} \phi(R_j, \mathbf{x}) \cdot b_j \cdot \mu_j(\mathbf{x}) \Big/ \sum_{j=1}^{N} \phi(R_j, \mathbf{x}) \cdot \mu_j(\mathbf{x}), \tag{6}$$

where $\phi(R_j, \mathbf{x})$ is a function describing the specificity of the fuzzy if-then rule R_j. The value of $\phi(R_j, \mathbf{x})$ becomes small when R_j includes other specific rules compatible with the input vector $\mathbf{x}$. In this case, the weight of R_j is discounted in the fuzzy reasoning. More specifically, we define $\phi(R_j, \mathbf{x})$ as follows:

$$\phi(R_j, \mathbf{x}) = \prod_{\substack{k \neq j \\ R_k \subseteq R_j}} (1 - \mu_k(\mathbf{x})), \tag{7}$$

where the inclusion relation $R_k \subseteq R_j$ between two fuzzy if-then rules R_k and R_j is defined by their antecedent fuzzy sets A_{ki}'s and A_{ji}'s as

$$R_k \subseteq R_j \Leftrightarrow A_{ki} \subseteq A_{ji} \text{ for } i = 1, 2, \cdots, n. \tag{8}$$

In (7), the weight of the fuzzy if-then rule R_j is discounted when R_j includes other specific rules. If R_j includes no fuzzy if-then rule, $\phi(R_j, \mathbf{x})$ is specified as $\phi(R_j, \mathbf{x}) = 1$.

First we applied the proposed fuzzy reasoning method to the 25 fuzzy if-then rules in Fig. 1. Since no inclusion relation holds among the 25 fuzzy if-then rule, the same result (i.e., Fig. 2) was obtained by the proposed fuzzy reasoning method as in the case of the simplified fuzzy reasoning method. Next we applied the proposed fuzzy reasoning method to the three fuzzy if-then rules R_A, R_B and R_C. Since the inclusion relation $R_C \subseteq R_B \subseteq R_A$ holds, $\phi(R_j, \mathbf{x})$ in (7) is written as

$$\begin{aligned} \phi(R_A, \mathbf{x}) &= (1 - \mu_B(\mathbf{x})) \times (1 - \mu_C(\mathbf{x})), \\ \phi(R_B, \mathbf{x}) &= 1 - \mu_C(\mathbf{x}), \\ \phi(R_C, \mathbf{x}) &= 1. \end{aligned} \tag{9}$$

The shape of the estimated nonlinear function $\hat{y} = y(\mathbf{x})$ is depicted in Fig. 3 (b). From Fig. 3 (b), we can see that the estimated nonlinear function coincides with our intuition.

3. Genetic-Algorithm-Based Approach

3.1. Basic Idea

We have already shown that general fuzzy if-then rules with many *"don't care"* conditions are necessary for constructing a compact fuzzy rule-based system with high comprehensibility for a high-dimensional problem. We have also modified the simplified fuzzy reasoning method for obtaining intuitively acceptable fuzzy reasoning results from a mixture of general and specific fuzzy if-then rules where default hierarchies exist. Our goal in this section is to extract a small number of fuzzy if-then rules from numerical input-output data for constructing a compact fuzzy rule-based system. Our approach to the fuzzy rule extraction is based on the two ideas we have already discussed: general fuzzy if-then rules and default hierarchies. Genetic algorithms are used in our approach for finding a small number of fuzzy if-then rules with high approximation ability.

We explain our approach using the five linguistic values in Fig. 1 for the simplicity of explanation. Of course, a collection of tailored linguistic values should be specified for appropriately describing each input (or output) variable in a particular application. As we have already explained, we also use *"don't care"* as an antecedent fuzzy set. When we approximately realize a nonlinear function with n input variables by fuzzy if-then rules, there exist $(5 + 1)^n$ combinations of antecedent fuzzy sets. For each combination of antecedent fuzzy sets, one of the five linguistic values is chosen as its consequent fuzzy set. Thus the total number of possible fuzzy if-then rules is $5 \times (5 + 1)^n$. Let S_{ALL} be the set of these $5 \times (5 + 1)^n$ fuzzy if-then rules. We denote a subset of S_{ALL} by S. Our rule extraction can be described as finding a compact subset S with high approximation ability. The approximation ability of the rule set S is measured by the total absolute error on training data $(\mathbf{x}_p, y_p), p = 1, 2, \cdots, m$:

$$e(S) = \sum_{p=1}^{m} |y_p - \hat{y}_p|, \qquad (10)$$

where $\hat{y}_p$ is the estimated output value for y_p. In (10), $\hat{y}_p$ is calculated by the proposed fuzzy reasoning method from the fuzzy if-then rules in the rule set S. Of course, we can use the total squared error as the performance measure instead of (10). If there is no compatible fuzzy if-then rule with the input vector $\mathbf{x}_p$, $\hat{y}_p$ can not be calculated. In this case, we define $|y_p - \hat{y}_p|$ as $|y_p - \hat{y}_p| = 1$, which is the maximum value of possible errors.

Since our goal is to find a small number of fuzzy if-then rules, the size of the rule set S (i.e., the number of fuzzy if-then rules included in S) is another criterion for measuring the quality of S. It is directly minimized by genetic algorithms. Thus our problem can be written as

$$\begin{aligned} \text{Minimize} \quad & z(S) = e(S) \\ & + W_{Compactness} \cdot Cardinality(S) \\ \text{subject to} \quad & S \subseteq S_{ALL}, \end{aligned} \qquad (11)$$

where $z(S)$ is an objective function to be minimized, $W_{Compactness}$ is a positive weight, and $Cardinality(S)$ is the number of fuzzy if-then rules included in S. When the direct minimization of the size of S is difficult, we introduce the upper limit $N_{\max}$ of the number of fuzzy if-then rules and modify the formulation in (11) as follows:

$$\begin{aligned} \text{Minimize} \quad & z(S) = e(S) \\ \text{subject to} \quad & S \subseteq S_{ALL} \\ & \text{and } Cardinality(S) \le N_{\max}, \end{aligned} \qquad (12)$$

3.2. Fuzzy Rule Selection

The rule extraction problem formulated in (11) can be handled as a rule selection problem in a similar manner to Ishibuchi et al.[10] where genetic algorithms were used for selecting a small number of fuzzy if-then rules for pattern classification problems. Let N be the total number of fuzzy if-then rules included in the rule set S_{ALL}. In this case, a subset S of S_{ALL} can be coded as a binary string of the length N: $S = s_1 s_2 \cdots s_N$ where $s_j = 1$ and $s_j = 0$ mean the inclusion and the exclusion of the j-th fuzzy if-then rule, respectively. That is, the binary string $S = s_1 s_2 \cdots s_N$ is decoded as $S = \{R_j | s_j = 1, j = 1, 2, \cdots N\}$. Since our objective function in (11) should be minimized, we define a fitness function as follows for applying genetic algorithms to the rule selection problem.

$$\begin{aligned} fitness(S) \; &= -e(S) \\ & - W_{Compactness} \cdot Cardinality(S). \end{aligned} \qquad (13)$$

Let us illustrate the rule selection method through computer simulations on several numerical examples. First we generated $21^2 = 441$ numerical input-output pairs from Fig. 3 (a) by specifying input vectors $\mathbf{x}_p = (x_{p1}, x_{p2})$ as $x_{pi} = 0.00, 0.05, \cdots, 1.00$ for $i = 1, 2$. Since the nonlinear function to be linguistically described has two input variables, the total number of possible fuzzy if-then rules is $N = 5 \times 6 \times 6 = 180$. Each subset of these 180 fuzzy if-then rules is denoted by a binary string of the length 180. A genetic algorithm was applied to the 441 input-output pairs for selecting a small number of fuzzy if-then rules from these 180 rules. The following three fuzzy if-then rules were selected:

R_A: y is *small*,
R_B: If x_1 is *small* then y is *medium small*,
R_C^*: If x_1 is *small* and x_2 is *small* then y is *medium*.

We can see that the obtained three fuzzy if-then rules linguistically describe the shape of the nonlinear function in Fig. 3(a) very well. For comparison, we applied the same genetic algorithm to the 441 input-output data using the simplified fuzzy reasoning method. In this case, we also obtained three fuzzy if-then rules. Two rules are R_A and R_B. The other rule is not R_C^* but R_C: "If x_1 is *small* and x_2 is *small* then y is *large*".

Next we generated $11^3 = 1331$ numerical input-output pairs from the following nonlinear function by specifying input vectors $\mathbf{x}_p = (x_{p1}, x_{p2}, x_{p3})$ as $x_{pi} = 0.0, 0.1, \cdots, 1.0$ for $i = 1, 2, 3$.

$$y_p = \frac{1}{2\left(1 + \exp\{\sum_{i=1}^{3}(-60x_{pi} + 55)\}\right)}, \quad x_{pi} \in [0, 1] \text{ for } i = 1, 2, 3. \qquad (14)$$

Since the nonlinear function involves three input variables, the total number of possible fuzzy if-then rules is $N = 5 \times 6 \times 6 \times 6 = 1080$. A genetic algorithm was applied to the 1331 numerical input-output pairs for selecting a small number of fuzzy if-then rules from these 1080 rules. The following two fuzzy if-then rules were selected:

y is *small*,
If x_1 is *large* and x_2 is *large* and x_3 is *large* then y is *medium*.

From the two fuzzy if-then rules, we can understand the shape of the input-output relation of the nonlinear function.

3.3. Rule Generation

When the total number of possible fuzzy if-then rules is intractably large, we can not use the rule selection method in the previous subsection. In that case, we use genetic algorithms for generating fuzzy if-then rules. Each fuzzy if-then rule is denoted by its antecedent and consequent fuzzy sets. For example, a fuzzy if-then rule "If x_1 is *small* and x_2 is *don't care* and x_3 is *medium small* then y is *large*" is coded as "1025" where each linguistic value is represented in the following manner:

0: *don't care*, 1: *small*, 2: *medium small*,
3: *medium*, 4: *medium large*, 5: *large*.

In general, a fuzzy if-then rule is denoted by a string of the length $(n+1)$ where n is the number of input variables of the nonlinear function to be approximated. A rule set S is coded as a concatenated string where each substring denotes a fuzzy if-then rule. When we solve the rule extraction problem formulated in (12), a rule set S is denoted by a string of the length $(n+1)\cdot N_{\max}$ where $N_{\max}$ is the upper limit of the number of fuzzy if-then rules. Since the objective function in (12) is to be minimized, we define a fitness function as follows:

$$fitness(S) = -e(S). \qquad (15)$$

By specifying $N_{\max}$ as $N_{\max} = 5$, we applied the rule generation method in this subsection to the numerical examples in the previous subsection. Almost the same simulation results as in the previous subsection were obtained by the rule generation method. Since the number of fuzzy if-then rules was not minimized in this method, unnecessary fuzzy if-then rules were sometimes included in the final solutions.

4. Conclusion

In this paper, we proposed a genetic-algorithm-based approach to the linguistic modeling of nonlinear functions with many input variables. For constructing a compact fuzzy rule-based system for a high-dimensional problem, we utilized general fuzzy if-then rules with many "*don't care*" conditions. Our fuzzy rule-based system was a mixture of general and specific fuzzy if-then rules. We modified the simplified fuzzy reasoning method for representing default hierarchies of fuzzy if-then rules. The default hierarchies mean that specific fuzzy if-then rules have priority over general rules. Two characteristic features of our linguistic modeling are the use of general fuzzy if-then rules and the fuzzy reasoning method based on the default hierarchies of fuzzy if-then rules. Through computer simulations, we demonstrated that genetic algorithms can find a small number of fuzzy if-then rules from numerical input-output data.

References

[1] M. Sugeno, "An introductory survey of fuzzy control," *Information Sciences*, vol. 36, no. 1/2, pp. 59-83, 1985.

[2] C. C. Lee, "Fuzzy logic in control systems: fuzzy logic controller Part I and Part II," *IEEE Trans. on Systems, Man, and Cybernetics*, vol. 20, no. 2, pp. 404-435, 1990.

[3] B. Kosko, "Fuzzy systems as universal approximators," *Proc. of 1st FUZZ-IEEE*, pp. 1153-1162, 1992.

[4] L. -X. Wang, "Fuzzy systems are universal approximators," *Proc. of 1st FUZZ-IEEE*, pp. 1163-1170, 1992.

[5] H. Ishibuchi, and T. Murata, "Minimizing the fuzzy rule base and maximizing its performance by a multi-objective genetic algorithm," *Proc. of 6th FUZZ-IEEE*, pp. 259-264, July, 1997.

[6] H. Ichihashi, and T. Watanabe, "Learning control by fuzzy models using a simplified fuzzy reasoning," *Journal of Japan Society for Fuzzy Theory and Systems*, vol. 2, no. 3, pp. 429-437, August, 1990 (in Japanese).

[7] J. H. Holland, K. J. Holyoak, R. E. Nisbett, and P. R. Thagard, *Induction: Processes of Inference, Learning, and Discovery*, MIT Press, Cambridge, 1986.

[8] D. Poole, "The effect of knowledge on belief: Conditioning, specificity and the lottery paradox in default reasoning," *Artificial Intelligence*, vol. 49, pp. 281-307, 1991.

[9] H. Ishibuchi, "Fuzzy reasoning method in fuzzy rule-based systems with general and specific rules for function approximation," *Proc. of 8th IEEE International conference on fuzzy systems*, 1999 (in press).

[10] H. Ishibuchi, T. Murata, and I. B. Turksen, "Single-objective and two-objective genetic algorithms for selecting linguistic rules for pattern classification problems," *Fuzzy Sets and Systems*, vol. 89, no. 2, pp. 135-150, 1997.