An Algorithmic Approach for Fuzzy Logic Application to Decision-Making Problems

결정 문제에 대한 퍼지 논리 적용의 알고리즘적 접근

C. J. Kim*
김 창 종*

※이 논문은 1995년도 한국과학재단의 연구비에 의하여 연구되었음

ABSTRACT

In order to apply fuzzy logic, two major tasks need to be performed: the derivation of fuzzy rules and the determination of membership functions. These tasks are often difficult and time-consuming. This paper presents an algorithmic method for generating membership functions and fuzzy rules applicable to decision-making problems; the method includes an entropy minimization for clustering analog samples. Membership functions are derived by partitioning the variables into desired number of fuzzy terms, and fuzzy rules are obtained using minimum entropy clustering. In the rule derivation process, rule weights are also calculated. Inference and defuzzification for classification problems are also discussed.

요 약

퍼지 논리를 적용하기 위해서는 두가지 과제가 이루어져야 하는데 그것은 퍼지룰의 유도와 멤버쉽함수의 결정이다. 이 과제는 어렵고 또한 시간을 요하게 된다. 본 논문에서는 문제에 적용 가능한 멤버쉽함수와 퍼지룰을 자동으로 유도하기 위한 알고리즘적 방법을 제시하고 있다. 이 알고리즘적 방법은 샘플을 구분하는 엔트로피 최소화의 원리에 입각하고 있다. 멤버쉽함수는 샘플을 연속적으로 구분하여 이루어지며 퍼지룰 또한 엔트로피 최소화 원리에 의하여 이루어진다. 퍼지룰의 유도에서는 룰 비중 또한 같이 계산된다. 결정 문제에 적용을 위한 추론법및 방법도 논의되었다.

I. Introduction

Fuzzy logic has been applied with reasonable success to many control problems for which only conventional control methods had previously been utilized. In such control problems, the value of fuzzy logic is found in that vague meanings and relationships, expressed in ordinary language, can be effectively formulated. The fuzzy inference procedure includes the translation of an analog value into membership grades which are

*수원대 전기전자정보통신공학부

defined by the membership functions of fuzzy terms.

Although fuzzy logic theory was introduced in the 1960s, its application to industrial control emerged in the early 1970s with a procedure for the control of a steam engine[1]. Since then, fuzzy logic has been applied in other control areas. Currently, fuzzy logic is involved in many industrial and commercial applications and home appliances. Even in decision-making problems such as power fault identification which involves not-well-defined conditions and seemingly unrelated parameters, fuzzy logic is being applied [2, 3]. To apply fuzzy logic, we must define fuzzy rules, fuzzy terms, and membership functions. It is often difficult and time-consuming to derive these rules and membership functions; by devising an automatic procedure for deriving membership functions and fuzzy rules, therefore, we can make fuzzy logic applications much easier.

An estimation model for fuzzy membership functions was introduced using fuzzy ensemble membership apportionment learning estimators [4]. However, this model estimates only membership functions and does not produce fuzzy rules. Recently, a "table-lookup" scheme for fuzzy rule generation for numerical input-output pairs was suggested [5]. This scheme, which aims to extract a rule for each input-output pair, however, determines the partitions of the domain interval and membership functions in an ad hoc manner. AI and neural network technique have also been applied to extract fuzzy rules from numerical data, however, they require the number of divisions in the input variable be defined in advance[6].

Clearly, an automatic process which can generate both membership functions and fuzzy rules directly from experienced sample data would be of considerably more value. Advanced applications such as learning fuzzy control need an adaptive method of representing fuzzy knowledge, so an attempt to automate fuzzy logic applications is a timely response to an important subject. The primary objective of this paper is to develop an algorithm which is capable of automating fuzzy logic applications in classification and decision-making problems. Using an algorithmic approach which utilizes the concept of entropy minimization, membership functions are generated and, fuzzy rules with rule weights are determined.

II. Formation of Structured Longuistic Variable

In fuzzy logic applications, membership functions have typically been determined by human experts. Accordingly, experience and common-sense are the two leading guidelines for determining the membership functions. Similarly, fuzzy rules have been devised from expert opinion. The fundamental problem with this approach is that the rules derived by the expert using experience and common sense are not always most suitable for an automatic controller. Furthermore, there is no way to assess whether the rule correctly and optimally represents the experienced sample data. Therefore, we introduce an algorithmic approach which, without human intervention, can be utilized for fuzzy logic applications. Guided by a theorem of maximum information extraction, this algorithmic approach generates membership functions and fuzzy rules from experienced sample data.

The automation of membership function derivation can be considered as an attempt to draw a structured linguistic variable in which fuzzy terms and their meanings can be characterized by an algorithm. One of the basic tools for fuzzy logic is the linguistic variable, i.e., a variable whose values are not numbers but words in a natural or artificial language[7]. A linguistic variable is characterized by a quintuple $(r, T(r), U, G, M)$,
where,

r: the name of the variable, its "label" or, sometimes, its value R

$T(r)$: the term set of r, that is, the set of names in r

U: the range of $T(r)$

G: a syntactic rule for generating R, the values of r

M: a semantic rule for associating each R with its meaning

A particular R, that is a name generated by G, is called a *term*. For example, if a linguistic variable r is defined with the label "age" in $U=[0, 100]$, then the terms of this linguistic variable, generated by the rule $G(r)$, could be called "old", "middle", "young", and so on. Therefore, $T(r)$ defines the term set of the variable r with $T(\text{age})=\{\text{old, middle, young}\}$. $M(r)$ is the rule that assigns meanings to these terms. A linguistic variable r is called *structured* if the term set $T(r)$ and the meaning $M(r)$ can be characterized algorithmically. For a structured variable, $M(r)$ and $T(r)$ can be regarded as algorithms which generate the terms and the meanings associated with them.

The above description of a linguistic variable can be rephrased as follows: $G(r)$ determines the fuzzy terms from a variable and $M(r)$ determines the membership functions of the fuzzy terms. Once the number of fuzzy terms is decided, the only unknown item in the linguistic variable is the rule $M(r)$. The algorithmic approach in this paper decides the rule for membership function formation; in a theoretical sense, therefore, one object of this paper is to draw a structured linguistic variable.

In industrial control application of fuzzy logic, a set of terms drawn from linguistic variables has been used to describe the states of the process. In particular, the error value and the change in error value are quantized into a number of points covering the range in U, and the values are then assigned as grades of membership in 3, 7, or another number of subsets[8]. However, the algorithmic approach of this paper has some constraints in the number of fuzzy terms; it generates fuzzy terms in power of 2's, i.e., 2, 4, 6, 8, and so on. Therefore, in this paper, the following 8-term set will be generated: PB, Positive Big; PM, Positive Medium; PS, Positive Small; PZ, Positive Zero; NZ, Negative Zero; NS, Negative Small; NM, Negative Medium; and NB, Negative Big. This approach, in addition, provides an automatic mechanism for generating fuzzy rules from the term set $T(r)$ and the meaning $M(r)$.

Ⅲ. Entropy Concept in Classification

The main idea behind the automatic generation of membership functions and fuzzy rules in decision-making and two-class identification problem is the concept of clustering. Using the entropy principle, parameter values in the sample data can be clustered. We first discuss the entropy concept by considering a classification of two-class ("true" and "false") samples. When we look at the samples in the "true" class, for example, we try to discover what it is that makes them "true". In other words, we try to find similarities among the parameters for "true" cases, which distinguish them from samples which are "false". This means that we try to find attributes or groups of attributes possessed by "true" samples and not by "false" samples. These attributes or groups of attributes then become part of the boundary separating the "true" samples from the "false" samples. To optimally separate "true" and "false" samples, we usually use a measure of information. The quantity of information, gain or loss, is a basic element for entropy calculation for clustering.

The main purpose of entropy minimization analysis in information theory is to determine the gain or loss of information in a given data set. This information quantity compares the contents of available data to some prior state of expectation. The higher one's prior estimate of the probability for an outcome, the lower the information gained by observing its occurrence. In general, the more probable the event is on the basis of what we already know, the lesser the information content is if and when the event occurs. In other words, when information gain is minimized, we reach at an optimal point for predicting the occurrence. A quantity of information is defined as proportional to the negative of the logarithm of probability [9].

If we assume that the probability that the i^{th} sample W_i is true is $P(W_i)$, and if we actually observe the sample W_i in the future and discover that it is true, then we gain the information,

$$I(W_i) = -k \ln P(W_i)$$

If we discover that it is false, on the other hand, we still gain the information,

$$I(W_i) = -k \ln [1 - P(W_i)]$$

Entropy is defined as the expected value of information. Thus, the expected value of the information to be gained by observing W_i can be expressed as follows (with $P_i = P(W_i)$:):

$$S(W_i, \sim W_i) = -k\,[P_i \ln P_i + (1-P_i) \text{ in } (1-P_i)]$$

To illustrate this concept we first discuss the entropy equation for sample clustering. Assume that we are seeking a threshold value for samples in the range of $[x_{min}, x_{max}]$ for a two-class problem (See Figure 1). By moving an imaginary threshold value x between x_{min} and x_{max} we can calculate the entropy for each value of x for p region $[x_{min}, x_{max}]$ and q region $[x_{min}, x_{max}]$, which is[10]:

$$S(x) = p(x) S_p(x) + q(x) S_q(x) \tag{1}$$

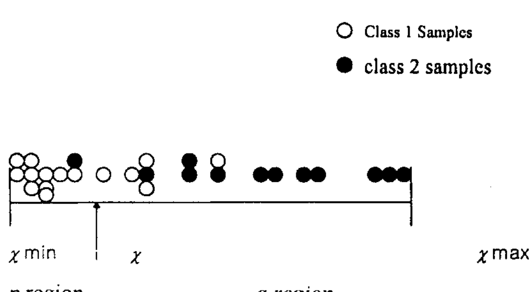

Fig. 1 Illustration of Threshold Value Calculation

where,

$p(x)$ is the probability that all samples are in the p region;

$q(x)$ is the probability that all samples are in the q region; and

$$p(x) + q(x) = 1.$$

Entropies of the p and q regions, $S_p(x)$ and $S_q(x)$, can be expressed by:

$$S_p(x) = -(p_1(x) \ln p_1(x) + p_2(x) \ln p_2(x)), \tag{2}$$

$$S_q(x) = -(q_1(x) \ln q_1(x) + q_2(x) \ln q_2(x)) \tag{3}$$

where,

$p_k(x)$ is the probability that the class k sample is in the p region, and

$q_k(x)$ is the corresponding conditional probability for the q region.

We calculate the entropies of equations (2) and (3) using relatively unbiased estimates of $p_k(x)$, $q_k(x)$, $p_k(x)$, $q_k(x)$. The relatively unbiased estimates for $p_k(x)$ and $p(x)$ are:

$$P_k(x) = \frac{n_k(x) + 1}{n(x) + 1} \tag{4}$$

$$P(x) = \frac{n(x) + 1}{n + 1} \tag{5}$$

where,

$n_k(x)$ = the number of class k samples located in the p region,

$n(x)$ = the total number of samples located in the q region, and

n = the total number of samples in the p and q regions.

Equations for $q_k(x)$ and $q(x)$ can be derived similarly. Using the estimates and the entropy equation, we calculate the entropy for each value of x. In reality, entropy S is a measure of the "fuzziness" of the outcome probability P_i. Also, from the reference

[9], it was shown that low level entropy points in the direction of a more nearly perfect sorting of events based on outcomes. In other words, the entropy minimum point divides two events so that the classification probability can be higher. This also means that the entropy point is really "fuzzy" in sorting of the two events. A value of x whose entropy is the minimum, $X = S_{min}(x_{min}, x_{max})$, is the optimal threshold value in the range of $[x_{min}, x_{max}]$, and this point also indicates the fuzziest point in two different classes.

Optimal division of the sample space will yield fuzzy terms for each parameter; the partitioning points (the entropy minimum points) and the in-between point decides the range of the membership functions. Using the same clustering method but with binary parameter values, fuzzy production rules will be drawn. Because the rule extraction process is performed over each individual fuzzy terms, the final production rule will consist of the integration of independent rules. The entropy of a set of possible outcomes of a trial in which one and only one outcome is true is expressed as the summation of the products of all probabilities and their logarithms. Therefore the entropy of all the samples is expressed by

$$S = -k \sum_{i=1}^{N} [P_i \ln P_i + (1-P_i) \ln (1-P_i)] \tag{6}$$

This entropy is smallest when the amount of information that we can expect to gain from further observation is least. Therefore, given all available information, it is possible to cluster using the minimum entropy principle. In entropy minimum state, all of the information has been extracted from the available sample data. This observation is very important to the algorithmic approach: when samples are the only source of information, maximum extraction of information is essential for an automated process. Therefore, in classification problems, the entropy principle is a useful tool for optimal clustering.

The clustering point in samples is called a threshold value between classes. The clustering point, the threshold value point, is actually the fuzziest point between two clusters. The threshold point indicates the overlapping region of two sets or two clusters, and the center point between two threshold point may indicate the most representative value for each cluster or set. Therefore, the regions may be divided by the threshold points and the in-between points and it will yield the fuzzy term region. Membership functions are shaped from the threshold and the in-between points. If we sub-divide once-clustered samples using the same entropy principle, we can sub-cluster the samples. Further fuzzy terms will be resulted in the sub-clustered region.

Ⅳ. Membership Function Generation

Using the entropy equations (2) and (3) with the estimates given by (4) and (5), we calculate the entropy for all the x 's in the sample value space. A value of x which yields the minimum entropy is taken to be the threshold value of the two fictitious partitions. We indicate this first threshold by X_1. This threshold value is calculated in the range of x_{min}, the minimum sample value, and x_{max}, the maximum sample value. If we replace the variables x_{min} and x_{max} by X_{01} and X_{02}, respectively, then we can indicate X_{11} by $X_{11} = S_{min}(X_{01}, X_{02})$, where $S_{min}(a, b)$ indicates the minimum entropy point in the $[a, b]$ region. The left side of the primary threshold may be called the negative side and the right side, the positive side. The threshold point, X_{11}, is the fuzziest point in that this point represents the center of the two clusters; the threshold point partially overlaps the two classes. Therefore, we simply find the middle points, one in between X_{11} and X_{02}, and the other in between X_{01} and X_{02}, and assign 1.0 degree to the two middle points. And proportional degree can be obtained by drawing two diagonal lines, one from $((X_{11}+X_{02})/2, 1.0)$ to $((X_{11}+X_{01})/2, 0.0)$ and the other from $((X_{11}+X_{01})/2, 1.0)$ to $((X_{11}+X_{02})/2, 0.0)$. Hence, there appear two fuzzy terms of NG(negative) and PO(positive) with trapezoid-shaped membership functions (see Figure 2 (a)):

$[X_{01}, (X_{11}+X_{02})/2]$: NG
$[(X_{11}+X_{01})/2, X_{02}]$: PO

We can draw another threshold point to sub-divide each side more precisely. Using the same procedure for entropy calculation, we can compute secondary threshold values from the positive and negative sides, $X_{21}=S_{\min}(X_{01}, X_{11})$ and $X_{22}=S_{\min}(X_{11}, X_{02})$, respectively. X_{21} is the fuzziest point in the region of NG (negative), and X_{22} the fuzziest point in PO(positive). Assuming that we have trapezoid-shaped functions of the fuzzy terms for both ends and triangular-shaped ones for the others, we have the following four fuzzy terms of NG(Negative), NZ(Negative Zero), PZ(Positive Zero), and PO(Positive). These four terms are illustrated in Figure 2(b).

$[X_{01}, (X_{21}+X_{11})/2]$: NG
$[(X_{21}+X_{01})/2, (X_{11}+X_{22})/2]$: NZ
$[(X_{11}+X_{21})/2, (X_{22}+X_{02})/2]$: PZ
$[(X_{11}+X_{21})/2, X_{02}]$: PO

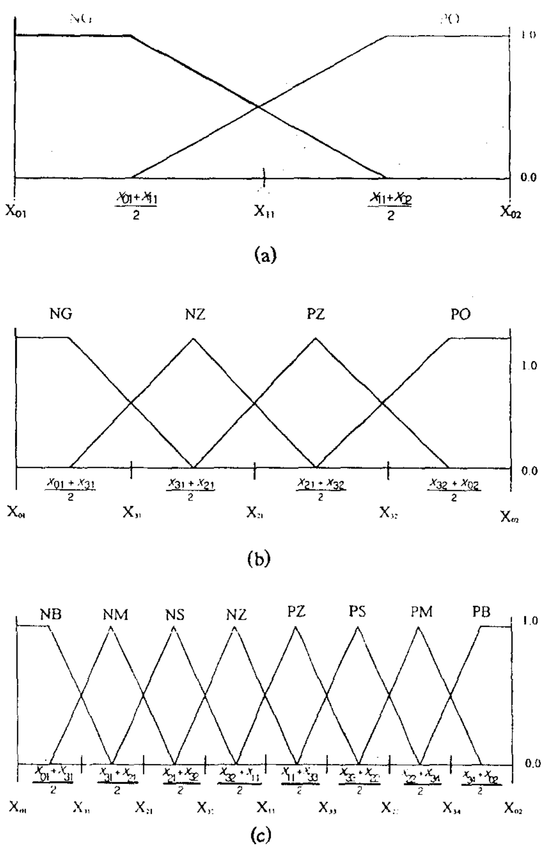

Fig. 2 Illustration of Membership Function Determination by Entropy Minimum Points

If we proceed one more level of threshold calculation in the clustered region, we obtain the following tertiary threshold values. Each tertiary threshold value and its range for minimum entropy search is shown below. The threshold finding for each level is illustrated in Figure 3.

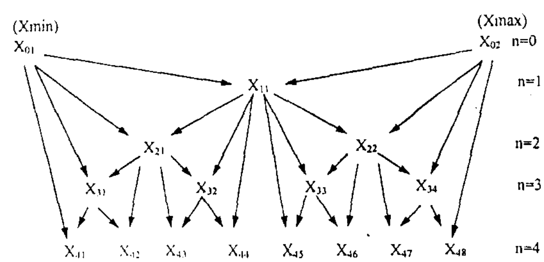

Fig. 3 The levels of Threshold Value Calculation

$X_{31}=S_{\min}(X_{01}, X_{21})$
$X_{32}=S_{\min}(X_{21}, X_{11})$
$X_{33}=S_{\min}(X_{11}, X_{22})$
$X_{34}=S_{\min}(X_{22}, X_{02})$

The areas of the 8 fuzzy terms are indicated below:

$[X_{01}, (X_{31}+X_{21})/2]$: NB
$[(X_{31}+X_{01})/2, (X_{21}+X_{32})/2]$: NM
$[(X_{31}+X_{21})/2, (X_{32}+X_{11}))/2]$: NS
$[(X_{31}+X_{21}/2, (X_{33}+X_{11})/2]$: NZ
$[(X_{32}+X_{11})/2, (X_{33}+X_{22})/2]$: PZ
$[(X_{33}+X_{11})/2, (X_{34}+X_{22})/2]$: PS
$[(X_{33}+X_{22})/2, (X_{34}+X_{02})/2]$: PM
$[(X_{22}+X_{34})/2, X_{02}]$: PB

Again, assuming trapezoidal shapes in both ends and triangular shapes for the others, we have total 8 membership functions determined by mechanically connecting the center points between two adjacent threshold value points as shown in Figure 2(c).

Also, we may draw the following relationship between the threshold level and the number of fuzzy terms (and membership functions):

$$L = 2^n \tag{7}$$

where

L is the number of fuzzy terms, and

n is the threshold level (1 for primary, 2 for secondary, 3 for tertiary, and so on).

These membership functions are not realistic and must be interpreted in care. In other words, the membership functions do not give a true picture of the real situation. PZ or NZ, for instance, does not correspond to the true 0 value of the input; reality and expert opinion are totally ignored for the determination of the membership functions. This irrelevance, however, does not lead to any problem for rule generation or inference. In reality, membership functions are meaningful only when they accurately represent the sample data from which the rules are derived.

V. Fuzzy Rule Geneation

A. Rule Extraction Principle

Fuzzy rule is to relate input and output variables. Since the entropy minimization principle has been proven effective for decision rule derivation with binary values [9], we will discuss the rule derivation with binary value first, then, the transform of the sample value into binary number for each fuzzy term. The rules for two-class problems will be generated from the acquired 8 fuzzy terms using the entropy principle.

Along with the entropy calculation, there is the problem of assigning a probability in cases where only one digit (or variable) has been observed "true" on z of n occasions. What makes it difficult to assign a probability is the feeling that what is observed is more likely than what is not, and that what is observed more often is more likely than what is observed less often. This probability can be expressed as

$$P = \lim_{n \to \infty} \frac{z}{n}$$

As n becomes larger and larger, z/n comes closer and closer to P. But it is not clear in what sense z/n is approaching a limit, which we presume to exist, and call P. In such cases it is possible to use the mean probability, $\overline{P}$ to represent P. Mean probability in the class separation is defined by[9].

$$\overline{P} = \frac{z+t}{n+t+f} \tag{8}$$

where,

t is the number of distinguishable "true" states, and

f is the number of distinguishable "false" states.

This mean probability, when there are only two classes ($t=1$ and $f=1$), becomes,

$$\overline{P} = \frac{z+1}{n+2} \tag{9}$$

The mean probability is used in the entropy equation for rule derivation and in rule weight calculation.

Fuzzy rule generation will choose an optimal rule from numerous candidates. This method involves finding a partition (or cluster separation) of feature space for which entropy S, the expected value of the conditional classification entropy, is a minimum. The entropy of a rule for a fuzzy term, using the mean probability of equations (8) or (9), is

$$S = -k \sum_{i=1}^{m} Y_i \, \overline{P}_i \ln \overline{P}_i \tag{10}$$

where,

m is the total number of steps, i.e., the total number of separated clusters,
Y_i is the number of samples covered by step i, and
k is a constant.

Theoretically, therefore, we check all the combinations of separating two classes and calculate their entropy to select one combination (a rule) whose entropy is smallest. For n parameters in binary number, for instance, there are $N=2^n$ combinations. Furthermore, there are N^2 ways of separating N numbers into two classes; if we have, for example, 15 parameters in binary numbers, there are $21^{15}=32768$ different ways. Even allowing that we usually do not have that many samples, we still have too many combinations to investigate. Therefore, a practical way to apply the entropy principle for production rule derivation is obviously needed.

B. Simplified Method for Rule Extraction

To simplify the extraction of rules, we investigated the relationships of the variables in the entropy equations. From equation (10), we can see that the closer $\overline{P}$ is to 1 or 0, the smaller the entropy S is. Also, from equations (8) and (9), it is apparent that the bigger z is, the bigger $\overline{P}$ is. Therefore, if we can find the biggest z, we can find the rule with minimum entropy: to find the biggest z, we use the concept of *digit index*. The digit index is defined as the ratio of correct separation of two classes using only a single digit of parameter (or feature). In other words, the index is a measure to select a digit which assures the minimum number of wrong classifications.

For digit index determination, we first calculate a quantity called the *digit count*. We count the number of 1's in the class 1 samples and the number of 0's in the class 2 samples. Then, we divide each number by the total number of samples in each class. The result is digit count, d. If we have n digits (or features), then, by this calculation, we can have n digit counts; d_1 through d_n. Next, we add all the digit counts of each class. If this value is close to 1, the digit (or feature) is not important for separation: the 1's and 0's have the same weight in both classes. If the value is not close to 1, this means that there are less 1's or 0's in a class. We formalize this idea by defining *digit* (or *feature*) index as follows:

$$I_n = \left| \sum d_n - 1 \right| \tag{11}$$

The digit (or feature) whose digit index is the maximum is the best separating point in rule extraction. We separate samples accordingly and eliminate those samples which were separated by the digit. If we chose the rule as "1xx for class 1" (x indicates "don't care"), then we delete all the samples (of both classes) whose first digit value is 1. We repeat this sequence of

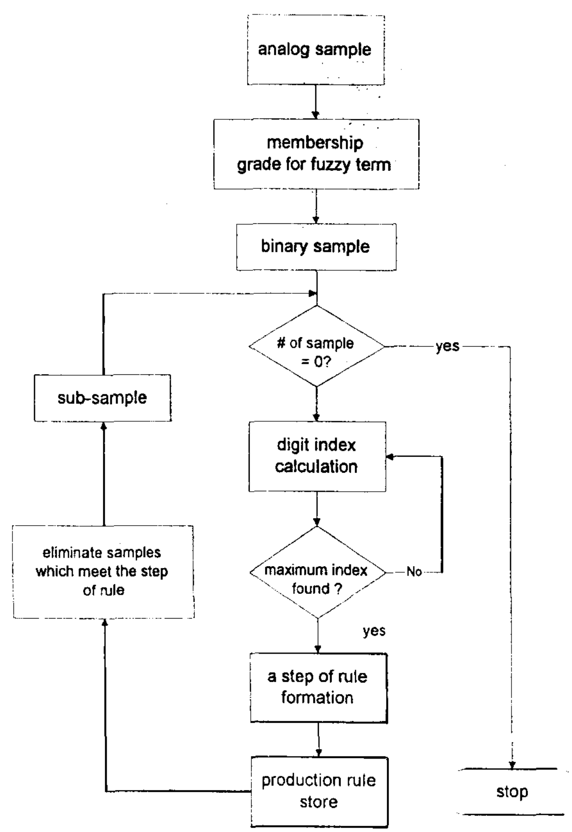

Fig. 4 Flow Chart for Fuzzy Rule Derivation

digit index calculation, formation of the rule, and elimination of the samples which satisfy the rule, until all the samples are accounted. Figure 4 shows the flow chart of this simplified rule generation procedure.

C. Illustration of the Rule Extraction

Fuzzy rule generation is performed separately for each fuzzy term. Therefore, we check the samples if they are the member of a particular fuzzy term, for example PB. If they are the member, then they are transformed into binary number 1s; otherwise, 0s for fuzzy term PB. This check and transformation process will be done for other fuzzy terms independently. Each fuzzy term rule resemble a decision tree in which the branch points indicate the divided search route. Each r fuzzy term rule has the form of "If..., else if..., else if..., end if". Therefore, the fuzzy rule generated by this method is somewhat different from the conventional one which includes as many fuzzy terms as chosen (8 in our case) in a single rule.

As an illustration for fuzzy rule extraction, we use data adopted form [10] for an imaginary fault identification situation. We have fifteen 3-variable samples to be classified into two classes. The sample data is shown below:

sample #	v1	v2	v3	class
1	0.210	1.477	2.420	1
2	0.180	1.435	5.012	1
3	0.203	1.184	5.245	1
4	0.106	1.154	6.012	1
5	0.202	1.057	7.034	1
6	0.185	-0.673	4.992	1
7	-0.170	4.628	3.420	2
8	0.724	1.114	5.940	2
9	0.035	3.944	5.120	2
10	0.167	4.262	3.420	2
11	0.169	4.000	6.011	2
12	0.045	1.251	5.093	2
13	0.017	3.904	9.024	2
14	-0.001	4.703	4.062	2
15	-0.118	4.640	5.872	2

We will derive rule for fuzzy term PS, and the region of the fuzzy term PS is assumed to be [0.165, 0.212] for the first variable (v1), [3.877, 4.774] for the second (v2), and [4.890, 6.036] for the third (v3), as shown in Figure 5.

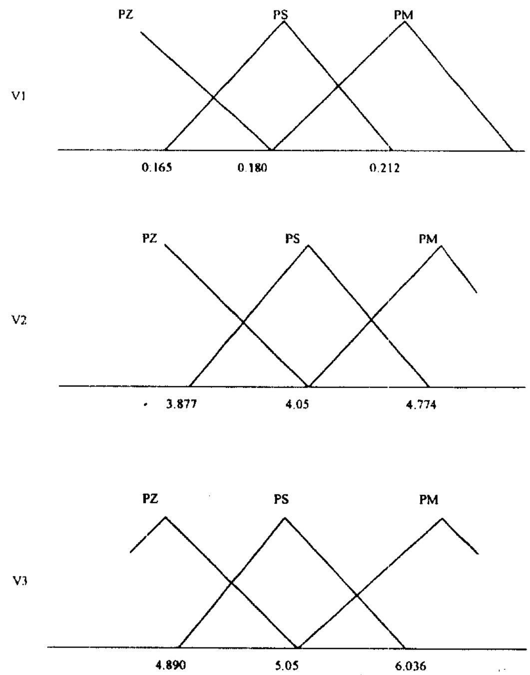

Fig. 5 Fuzzy term PS of the variables V1, V2, and V3

For binary conversion, the samples in each variable are translated into 1's or 0's depending on their membership status in the term PS. If the sample value is in the fuzzy term PS range, it is translated to 1, otherwise, 0. Then, the binary table for the fuzzy term PS will be resulted as indicated below:

sample #	D1	D2	D3	class
1	1	0	0	1
2	1	0	1	1
3	1	0	1	1
4	0	0	1	1
5	1	0	0	1
6	1	0	1	1
7	0	1	0	2
8	0	0	1	2
9	0	1	1	2
10	1	1	0	2
11	1	1	1	2
12	0	0	1	2
13	0	1	0	2
14	0	1	0	2
15	0	1	1	2

Now, we follow the steps to produce rules for the fuzzy term PS using the simplified rule extraction method.

Step 1

First, we find the digit index. For the samples of each digit (D1, D2, and D3), we count the number of 1's in the class 1 samples and the number of 0's in the class 2 samples. Then we divide the number of 1's by the number of samples in class 1, and the number of 0's by the number of samples in class 2 to calculate digit count; from the digit count, we get digit index. This first step of the process is tabulated below:

	D1		D2		D3	
	Class 1	Class 2	Class 1	Class 2	Class 1	Class 2
Number of 1 and 0's	5	7	0	2	4	4
Digit Count	5/6	7/9	0/6	2/9	4/6	4/9
Digit Index	0.61		0.78		0.11	

From the above table, the second digit has the biggest digit index, so we start the separation process with the second digit. Therefore we have "x1x for class 2" as the first step of the rule. Then $z_1 = 7$ (from class 2) and $n_1 = 7$ (from both classes), and the mean probability is $\overline{P_1} = (z_1 + 1)/(n_1 + 2) = 8/9 = 0.89$. This value is taken to be the weight of the first step of the PS rule. Eliminating the samples having 1 in their second variable, we have:

sample #	D1	D2	D3	class
1	1	0	0	1
2	1	0	1	1
3	1	0	1	1
4	0	0	1	1
5	1	0	0	1
6	1	0	1	1
8	0	0	1	2
12	0	0	1	2

Step 2

The process of the drawing digit index is illustrated in the table below:

	D1		D2		D3	
	Class 1	Class 2	Class 1	Class 2	Class 1	Class 2
Number of 1 and 0's	5	2	0	2	4	0
Digit Count	5/6	2/2	0/6	2/2	4/6	0/2
Digit Index	0.83		0.00		0.33	

We see that the digit index is biggest for the first digit (D1), so "1xx for class 1" becomes the second step of the rule for fuzzy term PS. Then $z_2 = 5$ and $n_2 = 5$ and, therefore, the weight of the step 2 of the rule is $\overline{P_2} = 6/7 = 0.86$. The remaining samples are:

sample #	D1	D2	D3	class
4	0	0	1	1
8	0	0	1	2
12	0	0	1	2

Step 3

The process of the drawing digit index is shown below:

	D1		D2		D3	
	Class 1	Class 2	Class 1	Class 2	Class 1	Class 2
Number of 1 or 0's	0	2	0	2	1	0
Digit Count	0/1	2/2	0/1	2/2	1/1	0/2
Digit Index	0.00		0.00		0.00	

As we see that all the digit indices are same, we can choose "xx1 for class 2" for the third step of the PS rule. Then, $z_3 = 2$ and $n_3 = 3$ and, therefore, the rule weight for step 3 of the rule is $\overline{P_3} = 3/5 = 0.6$.

After the third step, rule derivation stops because all the samples are eliminated. Therefore, the fuzzy rule for term PS consists of 3 steps. The fuzzy rule for term PS can be tabulated as follows:

PS Rule

Steps	V_n	Term	Class	Weight	Rule in Words
1	2	PS	2	0.89	IF V_2 is PS, THEN class_2
2	1	PS	1	0.86	ELSE IF V_1 is PS, THEN class_1
3	3	PS	2	0.60	ELSE IF V_3 is PS, THEN class_2
					ENDIF

The above procedure must be performed for other fuzzy terms; the final fuzzy rule will be a set of 8 independent production rules.

Ⅵ. Inference and Defuzzification

In the previous sections, we have discussed about the algorithmic procedure for membership function and production rule generation. However, this algorithmic approach is not complete unless suitable inference and defuzzification method for classification and decision-making problem are provided.

A. Inference

Inference is a mechanism by means of which a conclusion is drawn from sample data and rules. It is designed to evaluate the rules whose conditional parts satisfied. A popular inference method is "max", in which the final membership grade for an output is the union of the fuzzy membership grades which are the outputs of the individual rule. The values of the membership grades are determined by the degrees of membership in the conditional part of the rules [7]. If OR is used to form the conditional part ("max-max"), the grade value is determined by the maximum of the membership grades, and if AND is used ("max-min"), it is determined by the minimum of the grades. However, this inference method does not provide a proper scheme to handle rules accompanied by weights; to accommodate production rules with rule weights, therefore, a new inference method is required.

Two methods of inference are devised in this research. The first method is to check for a matched (non-zero) premise starting from the first fuzzy term (PB) to the last (NB). If a step (each step has the fuzzy term PB in the conditional part) of the rule is matched, then a triplet of the firing strength, the corresponding weight, and the class identification for the fuzzy term PB, $\{\mu_{PB}, W_{PB}, C_{PB}\}$ will be resulted. Then, we move to the next fuzzy term, PM. This process goes on to the last fuzzy term. Therefore, finished with this method, we have 8 triplets represented by $\{(\mu_j, W_j, C_j), j = PB, PM, PS, PZ, NZ, NZ, NM, NB\}$. We call this process the "overall match" method.

The other inference method is called the "step match" method because we check each "step" of the 8 fuzzy rules, one at a time. Unlike the "overall match" method, this does not check all the steps of a fuzzy term rule. Instead, this method checks the first step of the first term rule, and the first step of the second fuzzy term rule, and then the first step of the third fuzzy term rule, and so on. Depending upon the result of the match and fire at each step, the step check will either continue or stop. This means that if, for example, the first steps of any one or more fuzzy rules are matched and those of the other fuzzy rules are not, process stops; then, two triplets of the firing strength, the corresponding weight, and the class identification, $\{\mu_{PB}, W_{PB}, C_{PB}\}$ and $\{\mu_{PM}, W_{PM}, C_{PM}\}$, respectively, will be resulted. If we do not have any matched set in the first step of the fuzzy term rules, we move to the second step. This process will go on until there is a matched set or all steps are finished.

B. Defuzzification

Usually, more than one fuzzy rule may be matched and fired at the same time, so there should be a conflict resolution measure. This output decoding method is called defuzzification. Defuzzification is the process of converting the result of the inference into a non-fuzzy value which best represents the membership functions of an inferred fuzzy classification actor. One of the most famous methods of defuzzification is center of area method which can be represented by

$$x_d = \frac{\sum_{i=1}^{m} x_i \, \mu_k(x_i)}{\sum_{i=1}^{m} \mu_k(x_i)}$$

where,

m = number of quantized level of variable,

x_i = value of a variable at the quantized level i,

$\mu_k(x_i)$ = membership degree of the matched fuzzy term k at the value x_i, and

x_d = defuzzified value

The other popular method is the "mean of the maximum" method which can be represented by

$$x_d = \sum_{i=1}^{l} \frac{x_i}{l}$$

where, l is the number of quantized x values which reach their maximum membership degrees.

For classification and decision-making problems, and for fuzzy rules with rule weights, however, the conventional defuzzification method is not appropriate. Due to the nature of the problem, the output of the classification process should not be an analog value but a binary value, i.e., the output is not a quantity but a discrete status. This unique characteristic of classification problem and the introduction of rule weight require the development of new defuzzification method. A new method, "pivot balance defuzzification", is explained below.

The lever and pivot concept finds solutions in the unique environment of the classification problem: binary output and multiple sets of fired length, weight, and class identification. The basic idea of this method is to place the "weights" in the location designated by the firing "length", and then, to move the pivot to the position which balances the lever (see Figure 6). A firing strength determines the distance from the center of the lever, and the weight of the fired rule acts as a measuring weight. We place the weights on the left side of the lever if the class identification $C_k = 2$, and, on the right side if $C_k = 1$. If we scale the lever so that it is centered on 0, class 1

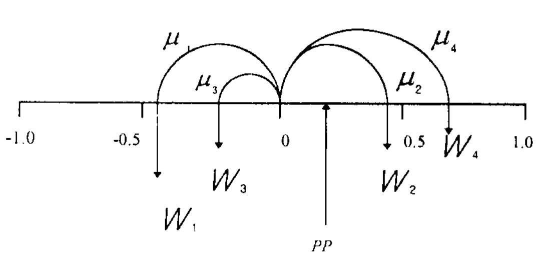

Fig. 6 Illustration of the Pivot and Balance Defuzzification

includes all points on the right (positive) side of the pivot point pp, and class 2 includes all points on the left (negative) side. Therefore, the sign of the final defuzzified output, the pivot point pp, decides the class of the sample: class 1 if pp is positive and class 2, if negative.

The pivot point pp for defuzzification, therefore, can be expressed by the following equation.

$$pp = \frac{\sum_{C_1} \mu_n W_n - \sum_{C_2} \mu_m W_m}{\sum_{C_1 \& C_2} W_l} \tag{11}$$

where,

μ_k is the firing strength of the matched rule of the term k, k = PB, PM ,..., NB, and

W_k is the weight of the matched fuzzy rule of the term k, k = PB, PM ,..., NB.

Ⅶ. Conclusions

An algorithmic method to automates the procedure for fuzzy logic application to decision-making and classification problems is presented. This approach is based on an entropy minimization principle to generate, using only sample data, membership functions and fuzzy rules. Membership function generation using the clustering principle is discussed and the rule derivation, along with the rule weight determination, is illustrated.

Ⅷ. Acknowledgment

The author wishes to acknowledge the financial support of the Korea Science and Engineering Foundation in the project # KOSEF 951-0912-033-2

References

1. E. H. Mamdani, "Application of Fuzzy Algorithms for Control of Simple Dynamic Plant," *Proceedings*

of the IEE, vol. 121, no. 12, pp. 1585-1588, December 1974.
2. C. J. Kim and B. D. Russell, "A Structure for Fuzzy Decision-Making System for Power System Protection," *Proceedings of the IEEE second International conference on fuzzy systems*, March 28-April 1, 1993, San Francisco, CA.
3. C. J. Kim and B. D. Russell, "Application of Fuzzy Expert Model for Power Systems Protection," *Proceedings of the Fifth International Fuzzy Systems Association World Congress*, pp. 1074-1077, Seoul, Korea, 1993.
4. B. V. Dasarathy, "Practical Models for Fuzzy Membership Estimation," *Proceedings of the Second International Workshop on Industrial Fuzzy Control & Intelligent Systems*, p. 187-193, December 2-4, College Station, TX, 1992.
5. Lin-X. Wang, *Adaptive Fuzzy Systems and Control: Design and Stability Analysis*, Prentice-Hall, Englewood Cliffs: NJ, 1994.
6. C. T. Lin, C. S. Lee, "Neural Network-based Fuzzy Logic Control and Decision System," *IEEE Transactions on Computer*, vol. 40, no. 2, pp. 1320-1336, 1991.
7. H. -J. Zimmermann, *Fuzzy Set Theory and Its Applications*, Second Edition, Kluwer Academic Publisher, 1991.
8. P. J. King, E. H. Mamdani, "The Application of Fuzzy Control Systems to Industrial Processes," *Proceedings of the Workshop on Discrete Systems and Fuzzy Reasoning*, pp. 235-243, Queen Mary College, University of London, January 1976.
9. R. Christensen, *Entropy Minimax Sourcebook*, vol. I-IV, Entropy Ltd., Lincoln, MA, 1980.
10. C. J. Kim, "An Intelligent Decision-Making System for Detecting High Impedance Faults," Ph.D. Dissertation, Texas A&M University, College Station, Texas, December 1989.

김 창 종(C.J.Kim) 정회원
1980년 : 서울대 공대 전기공학과 졸업(학사)
1982년 : 서울대 대학원 전기과 졸업(석사)
1983년~1985년 : LG산전연구소 연구원
1989년 : Texas A&M University 전기과 졸업(Ph.D.)
1990년~1991년 : Texas A&M University, 전기과 Post-Doc연구원
1992년~1994년 : Texas A&M University, 전기과 연구교수
1994년~현재 : 수원대 전기전자정보통신공학부(조교수)
※관심분야 : 지능제어응용, 예측및 인식에 대한 컴퓨터및 지능시스템 응용