FUZZY CONTROL SYSTEMS

Edited by

Abraham Kandel

Department of Computer Science and Engineering
University of South Florida, Tampa, Florida

Gideon Langholz

Department of Electrical Engineering
Florida State University, Tallahassee, Florida
and Tel-Aviv University, Israel

With Foreword by Lotfi A. Zadeh

CRC Press
Boca Raton Ann Arbor London Tokyo

Chapter 16
A Self Tuning Method of Fuzzy Reasoning by Genetic Algorithm

This chapter proposes a new self-tuning method for fuzzy reasoning based on the application of a genetic algorithm. Simplified fuzzy inference rules whose consequent parts are expressed in terms of real numbers are employed in this method. Using the genetic algorithm, the number of inference rules and the shapes of the membership functions in the antecedent parts are determined so as to optimize an information criterion. The consequent parts of the inference rules are optimized by the descent method from input-output data. Using the proposed method, the inference rules are determined so as to optimize the learning and generalization capabilities of fuzzy reasoning. Numerical examples are provided to illustrate the effectiveness of the proposed method.

0-8493-4496-8/94/$0.00 + $.50

A SELF-TUNING METHOD OF FUZZY REASONING BY GENETIC ALGORITHM

H. Nomura, I. Hayashi and N. Wakami
Central Research Laboratories,
Matsushita Electric Industrial Co., Ltd. Osaka JAPAN

1. INTRODUCTION

In order to provide fuzzy reasoning with learning function, various learning methods have been proposed. These methods include the self-tuning fuzzy controller employing a descent method[1-3] and the neural network driven fuzzy reasoning[4], which can optimize the shape of membership functions in inference rules from input-output data. However, these methods have suffered from the inherent prerequisite problems, such as an advanced setting of the number of inference rules, which have to be derived by trial and error.

The result of the learning depends on the number of inference rules. When the number of inference rules is small, the inference rules cannot express the input-output relation of given data well. On the contrary, when the number is large, the generalization capability of the inference rules is sacrificed because of the overfitting. Therefore, the number of inference rules has to be determined from a standpoint of overall learning capability and generalization capability. The work to determine the number of inference rules requires to designer a large number of experiments by trial and error.

In order to solve such problem, a new learning method of fuzzy reasoning by means of a genetic algorithm is proposed here. The genetic algorithm[5, 6] is an optimization method developed from the theory of biological evolution.

In this method, a simplified fuzzy reasoning in which the consequent parts are expressed in real numbers is employed. The number of inference rules and the shapes of membership functions in the antecedent part are determined by applying the genetic algorithm, and the real numbers of the consequent parts are derived by using the descent method.

In this chapter, the conventional learning method of fuzzy reasoning employing the descent method, which constitutes the foundation of the proposed method, is described first. Then, an algorithm to determine the structure of the antecedent parts by using the genetic algorithm is explained. Finally, in order to demonstrate the effectiveness of the present method, some numerical examples are reported.

2. A CONVENTIONAL SELF-TUNING METHOD

2.1. Learning Algorithm Using Descent Method

Simplified fuzzy reasoning in which the consequent parts are expressed by real numbers is employed in this method. Expressing input variables by $x_j, (j = 1, \ldots, m)$ and an output variable by y, the inference rules of the simplified fuzzy reasoning can be expressed by the following:

$$\text{Rule } i: \quad \text{If} \quad x_1 \text{ is } A_{i1} \text{ and}, \ldots, \text{ and } x_m \text{ is } A_{im} \quad \text{then} \quad y \text{ is } w_i \tag{1}$$

wherein $i, (i = 1, \ldots, n)$ is the number of the inference rules, $A_{i1}, \ldots, A_{im}$ are the membership functions in the antecedent part, and w_i is a real number in the consequent part. The output of the simplified fuzzy reasoning, y, can be derived by using the following equations:

$$\mu_i = \prod_{j=1}^{m} A_{ij}(x_j) \tag{2}$$

$$y = \frac{\sum_{i=1}^{n} \mu_i \cdot w_i}{\sum_{i=1}^{n} \mu_i} \tag{3}$$

where μ_i is a membership value of i-th inference rule.

By using a descent method, the real numbers w_i of the consequent parts are optimized from the input-output data[1, 2]. The descent method can alter the tuning parameters so as to minimize an objective function H, which is expressed by the following equation in this case:

$$H = \frac{1}{2}(y^{*p} - y^p)^2 \tag{4}$$

where y^{*p} is a desirable output data for the p-th input data $(x_1^p, \ldots, x_m^p)$, and y^p is an output of the fuzzy reasoning corresponding to the same p-th input data $(x_1^p, \ldots, x_m^p)$. The objective function H means the squared inference error.

Using the descent method, the learning rule of the real numbers in the consequent parts can be expressed by the following:

$$\begin{aligned} w_i(t'+1) &= w_i(t') - K \cdot \frac{\partial H}{\partial w_i} \\ &= w_i(t') - K \cdot \frac{\mu_i^p}{\sum_{i=1}^{n} \mu_i^p}(y^p - y^{*p}) \end{aligned} \tag{5}$$

where t' is the number of iteration of learning, μ_i^p is a membership value of the i-th inference rule corresponding to the p-th input-output data, and K is a constant.

By applying this learning rule to the input-output data repeatedly, the consequent parts are updated so as to minimize the objective function. In this case, the result of learning doesn't converge into a local optimum, but into a global optimum, because $\partial^2 H/\partial w_i^2 \geq 0$ is obtained for all i [7].

2.2. Problems of Conventional Self-Tuning Method

The input-output data for learning (Training Data: TRD), and the input-output data for evaluation (Checking Data: CHD), are expressed by the following:

$$\begin{array}{lll} \text{TRD} & : & (x_1^p, \ldots, x_m^p, y^{*p}), \quad p = 1, \ldots, P \\ \text{CHD} & : & (x_1^q, \ldots, x_m^q, y^{*q}), \quad q = 1, \ldots, Q. \end{array}$$

The inference errors for these two types of input-output data are specified respectively by the following equations:

$$E_{TRD} = \frac{1}{P} \sum_{p=1}^{P} (y^p - y^{*p})^2 \tag{6}$$

$$E_{CHD} = \frac{1}{Q} \sum_{q=1}^{Q} (y^q - y^{*q})^2. \tag{7}$$

Figure 1 shows generalized relations between the number of inference rules and the inference errors E_{TRD}, E_{CHD} derived by the optimized inference rules using the descent method. In this method, the larger the number of inference rules, the smaller the inference error E_{TRD} obtained. However, the inference error E_{CHD} becomes larger for a larger number of inference rules after it exceeded a certain threshold value as shown in Figure 1. This phenomenon is caused by an excessive learning applied to the TRD.

The generalization capability of the inference rules can be expressed by the inference error E_{CHD}. Therefore, it can be said that the generalization capability of the inference rules would be lower if an excessive number of inference rules were applied for the learning.

In this conventional self-tuning method, the designer has to search the optimal number of inference rules by trial and error. The work to search the optimal number of inference rules requires the designer to perform a large number of experiments.

The descent method can optimize not only the real numbers in the consequent parts, but also the membership functions in the antecedent parts[1]. For the learning of only the consequent part, the result of learning converges into an global optimum in general. However, the result of learning the antecedent part can converge into a local optimum.

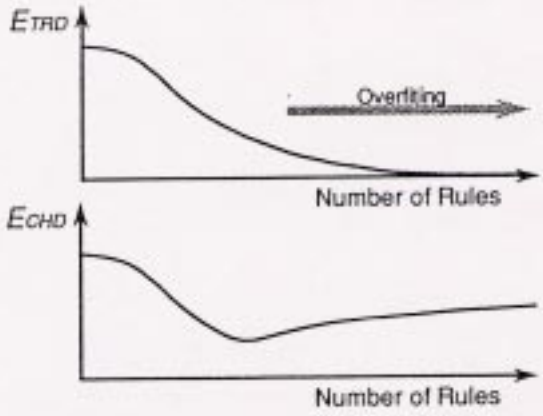

Figure 1: Transition of Inference Errors

3. OPTIMIZATION OF THE INFERENCE RULES BY GENETIC ALGORITHM

The proposed method is to optimize the number of inference rules and the shapes of the membership functions in the antecedent parts by a genetic algorithm.

3.1. Genetic Algorithm

A genetic algorithm (abbreviated GA) [5, 6] is a method to obtain an optimal solution by applying a theory of biological evolution. The most advantageous feature of the GA is a possibility of escaping from local optimum because of probabilistic operations such as *crossover* and *mutation*. In the GA, a solution candidate s_r which maximizes an objective function $E(s_r)$ called *fitness*, is searched. The solution candidate is expressed by the string, called *individual*, which is expressed by the following:

$$s_r = L_{r1} L_{r2} \ldots L_{rG} \tag{8}$$

where $L_{rg}, (g = 1, \ldots, G)$ is a variable taking a value of either "1" or "0". For instance, an example of the individual s_r with $G = 13$ is expressed by the following string:

$$s_r = 1001000110011\,. \tag{9}$$

A set of individuals, S, called *population*, is expressed as follows:

$$S = \{s_1, s_2, \ldots, s_R\}. \tag{10}$$

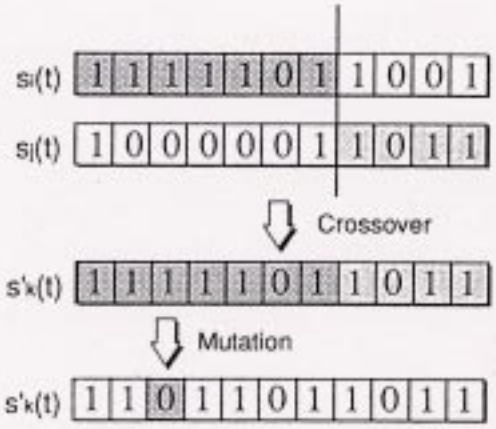

Figure 2: Operators in Genetic Algorithm

3.2. A Learning Procedure of GA

An optimal solution is searched by applying the following procedure:

1) The individuals $s_1, s_2, \ldots, s_R$ which constitute a population $S(t)$ of the 0-th generation ($t = 0$) are determined by uniform random numbers.

2) The fitness $E(s_r)$ for each individual s_r is derived to determine a selection probability $P_{s_r}(t)$, which is expressed by the following:

$$P_{s_r}(t) = \frac{E(s_r(t))}{\sum_{r=1}^{R} E(s_r(t))} \tag{11}$$

where $r = 1, \ldots, R$.

And, then, the number of the subsequently produced individual, k, is initialized at 1.

3) Two individuals $s_i(t)$ and $s_j(t)$ are selected out from the population $S(t)$ in accordance with the selection probabilities $P_{s_i}(t)$ and $P_{s_j}(t)$.

4) An operation called crossover, shown in Figure 2, is applied to the individuals $s_i(t)$ and $s_j(t)$. The crossover operation selects a boundary in the strings with probability of $1/(G-1)$, and exchanges the blocks of strings about the boundary. One of the two individuals produced by this operation is selected at random, and is nominated as the new individual $s'_k(t)$.

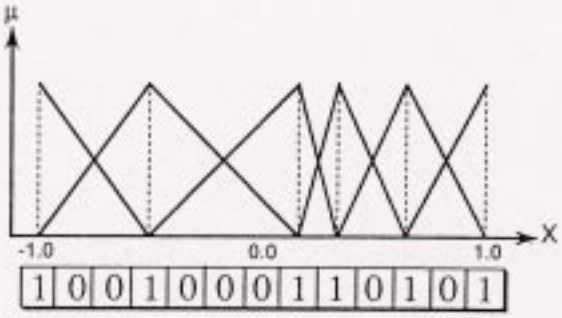

Figure 3: Membership Functions

5) An operation called mutation is applied to $s'_k(t)$. By this, each element of the individual $s'_k(t)$ is reversed according to a mutation probability P_m. Taking the example showing Figure 2, the element positioned third from the left end is reversed by the mutation.

6) The number of newly produced individuals, k, is compared with the total number of individuals, R, and if $k < R$, k is incremented by one, and steps (3) to (6) are repeated. Otherwise, the algorithm proceeds to the next step.

7) The new population, $S'(t) = \{s'_1(t), \ldots, s'_R(t)\}$, produced in steps (3) to (6), substitute into the population on the next generation $S(t+1)$.

8) The generation t is incremented by one, and the steps (2) to (8) are repeated until the terminating conditions are satisfied.

3.3. Algorithm to Optimize Inference Rules

The optimization of the number of inference rules and the shapes of membership functions in the antecedent part by means of the GA is described here.

Figure 3 shows the membership functions in the antecedent part employed in the simplified fuzzy reasoning. The membership function takes triangular shape, and the width of each membership function is defined to be the length between the centers of neighboring membership functions.

The number and shapes of membership functions can be expressed in terms of strings consisting of "0" and "1" as shown in Figure 3, wherein the center position of each membership function is expressed by "1". The string can be expressed as follows:

$$L_{j1}L_{j2}\ldots L_{jG} \quad \text{where} \quad j = 1, \ldots, m. \tag{12}$$

$L_{jr}, (r = 1, \ldots, G)$ is a variable taking a value of either "0" or "1".

In this method, since the string is provided for each of the input variables x_j, a string combining the strings L_{jr} provided for each input variable x_j, which is expressed by $L_{11}...L_{1G}L_{21}...L_{2G}...L_{m1}...L_{mG}$, is considered an individual in the GA. The optimal number of membership functions and the center positions of these are searched for each input variable by the GA.

Assuming the existence of the membership functions on both sides of the domain of each input variable, the following equations could be formulated.

$$\begin{cases} L_{j1} = 1 \\ L_{jG} = 1 \quad \text{where} \quad j = 1, ..., m. \end{cases} \tag{13}$$

As the fitness in the GA, an information criterion[8] shown below is employed here:

$$C = N \cdot \log(E_{TRD}) + 2(\text{The number of parameters}) \tag{14}$$

The information criterion C shows the overall capability for learning and generalization of fuzzy reasoning. A smaller value of the information criterion is considered to mean better inference rules. Expressing the total number of membership functions allocated to the input variable x_j by N_j, the number of parameters shown in Eq. (14) can be expressed by the following:

$$\text{The number of parameters} = \sum_{j=1}^{m}(N_j - 2) + \prod_{j=1}^{m} N_j. \tag{15}$$

The fitness $E(s_r)$ in the GA is defined by the formula:

$$E(s_r) = \max_{r} C(s_r) - C(s_r). \tag{16}$$

In this method, the number of membership functions and the center positions of the membership functions maximizing the fitness $E(s_r)$ are derived by using the GA.

Although the information criterion expressed by Eq. (14) is employed as a fitness in this case, the use of some other objective function such as the unbiasedness criterion could be better depending on the actual problem.

3.4. Self-Tuning Procedure

The procedures to obtain the optimal inference rules using the GA is shown below:

[Step 1] All of the individuals $s_r(t)$ where $r = 1, ..., R$ on the 0-th generation ($t = 0$) are determined by uniform random numbers. In a concrete form, because of the conditions given by Eq. (13), $L_{j2}, \ldots, L_{jG-1}$ values of individuals $s_r(t)$ are set at either "0" or "1" by the uniform random numbers.

[Step 2] The learning by the descent method is applied according to [Step 2-1] to [Step 2-5] in order to determine the real numbers of the consequent parts. These processes have to be applied to all individuals $s_r(t)$, where $r = 1, ..., R$.

To begin with, the number of individual, r, is initialized at 1.

[Step 2-1] The number and the shapes of membership functions in the antecedent parts are determined according to the string of the individual $s_r(t)$. Then, the number of the input-output data, p, is initialized at 1, and the number of iterations of learning by descent method, t', is initialized at 1.

[Step 2-2] The fuzzy reasoning is applied to the p-th input data $(x_1^p, \ldots, x_m^p)$ by using Eqs. (2) and (3), in order to determine the membership value u_i^p of each inference rule and to obtain the output of the fuzzy reasoning y^p.

[Step 2-3] Based on the output of the fuzzy reasoning y^p, the membership value u_i^p and the output data y'^p, the real numbers in the consequent part, w_i, are updated by using Eq. (5).

[Step 2-4] Comparing the number of input-output data, p, with the total number of input-output data, P. If $p < P$, the algorithm is returned to [Step 2-2] after adding 1 to p, otherwise, the algorithm proceeds to next step.

[Step 2-5] In this step, a change of the inference error, $E_{TRD}(t') - E_{TRD}(t'-1)$, is derived. When the change satisfies the following formula, learning by the descent method is terminated:

$$|E_{TRD}(t') - E_{TRD}(t'-1)| < \delta \tag{17}$$

where δ is a threshold value to judge the convergence of the inference error E_{TRD}, which has to be set in advance.

If Eq. (17) is not satisfied, after adding 1 to t' and initializing p to 1, the process is returned to [Step 2-2].

If Eq. (17) is satisfied, the fitness $E(s_r)$ is derived from the values of converged inference errors E_{TRD} by applying Eqs. (14) to (16), and the selection probability $P_{sr}(t)$ are derived by Eq. (11). Then, if $r < R$, the algorithm is returned to [Step 2-1] after adding 1 to r, otherwise, the number of the individual produced next, k, is initialized at 1, and the algorithm proceeds to next step.

[Step 3] Two individuals $s_i(t)$ and $s_j(t)$ are selected from $S(t)$ according to the selection probabilities $P_{si}(t)$ and $P_{sj}(t)$.

[Step 4] The crossover is applied to these two selected individuals to derive a new individual $s'_k(t)$. In this case, since the fuzzy reasoning has m input variables, m-points crossover[5] is applied.

[Step 5] The mutation is applied to each of the elements of individual $s'_k(t)$ according to the mutation probability P_m.

[Step 6] The series of processes, [Step 3] to [Step 5], is repeated until the number of new individual , k, becomes R.

[Step 7] A new population $S'(t) = \{s'_1(t), s'_2(t), \ldots, s'_R(t)\}$ produced by the processes from [Step 3] to [Step 6] is defined as $S(t+1)$.

[Step 8] The number of the generation, t, is incremented by one, and the processes from [Step 2] to [Step 8] are repeated until a convergence of the population S is obtained.

An individual having the highest fitness in the converged population is defined as the final solution.

4. NUMERICAL EXAMPLES

In order to demonstrate the usefulness of the GA method, two simple identification problems of nonlinear systems are discussed here.

The two nonlinear systems having single input and single output are shown below:

$$\text{SYSTEM 1} \ : \ y = \begin{cases} -x & x \leq 0 \\ x^2 & x > 0 \end{cases} \tag{18}$$

$$\text{SYSTEM 2} \ : \ y = 1 - x - x^3 + x^5 + R_g \tag{19}$$

where R_g is a function that generates the random numbers with gaussian distribution having an average value of 0.0 and a standard deviation of 1.0.

By changing the input x in Eqs. (18) and (19) in the range [-1, 1] according to the uniform random numbers, input-output data are generated respectively. The produced input-output data are divided into the TRD and the CHD in each system.

By using the GA method, each of the input-output relations given by Eqs. (18) and (19) is identified in terms of fuzzy inference rules from the produced TRD.

The initial conditions for this GA method are shown in the following:

The total number of individuals	:	$R = 20$,
The length of individual	:	$G = 13$,
The mutation probability	:	$P_m = 0.01$,
The number of generations	:	$t = 50$,
The threshold value	:	$\delta = 1.0 \times 10^{-5}$.

Figure 4 shows the input-output relation of SYSTEM 1. In this example, 40 TRD and 40 CHD are used. The square dots in Figure 4 show the TRD.

Figure 5 shows the membership functions obtained by applying the GA to the TRD of SYSTEM 1. The number of inference rules is determined at 7 by the GA. In the region ($x > 0$) having non-linear input-output relation, more inference rules are assigned than in the region ($x \leq 0$) having linear input-output relation. This result indicates that this self-tuning method can arrange the membership functions in accordance with the nonlinearity of the given TRD.

Figure 6 shows the output of fuzzy reasoning using the optimized membership function of Figure 5. In Figure 6, the output of fuzzy reasoning coincides with the true output of Eq. (18) approximately. Therefore, it can be said that this self-tuning method acquires the input-output relation of given TRD well. Using the inference rules obtained by the GA, the E_{TRD} and E_{CHD} became 2.1×10^{-6} and 5.7×10^{-6}, respectively.

In order to show the higher learning capability of this method, a comparison with a conventional self-tuning method was conducted. As the conventional method, the self-tuning controller by the descent method[1] only was employed, and the inference rules were constructed from the same TRD.

In Figure 7, relations of the conventional method between the number of inference rules and the inference errors E_{TRD}, E_{CHD} are shown. The numbers of inference rules in the conventional method were varied from 2 to 27 manually. Figure 7 proves that a lower inference error E_{TRD} is obtained for a higher number of inference rules, while a higher inference error E_{CHD} is obtained for a higher number of inference rules exceeding a certain threshold. As shown in Figure 7, the number of inference rules which gives a minimum E_{CHD} is 7. The values of E_{TRD} and E_{CHD} at 7 inference rules are 3.0×10^{-5} and 6.3×10^{-5}, respectively, both of which are larger than those obtained by the GA. This shows that the inference rules obtained by the GA have a higher learning capability than those obtained by the conventional self-tuning method.

In order to demonstrate that the solution obtained by the GA doesn't converge into a local optimum but to a global optimum, a result of searching in total space is shown next. In this example, the total number of solution candidates is $2^{11}(= 2048)$. The learning of the consequent parts by the descent method is applied to all of the 2048 solution candidates in order to derive the value of the information criterion C. The results are shown in Table 1, where they are sorted in terms of the value of the information criterion. Table 1 shows that the optimal solution obtained by the total search is 1000001111101. This solution coincides with the one obtained by the GA. This example shows that the optimal inference rules minimizing the value of the information criterion is indeed obtained by the GA.

The result of learning applied to SYSTEM 2 is shown next. It is very often that noise component is included in observed input-output data. This example simulates the case where input-output data include noise component. Through this example, the generalization capability of the GA method is shown.

Figure 8 shows the input-output relation of SYSTEM 2, the thick line shows

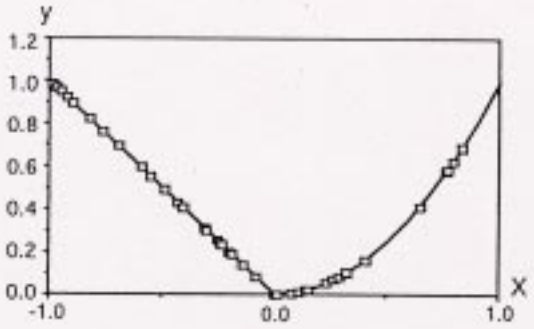

Figure 4: Input-output Relation (SYSTEM 1)

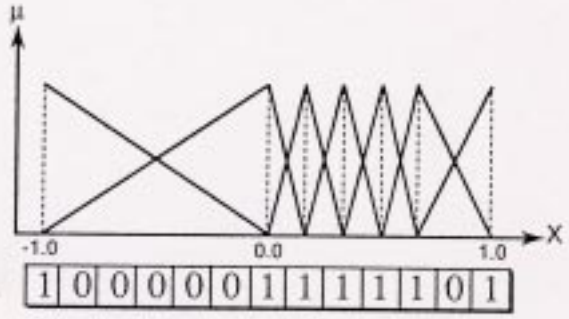

Figure 5: Optimized Membership Functions (SYSTEM 1)

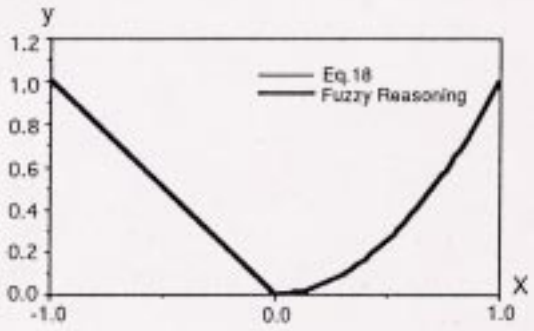

Figure 6: Output of Fuzzy Reasoning (SYSTEM 1)

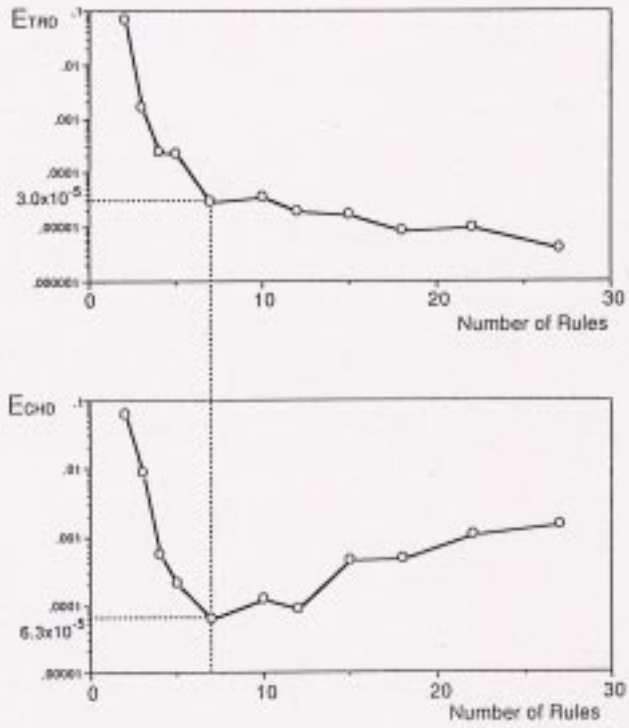

Figure 7: Transition of Inference Errors (SYSTEM 1)

Table 1: Result of Searching in Total Space (System 1)

No.	Strings	C
1	1000001111101	-496.9
2	1000011111101	-494.7
3	1000101111101	-494.3
4	1001001111101	-493.8
5	1010001111101	-493.5
6	1100001111101	-493.1
7	1000001111111	-490.7
8	1001011111101	-490.7
	.....	
2048	1000000000011	-94.0

the true input-output relation without random numbers, the square dots show the TRD. Since a term having random numbers is included in Eq. (19), a noise component is included in the produced TRD and CHD. In this example, 400 TRD and 400 CHD are used.

If the conventional self-tuning method were applyed for such TRD by giving a large number of inference rules, the generalization capability of the inference rules could be particularly low. Therefore, the searching of an optimal number of inference rules become more important in such case.

Figure 9 shows the membership functions obtained by applying the GA to the TRD of SYSTEM 2 . The number of inference rules is determined at small number 4. This result shows that the number of inference rules is determined so as to avoid overfitting by the GA.

Figure 10 shows the output of the fuzzy reasoning optimized by the GA, the thick line means the output of the optimized fuzzy reasoning, the thin line shows the true output without random numbers. Since Figure 10 shows that the output of the fuzzy reasoning is not constrained by the noise of the TRD, and coincides with the true output approximately, it can be said that the GA method has high generalization capability.

The results of a total search similar to the one carried out for SYSTEM 1, are shown in Table 2. The optimum solution obtained by the total search coincides with the solution 1000010001001 obtained by the GA. This example shows also that optimum inference rules can be obtained by the GA.

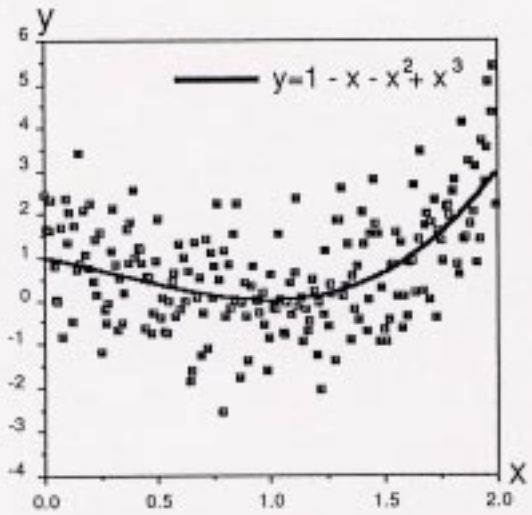

Figure 8: Input-output Relation (SYSTEM 2)

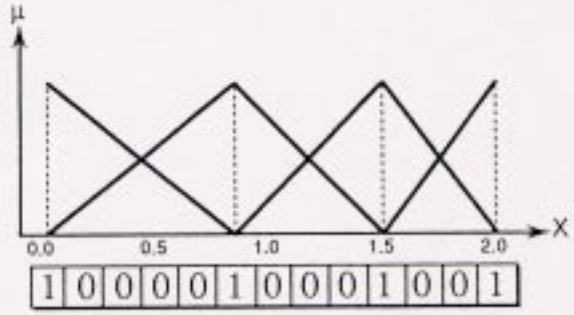

Figure 9: Optimized Membership functions (SYSTEM 2)

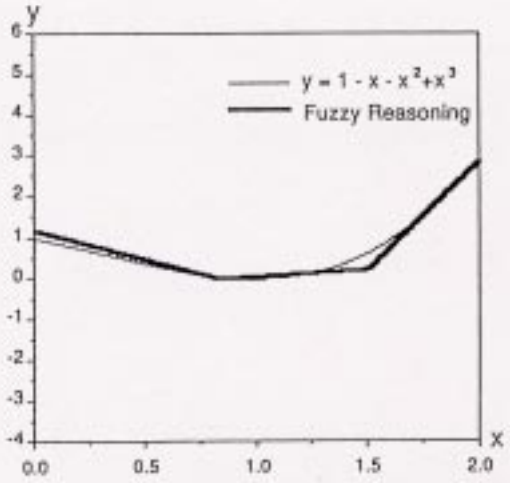

Figure 10: Output of optimized fuzzy reasoning (SYSTEM 2)

Table 2: Result of Searching in Total Space (SYSTEM 2)

No.	Strings	C
1	1000010001001	29.2
2	1000000100101	31.0
3	1000100001001	32.4
4	1100010001001	33.3
5	1000010001101	33.4
6	1000100010101	33.7
7	1000001001101	33.7
8	1010010001001	33.8
	.....	
2048	1000000000001	391.4

5. CONCLUSION

A new self-tuning method of fuzzy reasoning by the genetic algorithm is proposed here. With this method, the number of inference rules and the shapes of membership functions in the antecedent part are determined by the genetic algorithm so as to optimize an information criterion expressing the quality of the inference rules.

Using two numerical examples, we have shown that this GA method has the higher learning capability than the conventional self-tuning method using the descent method.

As a future problem, the effectiveness of the proposed method, when it is applied to more complicated systems having multi-inputs, should be established.

REFERENCES

[1] Nomura, H., Hayashi, I., Wakami, N., "A self-tuning method of fuzzy control by descent method". *Proceedings of 4th IFSA Congress*, Engineering, pp. 155-158, 1991.

[2] Nomura, H., Hayashi, I., Wakami, N., "A learning method of fuzzy inference rules by descent method". *Proceedings of IEEE International Conference on Fuzzy Systems*, pp. 203-210, 1992.

[3] Nomura, H., Hayashi, I., Wakami, N., "A self-tuning method of fuzzy reasoning by delta rule and its application to a moving obstacle avoidance". *Journal of Japan Society for Fuzzy Theory and Systems*, Vol. 4, pp. 379-388, 1992. (in Japanese)

[4] Hayashi, I., Nomura, H., Yamasaki, H., Wakami, N., "Construction of fuzzy inference rules by NDF and NDFL". *International Journal of Approximate Reasoning*, Vol.6, pp. 241-266, 1992.

[5] Goldberg, D. E., *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison Wesley Publishing Company, 1989.

[6] Holland, J. H., *Adaptation in Natural and Artificial Systems* , The University Michigan Press, 1975.

[7] Ichihashi, H., Watanabe, T., "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, 1990. (in Japanese)

[8] Akaike, H., "A new look at the statistical model identification". *IEEE Transactions on Automatic Control*, AC-19, No. 6, 1974.

[9] Nomura, H., Hayashi, I., Wakami, N., "A self-tuning method of fuzzy reasoning by genetic algorithm". *Proceedings of the 1992 International Fuzzy Systems and Intelligent Control Conference*, pp. 236-245, 1992.

[10] Nomura, H., Hayashi, I., Wakami, N., "A learning method fusing fuzzy reasoning and genetic algorithm". *Proceedings of the IMACS/SICE International Symposium on Robotics, Mechatronics and Manufacturing Systems*, pp. 155-160, 1992.