Electronics and Communications in Japan, Part 2, Vol. 90, No. 9, 2007
Translated from Denshi Joho Tsushin Gakkai Ronbunshi, Vol. J89-A, No. 12, December 2006, pp. 1142–1152

Realization of Feature Description Systems for Clusters by Rule Generation Based on Genetic Programming and Its Applications

Jianjun Lu,[1,3] Yoshinori Kishikawa,[2] and Shozo Tokinaga[1]

[1]Graduate School of Economics, Kyushu University, Fukuoka, 812-8581 Japan

[2]Faculty of System Science and Technology, Akita Prefectural University, Honjyou, 015-0055 Japan

[3]China Agricultural University, Beijing, 100083 China

SUMMARY

This paper deals with the realization of feature description systems for clusters by rule generation based on genetic programming (GP) and its applications. First, the data are divided into several clusters by using conventional clustering algorithms. Then logical variables corresponding to the categorical variables are introduced, and the logical expressions using these logical variables are defined as rules to extract the targeted cluster from the dataset. The rules are improved by GP so that they are valid (become true) only for the targeted cluster. Unlike ordinary GP procedures, the fitness of individuals is defined as proportional to the number of hits inside the targeted cluster, but also to the inverse of the number of hits outside the targeted cluster. In simulation studies, the system is applied first to artificially generated samples and clusters to examine the performance of the system, and then to personal loan assessment problems, after which the evaluation of several kinds of clustering problems is summarized. © 2007 Wiley Periodicals, Inc. Electron Comm Jpn Pt 2, 90(9): 87–97, 2007; Published online in Wiley InterScience (www.interscience.wiley.com). DOI 10.1002/ecjb.20380

Key words: feature description of clusters; genetic programming; rule generation; number of hits.

1. Introduction

Recently, the development of large information storage systems such as data warehouses has made it possible to retrieve or reuse huge volumes of data. Data retrieval systems help to support the decision processes in various fields such as financial activities [1, 2]. In particular, the characterization of a set of data extracted from storage according to a standard measure (these data sets are called clusters below) will help to provide highly intelligent and useful information.

Generally, problems of data clustering are categorized into two groups. The first one consists of finding the unknown cluster to which the underlying data should belong (cluster estimation), and the second is characterization of the cluster features of the underlying group (feature description of clusters). For cluster estimation, various methods such as multivariate analysis are applicable. However, for the feature description of clusters, direct extension of cluster estimation is not relevant.

In addition, it is necessary to create linguistic expressions for the description of cluster features rather than numerical descriptions. Even though there are systems oriented to linguistic expressions, such as ID3, these systems usually need multiple clusters to distinguish between clusters (called pair-samples). For example, if we find a linguistic description for a cluster having "good" as a prescribed value, we also need a cluster having "bad" as a prescribed

Contract grant sponsor: Japan Society for the Promotion of Science under grant number (B) 15310120.

value [3–6]. Thus, these methods cannot be directly used to describe the features of a single isolated cluster.

This paper deals with feature description systems for clusters using rule generation based on genetic programming (GP), and their applications to data mining [7, 8]. In the method, we prepare various kinds of logical expression (having a tree structure and called individuals) for the data (called samples) in the underlying cluster by using variables for categorical values, and we then improve the logical expressions by using GP. As the fitness of each individual, we use the number of hits for individuals (cases in which the logical expression corresponds to individuals) for the samples in the cluster. Then we use the GP procedure to improve the capability of the logical expressions until a stable description for the cluster is obtained.

The GP method has been successfully applied to the approximation of chaotic dynamics, and also to the knowledge representation of agents in simulation studies of artificial markets [9–18]. The method is also usable to recognize time series segments of stock prices and for time series prediction.

In the first stage, we extract a group of samples as a cluster by using relevant numerical evaluations. Then, in the next step, we assume that all variables assigned to the samples are logical variables whose values are categorical values. In the third step, we prepare a pool of individuals which correspond to the tree structure of the logical expression, using logical variables to describe the features of the cluster. Then we use the GP procedure to improve the capability of the logical expressions (individuals) so that the logical expressions are true only for the samples in the underlying cluster, and are false value for samples outside the cluster.

The definition of individuals is slightly different from that in ordinary GP systems. The fitness of individuals is proportional to the number of hits for the samples in the underlying cluster, but also is inversely proportional to the number of hits for the samples outside the cluster. By an extension of the definition of fitness, we can retain stable individuals (logical expressions) which are true only for the samples in the cluster.

As applications, we first examine the ability of the system to handle artificially generated samples. We then apply the method to the evaluation decision making for personal loans in order to demonstrate its effectiveness. The feature description system is also applied to eight groups of samples arbitrarily collected from various databases.

Below, in Section 2, we present an overview of the system treated in the paper. Section 3 describes the basics of the GP method in relation to the feature description of clusters. In Section 4, we give several example datasets to show the capabilities of the system.

2. Feature Description System for Clusters Based on GP

2.1. Why feature description for a single cluster?

At the outset, we would like to explain the scope of this paper, which is different from that of conventional investigations of clustering and feature description. First, the usual clustering methods deal with problems of classifying samples into several groups (clusters) by using input variables referring to the distance among samples. But our approach is mainly interested in the explanation (description) of features for samples in a cluster by using logical expressions which are easily transformed into natural language. This kind of feature description is useful for characterizing certain arbitrarily collected samples.

Second, even though there are several conventional methods for feature description using tree structures, such as ID3, in our method we assume that there is only a single cluster for which the feature description should be found. Conventional methods such as ID3 need multiple separated and oppositional clusters to find the tree structure for feature description. For example, a cluster includes a group assigned “yes” as a prescribed value, and another group assigned “no” is also needed. But in our method, samples belonging to a single cluster are assumed, and we do not need multiple clusters to find feature descriptions. The advantage of using a single cluster is twofold. First, we do not need to collect multiple clusters, which makes data mining easier. Second, the feature description ability is improved. Since conventional methods depending on multiple clusters try to build a single tree structure to be used in clustering multiple clusters, the resultant clustering is inefficient compared to our method, in which a tree structure must be built only for a single cluster [23, 24].

These remarks on the introduction of the GP method into feature description will be developed below.

2.2. Overview of system configuration

An overview of our feature description system for clusters based on GP treated is presented below [7]. The system shown in Fig. 1 is composed of three subsystems.

(1) Description of samples by categorical variables

Generally, two kinds of variables (numerical variables and logical variables) can be used to characterize the samples, but in our system we assume only logical variables. The numerical variables are transformed into categorical variables by conventional methods of discretization (details are omitted here) [16, 17].

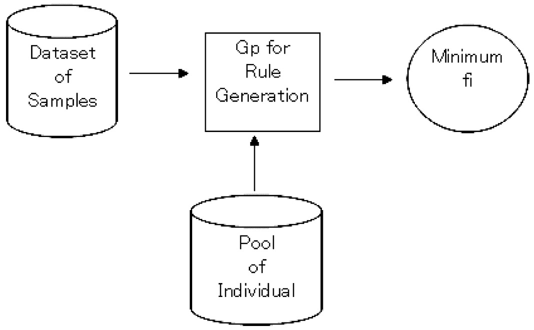

Fig. 1. Overview of system configuration.

(2) Generation of initial individuals for logical expressions

It is assumed that the logical expressions represented by the categorical variables are used to describe the features of samples in the cluster which correspond to the individuals. For tractability, it is assumed that the logical expressions have a binary tree structure. At the beginning of the GP procedure, we generate an initial pool of individuals (say, 1000 individuals) by using random numbers.

(3) Definition of fitness of individuals

The fitness of the k-th individual in the GP procedure is defined in terms of the number of hits, that is, the number of samples in the cluster for which the logical expression corresponding to the individual is true. The number of hits c for samples outside the underlying cluster in the whole dataset is also used. We first define the index

$$y_k = T - h_k^2/n_k \tag{1}$$

where we use the following notation:

n_k: the number of samples in the whole dataset for which the logical expression for individual k is true;
h_k: the number of samples in the underlying cluster c for which the logical expression for individual k is true;
T: the total number of samples in cluster c.

The fitness of individual k (denoted as f_k) is defined by adding some positive number to y_k, and taking the inverse of the number:

$$f_k = (a + y_k)^{-1} \tag{2}$$

If the number of hits for logical expressions covering samples in the cluster increases, then accordingly, measure (1) becomes closer to zero. Since the denominator of Eq. (1) includes the number n_k as the second term, the logical expressions are improved so that they cover only samples in cluster c, and as many of those as possible. On the other hand, if the logical expression is true for samples outside of the cluster c, the fitness of the individual is small.

After calculating the fitness of individuals, we apply the GP procedure in order to improve its feature description ability for the cluster.

(4) Termination of search for feature description

If the maximum fitness value of individuals becomes sufficient and further improvement is not expected, we terminate the GP procedure. As a result, we have a feature description derived from the individuals with greater fitness.

2.3. Extraction of clusters

In this paper we assume that the samples in a cluster c have already been obtained, and we focus only on finding the feature description based on the GP procedure. Therefore, we do not explain further how to extract a cluster from the whole dataset. However, in the simulation studies used to examine the capability of the system, we note the following points of importance in avoiding trivial cases of applications.

(1) No deterministic numerical or logical variables

We do not use deterministic numerical or logical variables for cluster extraction; otherwise the problem of feature description would become trivial. For example, if we use only a single categorical variable to extract clusters, it is clear that the features of the cluster are described by the variable. Therefore, when we use conventional methods for the extraction of clusters, we use at least six numerical and categorical variables.

(2) Steady extraction of clusters

We try to find a combination of variables for the extraction of clusters that enables us to avoid cases in which the size of the cluster samples is very small (corresponds to rare events). Even though the feature description of rare events is also interesting, we are now focusing on the steady extraction of clusters to evaluate the system.

(3) Unification of methods for cluster extraction

Generally, procedures for extracting clusters include various problems, such as the definition of mean values for the clusters, and the distance from these mean values. These variations sometime affect the performance of the system. However, we are not primarily interested in clustering itself, and we therefore utilize the ordinary method (called the centroid method). Of course, variations of the clustering method and its effect on system capabilities should be discussed in any case; but in simulation studies in which we changed the clustering method from the ordinary centroid method to several other alternatives, we found no significant differences in system performance. Therefore, in the following discussion we assume that the samples of the targeted cluster have already been defined (collected) from the whole dataset, so that the only task of the system is feature description.

3. Logical Rules for Feature Description and the GP

3.1. Basics of the GP

For simplicity, we start with GP operations on arithmetic expressions. The GP is an extension of the conventional GA (genetic algorithm) in which each individual in a population (pool of individuals) is a computer program composed of arithmetic operations, standard mathematical expressions, and variables [9–21].

There are several ways to represent mathematical expressions in the GP. Among them, the prefix representation is attractive due to its simplicity for GP operations.

The prefix representation is equivalent to a tree representation in which the external points (leaves) of the tree are labeled with terminals (i.e., constants and variables), the root of the tree is labeled with a primitive function such as a binomial operation +, –, ×, /, or the operation of taking the square root of a variable. For example, if we have a prediction for the time series $x(t) = [3 \times x(t-1) - x(t-2)] \times [x(t-3) - 4]$, then we have the corresponding next prefix representation

$$\times - \times 3x(t-1)x(t-2) - x(t-3)4 \qquad (3)$$

The equations represented by using the prefixes are interpreted on the basis of stack operations.

We must assure that after initialization, crossover, and mutation, we have a valid tree representation. For this purpose, the stack count (denoted as *StackCount*) is useful [14]. *StackCount* is the number of arguments pushed onto the stack minus the number of arguments removed from it. The cumulative *StackCount* never becomes positive until we reach the end, at which point the overall sum must still be 1.

By using *StackCount*, we can identify the terminals of the subtree which are candidates for the crossover operation. The basic rule is that any two loci on the two parent genomes can serve as crossover points so long as the current *StackCount* just before those points is the same. The crossover operation creates new offspring by exchanging subtrees between two parents.

Before applying the genetic operation, we must evaluate the ability of each individual (tree structure). This ability is called the fitness, and is calculated by comparison of the predicted value for the individual and the observed value. Usually, we calculate the root mean square error (*rmse*) between $x(t)$ and $\tilde{x}(t)$ and use it as the fitness. The fitness S_i of the i-th individual is defined as the inverse of *rmse*.

We iterate the following steps until the termination criterion is satisfied [8–21]:

(Step 1)

Generate an initial random population of the functions and terminals of the problem (constants and variables).

(Step 2)

Execute each program (evaluation of system equation) in the population and assign it a fitness value by using the fitness measure. Then, sort the individuals according to the fitness S_i.

(Step 3)

Create a new population of programs by applying one of two primary operations (see below). These operations are applied to individuals chosen with a probability p_i based on the fitness. The probability p_i is defined for the i-th individual as

$$p_i = (S_i - S_{min}) / \sum_{i=1}^{N} (S_i - S_{min}) \qquad (4)$$

where S_{min} is the minimum value of S_i, and N is the population size.

Create new individuals (offspring) from two existing ones by genetically recombining randomly chosen parts of two existing individuals by the crossover operation applied at a randomly chosen crossover point.

(Step 4)

If the result designation is obtained by the GP (the maximum value of the fitness becomes larger than the prescribed value), then terminate the algorithm, otherwise return to Step 2.

We apply the mutation operations defined as follows at a probability p_M.

(Global mutation)

Generate an individual I_s, and select a subtree which satisfies consistency of the prefix representation. Then, select at random a leaf in the individual to which the mutation is applied, and replace the leaf by the subtree of the individual I_s.

(Local mutation)

Select at random a leaf in the individual to which the mutation is applied, that is, replace the parameter at the leaf by another value (a primitive function or a variable).

3.2. Application of GP to logical rule generation

In previous investigations, we used the GP procedure to generate and improve logical rules (expressions) for several tasks [6, 12, 14, 15]. To simplify the method of GP, we assume that the logical expressions are represented in binary forms in which two predicates are combined with logical operators; but this restriction does not limit the applicability of the system. Additionally, the separation of the whole system into subsystems makes system configuration easier. One subsystem is used to manage the arithmetic expressions included in the predicates, and the other subsystem is used to manage the logical expressions.

Basically, the GP procedure developed for the approximation of arithmetic expressions (functional forms) is easily extended to the approximation of logical expressions by changing the operators and operands. The logical expressions included in the production rules are the same as the arithmetic expressions using prefix representation, with the operands replaced by propositions and the arithmetic operators replaced by logical operators:

numerical variables $v_i \rightarrow$ logical variables X_i
arithmetic operators $+, \times \rightarrow$ OR, AND

In this paper we assume that all of the samples are characterized by logical variables, then use a relatively simple method to generate logical expressions. Suppose that there are categorical variables $v_1, v_2, \ldots, v_m$, and that these variables can have the values $s_1, s_2, \ldots$. For example, if the logical variables v_1, v_2 take the values s_3, s_5, then we have

$$v_1 = s_3, v_2 = s_5 \tag{5}$$

These binary expressions are then used as predicates in the logical expressions in the GP. For example, we define new logical variables X_{kj} represented by an input variable v_i, such as

$$X_{kj} = \begin{cases} True, & \text{if } v_k = s_j; \\ False, & \text{otherwise} \end{cases} \tag{6}$$

We also define the fitness of individuals as the degree to which the accuracy of the generated rules corresponds to the underlying individual. To improve the fitness of individuals, we apply the GP operations to the logical expressions.

The fitness of individuals is evaluated as follows.

(1) Calculation of logical values

By substituting the values of input variables v_i, we can evaluate X_{kj} included in the predicates.

(2) Interpretation of propositions

Since we know the values of the predicates as logical values, we can calculate the values of logical expressions using logical operators.

(3) Interpretation of logical formulas

Finally, we can determine the logical value of the whole logical formula (individual) by applying logical operations among propositions. Then the value is compared with the prescribed observation r to calculate the fitness.

4. Applications

4.1. Feature description for artificial samples

Before applying the proposed method of feature description to a real dataset, we examine the ability of the system by using artificially generated samples. In the simulation study, we assume we have a cluster c whose samples are already characterized by certain features, and other samples besides the cluster are also mixed into the dataset. Then, the feature description for the cluster c is obtained by the GP procedure, and the result is compared with given (known) settings.

In the following, for simplicity, it is assumed that the categorical variables assigned to samples are denoted as $v_1, v_2, \ldots, v_m$, and the variables take the values 1, 2, ... The ranges of these categories are defined as $r_1, r_2, \ldots, r_m$. It is also assumed that the categorical variables v_i, $i = 1$ to m for the cluster c which should be extracted and whose feature should be truly described have the same value of 1. The categorical variables v_i, $i = 1$ to m for samples belonging to other clusters have random numbers different from 1.

To check the ability of the system of the paper, the following points are examined by the simulation studies.

(1) Validity of extraction of samples belonging to cluster c

If the extraction of cluster c has truly succeeded, then we must have $y_i = 0$, $h_i = n_i = T$. The ability of the GP method can be tested by checking these values.

(2) Time of extraction depending on the number of categorical variables

It seems likely that if the number of categorical variables is large, then the time necessary for the extraction of clusters by the GP method will be longer. Therefore, the relation between the number of categorical variables m and the time for extraction of the cluster should be estimated.

(3) Time of extraction depending on ranges of categorical variables

Similarly, if the ranges of categorical variables r_i, i = 1 to m are large, then the number of combinations of variables becomes large and the time necessary for the extraction of clusters by the GP method will be longer. Therefore, the relation between the ranges of categorical variables and the time for extraction of the clusters should be estimated.

(4) Categorical values assigned to samples besides cluster c

It is assumed that the categorical variables for samples in cluster d other than cluster c have values different from 1. In these cases, it seems likely that the time for extraction of cluster c may depend on whether all categorical variables in samples of cluster d are set different from 1, or only a restricted number of categorical variables take values different from 1. If a smaller number of categorical variables of samples in cluster d have values different from 1, then the differentiation between the clusters c and d becomes small, resulting in long time consumption for feature description.

Considering the above reasoning, we classify the cases for simulation studies for the cluster extraction and feature description as follows.

numbers of categorical variables: m = 3, 4, 5
ranges of categorical variables: r_i = 2, 3, 4

The setting of the categorical variables in clusters d other than cluster c is given as follows:

Case A: only one randomly selected categorical variable is assigned a value different from 1.

Case B: two randomly selected categorical variables are assigned values different from 1.

Case C: three randomly selected categorical variables are assigned values different from 1.

The condition for the GP procedure is given as follows. In particular, the length of the array corresponding to the size of individuals is chosen sufficiently large to confirm the final cluster extraction performance:

number of samples in cluster c: T = 100
number of samples outside cluster c: 100
maximum size of array in individuals: M_s = 10
size of pool of individuals: 1000

Table 1 shows typical examples for the cases where m = 3, r_i = 3, and the categorical variables are selected in accordance with Case C. In the table, the optimal values of h_k, n_k, y_k in Eq. (1) for the individuals having the highest fitness (for simplicity, we denote them as h, n, y) are plotted against the number of generations of the GP procedure (denoted as N_{GP}).

It is seen from the table that if $m \leq 4$ the extraction of cluster c is completed after about 500 generations of the GP procedures, and the logical expression finally obtained describes the true features of cluster c.

Additionally, in Table 2, the comparison of the time until convergence of extraction is shown versus the variations of the number of categorical variables other than the cluster c for Cases A, B, and C (in Table 2, they are denoted as N_F). Table 2 shows the results only for the case in which the range is r_i = 3, and for Case C, but for the other cases we find small changes of the values in N_F, and therefore, only the result for the case r_i = 3 is shown. Moreover, it is omitted in Table 2; in the final stage of GP procedure, we can extract only the samples in cluster c, and we have y = 0, n = h = T, which affirms that cluster extraction is completed.

The results obtained from the simulation studies suggest that if the number m of categorical variables is 3 or less, the extraction of clusters is successfully realized without strict dependence on the number of ranges of categorical variables inside 300 GP generations. This fact suggests that if the number of categorical variables used for the feature description is relatively small, then the GP procedure treated in the paper is still effective even if the number of samples is large.

However, when $m > 4$, the time (GP generations) to extract clusters becomes very large. The reason for the increased extraction time seems to be an increase in the number of combinations of logical variables in logical expressions. We find that if the number of categorical variables is greater than 7, then the time for cluster extraction becomes larger than 600 GP generations.

Table 1. Example of relation among h, n, y, N_{GP} (Case C, m = 3, r_i = 3)

N_{GP}	1	50	100	150
h	4	16	23	30
n	47	19	34	30
y	29.7	16.5	14.4	0

Table 2. Feature of cluster extraction for Case A, B, C

$r_i = 3$	$m = 3$	$m = 4$	$m = 5$
Case A	172	326	367
Case B	153	291	352
Case C	126	274	334

4.2. Applications to German credit data

An experiment on real-life credit-risk evaluation was performed using German credit data. The German credit data are obtainable from a website. The data consist of 1000 records of personal loans, and the input variables for one record include 7 numerical data and 13 categorical data [3, 22].

Even though the original purpose of the dataset is the generation of accept/deny rules for personal loans, we use the dataset to examine the capabilities of the proposed GP method. First, we select 100 samples at random from the dataset and classify them into three clusters by using the following seven numerical variables based on a conventional software package.

y_1: terms of credit (months until end of repayment)
y_2: credit amount (applied amount to be loaned)
y_3: interest rate of credit (interest rate of loan)
y_4: length of current residence (months at current address)
y_5: age (current age of applicant)
y_6: number of credit accounts (how many credit applications in existence)
y_7: number of dependents (number of persons in family)

Then, we assume one cluster (say cluster *c*) is the target cluster whose features must be described, and the other samples belonging to other clusters are regarded as samples outside cluster *c*. To extract the feature description for the cluster *c*, we use the following six categorical variables.

x_1: state of saving account (no, ≤ 200 DM, > 200 DM, 0)
x_2: length of contract (shorter than 30 months, longer than 30 months)
x_3: credit history (no, completed, current repayment, one default, risky)
x_4: purpose of credit (for example, car)
x_5: state of savings account (≤ 100 DM, 100 to 500 DM)
x_6: guarantor (yes, sharing, no)

The conditions for the simulation studies are as follows.

Table 3. Relation among *h*, *n*, *y*, N_{GP}

N_{GP}	1	100	200	300	400	500	600
h	6	12	17	23	25	26	30
n	21	14	51	54	37	30	30
y	28.2	19.5	24.3	19.9	12.7	7.7	0

maximum size of array of individuals: 10
size of pool of individuals: 1000

Table 3 shows the optimal values of h_k, n_k, y_k (we denote them as *h*, *n*, *y*) versus the number of generations of the GP procedure (denoted as N_{GP}). It is seen from the table that after about 600 generations of the GP procedures the feature extraction (description) is completed, and the logical expression finally obtained describes the true features of cluster *c*. Table 4 depicts several logical expressions corresponding to the feature description of cluster *c*. It is seen that these expressions are simple enough that their meaning can be interpreted.

4.3. Applications of feature description to real data

In the following, we explain the simulation studies of the proposed feature description applied to multiple real datasets, and discuss the average performance. The details of these datasets are summarized in Table 5, and we omit an explanation of the method of collection and the sources of these datasets. In Table 5, we give the names of the dominant categorical variables.

First, we select about 100 to 300 samples from the dataset at random, and then we divide these samples into three clusters by using a conventional numerical method of clustering. Then, in the next step, we apply the proposed GP procedure for the extraction of clusters and feature description of these three clusters, independently. That is, if we focus on cluster *c*, the samples belonging to clusters *d* different from the underlying cluster *c* are regarded as samples outside cluster *c*. The simulation results for the

Table 4. Example of finally obtained logical expression describing cluster characteristics

And X_{53} And And X_{23} X_{42} X_{14}
And And X_{21} X_{62} Or X_{13} X_{41}
And Or X_{53} X_{61} And X_{23} X_{41}
Or X_{62} And X_{41} And X_{33} X_{22}

Table 5. Overview of data

No.	Overview	Main categorical variables
No.1	election	age,carriers,candidates
No.2	employee	carrier,initial salary,race
No.3	opinion for life	occupation,satisfaction,studies
No.4	house sales	region,ranks of prices
No.5	customer data	yearly income,kinds of support
No.6	social life	hobby,occupations
No.7	graduated	graduate year,gender,initial salary
No.8	reputation of firm	trust,brand power

three categories are summarized, and the average performance is evaluated.

The conditions for the simulation studies are summarized as follows.

maximum size of array in individuals: $M_s = 10$
size of pool of individuals: 1000

Table 6 shows the average of the sample numbers T included in each cluster, the average of the number of categorical variables m, and the average of ranges r_i of the first five categorical variables for these three clusters. Table 7 gives the number of GP generations N_F necessary to obtain the final result of feature description. We note that by using the proposed feature description method, we finally obtain 100% correct classification of samples to the underlying clusters; but the result is omitted here.

As can be seen from the result, the GP procedure to extract the clusters and to give feature descriptions works effectively after 500 or 600 GP generations even for real-world data, despite wide variations.

4.4. Indirect comparison of capability

As mentioned earlier, conventional methods such as ID3 have the purpose of presenting tree structures for classifying samples into several clusters. However, in these cases there must exist multiple clusters (groups) that have been assigned prescribed values. Therefore, these conventional methods are basically different from the method treated here, where we can obtain feature descriptions for a certain (single) cluster of samples.

Table 6. Number of samples and categorical variables (numbers and ranges)

No.	T	m	r_1	r_2	r_3	r_4	r_5
No.1	44	4	3	3	3	2	-
No.2	76	9	3	3	3	3	3
No.3	145	5	2	3	3	3	3
No.4	122	6	3	3	3	3	3
No.5	67	5	3	3	4	4	3
No.6	58	9	2	3	3	3	3
No.7	36	5	3	2	3	3	3
No.8	31	7	3	3	3	3	3

Table 7. GP generation necessary for cluster extraction

cases	No.1	No.2	No.3	No.4
N_F	500	550	500	550
cases	No.5	No.6	No.7	No.8
N_F	600	500	600	500

But a kind of indirect comparison of capabilities is possible. For example, in the first stage we divide all samples into three clusters assigned prescribed values, and then apply the conventional methods of clustering. In the following we describe comparative studies using ID3, a typical inductive method [6]. Additionally, we try to use the conventional multivariate method for classification (Multivariate Discriminant Analysis: MDA), using only the numerical variables as inputs.

First, we prepare three clusters (cluster A, B, and C) for each dataset, having similar sample sizes. For example, if the underlying cluster is cluster A, then clusters B and C are regarded as samples outside cluster A. We use the software package for MDA analysis, and use the notations A, B, and C as the prescribed values for the samples. In the application of ID3, to avoid redundancy of the tree structure, we terminate the generation of the tree structure when about one-quarter of the samples remain to be unclassified (so-called pruning of the tree and leaves). However, we present the result for the cases where pruning of trees (leaves) is not applied. We apply the generation of the tree structure until all samples have been classified by the tree. We denote these cases as ID3-f below.

Additionally, since many logical expressions generated by the GP method are given as logical products, we try to use the conventional method to generate association rules (correlation rules). These cases are denoted as CRULE in the following.

Table 8 shows the results of comparative studies of clustering for the eight real-world datasets treated in the previous section, and for the German Credit data (denoted as G in the table) by comparing the results obtained using ID3, ID3-f, MDA, and CRULE. The values in Table 8 are the rates of true classification, that is, the probability that the prescribed cluster is the same as the cluster obtained (estimated) by each method. For simplicity, the result for the proposed GP method is omitted from the table, because the classification rate of the GP method is always 100%.

Table 8. Classification rate by ID3 and MDA (%)

method	No.1	No.2	No.3	No.4	No.5
ID3	69.1	82.3	88.6	68.7	72.4
ID3-f	100	100	100	100	100
MDA	63.1	50.5	77.7	60.7	66.2
CRULS	53.2	47.6	56.7	39.1	56.0
method	No.6	No.7	No.8	No.G	
ID3	76.0	82.3	81.5	74.9	
ID3-f	100	100	100	100	
MDA	73.9	68.0	72.6	62.4	
CRULS	48.9	65.4	51.2	43.5	

Table 9 gives a comparison of the complexity of our method in the paper (GP method) and that of the ID3, CRULE method. In the table, we show the number of nodes and leaves in the tree structure if the logical expressions for feature descriptions are represented in tree structures. In Table 9, l-GP, l-ID3, l-ID3-*f* refer to the number of leaves in the GP, ID3, ID3-*f* methods, and n-GP, n-ID3, n-ID3-*f* refer to the number of nodes in the GP, ID3, ID3-*f* methods, respectively. *a*-CRULE refers to the number of logical products in the generation of association rules.

As can be seen from the results, while the true classification rate of the GP method (our method) is 100%, the corresponding rates of the ID3 and MDA methods are lower than 100%. One reason for the worse classification performance is the method of tree structure construction: in the ID3 method, in principle the same tree structure is applied to several (all) clusters for comprehensive and simultaneous classification, which results in deteriorated classification. Additionally, in the MDA method only the linear discriminant functions are used for classification, so that the results are worse than in the cases of GP methods where logical expressions are utilized.

Table 9. Comparison of complexity in rules obtained by our method (GP) and ID3

method	No.1	No.2	No.3	No.4	No.5
l-GP	4	5	4	4	4
l-ID3	8	16	11	13	11
l-ID3-f	32	65	43	50	43
n-GP	3	4	3	3	3
n-ID3	6	11	7	8	7
n-ID3-f	27	51	31	36	34
a-CRULE	12	9	8	11	9
method	No.6	No.7	No.8	No.G	
l-GP	5	4	4	5	
l-ID3	17	11	13	22	
l-ID3-f	65	43	51	87	
n-GP	4	3	3	4	
n-ID3	12	7	8	12	
n-ID3-f	49	32	39	63	
a-CRULE	13	8	12	10	

In addition, the size of the tree structures obtained by the ID3 method is somewhat larger than the size of the trees obtained by the GP method (our method), and they are not relevant for simple description of features. In contrast, the tree structure obtained by the GP method is simple and uses only five or six leaves and nodes on average.

As can be seen from the results, even in comparative (indirect) studies assuming that the samples are classified into several clusters, we find relatively better results for the GP method than for the ID3 and MDA methods with respect to feature extraction and description.

5. Conclusions

This paper has treated the realization of retrieval and feature description systems for clusters by using logical rule generation based on GP. The GP procedure for improving logical expressions was applied to feature description of the targeted clusters. The fitness of individuals was defined in proportion to the hits of the corresponding logical expression on the samples in the targeted cluster *c*, but also in inverse proportion to the hits outside cluster *c*. As applications, the GP method was applied to artificially generated samples and various real-world data.

In the future it will be necessary to apply the method of transformation of logical expressions to natural language. Further investigations will be conducted by the authors.

Acknowledgment. This research was supported in part by the Japan Society for the Promotion of Science under grant number (B) 15310120, and the authors thank the organization.

REFERENCES

1. Piatetsky G, Frawley WJ. Knowledge discovery in database: An overview. In: Knowledge discovery in database. AIII/MIT Press; 1991.
2. Freitas AA. Data mining and knowledge discovery with evolutionary algorithms. Springer-Verlag; 2002.
3. Baesens B, Setiono R, Mues C, Vanthienen J. Using neural network rule extraction and decision tables for credit-risk evaluation. Management Science 2003;49:313–329.
4. Craven MW, Shavlik JW. Extracting tree-structured representations of trained networks. In: Touretzky D,

Mozer M, Hasselmo M (editors). Advances in Neural Information Processing Systems Vol. 8, p 24–30. MIT Press; 1996.

5. Tokinaga S, Lu J, Ikeda Y. Neural network rule extraction by using the genetic programming and its applications to explanatory classifications. IEICE Trans Fundam 2005;E88-A:2627–2635.
6. Quinlan JQ. C4.5 programming for machine learning. Morgan Kaufmann; 1993.
7. Wong ML, Leung KS. Data mining using grammar based genetic programming and applications. Kluwer Academic; 2000.
8. Lu J, Kishikawa Y, Tokinaga S. Realization of feature descriptive systems for clusters by using rule generations based on the genetic programming and its applications. IEICE Trans Fundam 2006;E89-A: 2627–2635.
9. Ikeda Y, Tokinaga S. Approximation of chaotic dynamics by using smaller number of data based upon the genetic programming. IEICE Trans Fundam 2000;E83-A:1599–1607.
10. Ikeda Y, Tokinaga S. Controlling the chaotic dynamics by using approximated system equations obtained by the genetic programming. IEICE Trans Fundam 2001;E84-A:2118–2127.
11. Yababe M, Tokinaga S. Applying the genetic programming to modeling of diffusion processes by using the CNN and its applications to the synchronization. IEICE Trans Fundam 2002;J85-A:548–559. (in Japanese)
12. Ikeda Y. Estimation of the chaotic ordinary differential equations by co-evolutional genetic programming. IEICE Trans Fundam 2002;J85-A:424–433. (in Japanese)
13. Chen X, Tokinaga S. Approximation of chaotic dynamics for input pricing at service facilities based on the GP and the control of chaos. IEICE Trans Fundam 2002;E85-A:2107–2117.
14. Chen X, Tokinaga S. Synthesis of multi-agent systems based on the co-evolutionary genetic programming and its applications to the analysis of artificial markets. IEICE Trans Fundam 2003;E86-A:1038–1048. (in Japanese)
15. Ikeda Y, Tokinaga S. Chaoticity and fractality analysis of an artificial stock market by the multi-agent systems based on the co-evolutionary genetic programming. IEICE Trans Fundam 2004;E87-A:2387–2394.
16. Lu J, Tokinaga S. An aggregated approximation for modeling of time series based on the genetic programming and its application to clustering. IEICE Trans Fundam 2005;J88-A:803–813. (in Japanese)
17. Lu J, Tokinaga S, Ikeda Y. Explanatory rule extraction based on the trained neural network and the genetic programming. Journal of the Operations Research Society of Japan 2006;149:66–82.
18. Ikeda K, Chen X, Tokinaga S. Analysis of chaotic behavior of input pricing realized by the multi-agents systems based on the C-evolutionary genetic programming and its applications. IEICE Trans Fundam 2006;J89-A:298–307. (in Japanese)
19. Koza JR. Genetic programming. MIT Press; 1992.
20. Koza JR. Genetic programming II: Automatic discovery of reusable programs. MIT Press; 1994.
21. Keith MJ, Martin MC. Genetic programming in C++: Implementation issues. In: Kinnerar KE Jr (editor). Advances in genetic programming. MIT Press; 1994.
22. http://www.liacc.up.pt/ML/statlog/datasets/german/german.doc.html
23. Han J, Micheline K. Data mining: Concepts and techniques. Morgan Kaufmann; 2000.
24. Claude S. Data mining with Microsoft SQL Server 2000 technical reference. Microsoft Press; 2001.

AUTHORS (from left to right)

Jianjun Lu (student member) received his B.S. and M.S. degrees in agricultural electrification and automation from China Agricultural University, Beijing, China, in 1997 and 2002, and a Doctor of Economics degree from Kyushu University in 2007. In 2002 he joined the faculty of China Agricultural University, where he is currently a lecturer. Since 2007 he has been a science special researcher at Kyushu University. His research interests include time series analysis, agent theory, and management information systems.

Yoshinori Kishikawa (member) received his B.S., M.S., and Ph.D. degrees in economics from Kyushu University in 1997, 1999, and 2002. Since 2002 he has been on the Faculty of System Science and Technology, Akita Prefectural University, where he is currently an assistant professor. His research interests include management information systems, fuzzy theory, and regional analysis.

Shozo Tokinaga (member) received his B.S., M.S., and Ph.D. degrees in computer science and communications engineering from Kyushu University in 1971, 1973, and 1977. From 1977 to 1979, he was on the faculty of Kitakyushu Technical College, and from 1979 to 1986 he was at Oita University as an associate professor. Since 1986 he has been with the Department of Economic Engineering, Graduate School of Economics, Kyushu University, where he is currently a professor. From 1989 to 1990 he was a visiting scholar at the University of California and the University of Texas. His research interests include management information systems, business expert systems, and digital signal processing.