Applied Artificial Intelligence, 15:825–842, 2001

0883-9514/01 $12.00 +.00

COMBINED METHOD OF GENETIC PROGRAMMING AND ASSOCIATION RULE ALGORITHM

AYAHIKO NIIMI and EIICHIRO TAZAKI
Department of Control and Systems Engineering,
Toin University of Yokohama, Japan

Genetic programming (GP) usually has a wide search space and a high flexibility. Therefore, GP may search for global optimum solution. But, in general, GPs learning speed is not so fast. An apriori algorithm is one of association rule algorithms. It can be applied to a large database. But it is difficult to define its parameters without experience. We propose a rule generation technique from a database using GP combined with an association rule algorithm. It takes rules generated by the association rule algorithm as initial individual of GP. The learning speed of GP is improved by the combined algorithm. To verify the effectiveness of the proposed method, we apply it to the decision tree construction problem from the University of California at Irvine (UCI) machine-learning repository, and rule discovery problem from the occurrence of the hypertension database. We compare the results of the proposed method with prior ones.

We now have many huge databases about various fields and we want to obtain new knowledge from them. Knowledge discovery in databases (KDD) is one of such topics. The KDD process contains the following steps (Berry & Linoff, 1997; Liu & Motoda, 1998a, 1998b):

1. data warehousing,
2. target data selection,
3. cleaning,
4. projection and reduction,
5. model selection,
6. data mining,
7. evaluation and interpretation,
8. consolidation of the discovered information.

The authors would like to express their thanks to Dr. Katusmi Yoshida at the Department of Preventive Medicine, St. Marianna University, Kawasaki, Japan, for the medical dataset for the occurrence of hypertension, and helpful discussions.

Address correspondence to Eiichiro Tazaki, Department of Control and Systems Engineering, Toin University of Yokohama, 1614 Kurogane-cho, Aoba-ku, Yokohama 225–8502, Japan.

The data warehousing step means data collecting from a lot of different sources (they are a kind of database) dependent on applying applications. Target data step selection means selecting data to make a dataset. Cleaning step means adjusting the dataset by deleting missing values and low association values. In the model selection step, we need to select the best modeling expression and model algorithms (classification or pattern-matching algorithm) for application. In the data mining step, we use some modeling algorithms (classification or pattern-matching algorithm) to extract interesting and/or useful models. The evaluation and interpretation step means validating the results and considering their meanings. The final step of consolidation means arranging the discovered information more easily to be checked and reused.

In common, each step is referred to as follows:

- data warehousing ⇒ data warehousing,
- target data selection, cleaning, projection, and reduction ⇒ preprocessing,
- model selection, data mining ⇒ data mining,
- evaluation and interpretation, consolidation ⇒ post-processing.

For data mining, various techniques have been proposed for the construction of the inference system using classification learning. In general, the learning speed of a system using a genetic programming (GP) (Koza, 1992, 1994) is slow. Moreover, both problem background knowledge and design skill are necessary for the system designer. However, a learning system which can acquire higher-order knowledge by adjusting to the environment can be constructed, because the structure is treated at the same time.

On the other hand, there is the apriori algorithm (Terabe, Katai, Sawaragi, Washio, & Motoda, 2000), a rule-generating technique for large databases. This is an association rule algorithm. The apriori algorithm uses two values for rule construction: a support value and a confidence value. Depending on the setting of each index threshold, the search space can be reduced, or the candidate number of association rules can be increased. However, experience is necessary for setting an effective threshold.

Both techniques have advantages and disadvantages as above. In this article, we propose an extended genetic programming using apriori algorithm for rule discovery. By using the combined rule generation learning method, it is expected to construct a system that can search for flexible rules in large databases.

To verify the effectiveness of the proposed method, we discuss the learning method by genetic programming combined with the association rule construction method by the apriori algorithm and the automatically defined function technique. We apply it to the decision tree construction problem from the University of California at Irvine (UCI) machine learning

repository, and the rule discovery problem from the occurrence of the hypertension database. We compare the results of the proposed method with prior ones.

ALGORITHM OF ASSOCIATION RULE CONSTRUCTION

The association rule is one of the expressions that are often used by the basket analysis. In the association rule analysis, each case generally targets the transaction form data. The transaction form data is an item set as a minimum unit of the data description. Co-occurrence pattern among item sets in the case is extracted as an association rule. Equation 1 is one of association rule samples. The association rule is expressed as a co-occurrence pattern of the item set in the case data (Kitsuregawa, 1997; Terabe et al., 2000).

$$B \Rightarrow H \tag{1}$$

B is conditions of association rule.
H is conclusions of association rule.

By the ordinary analysis technique, a lot of calculation time needs to extract the association rule from a large database. The apriori algorithm can extract the association rule from a large database by achieving the high efficiency of the search in realistic time.

In the basket analysis, there are two key principles that are called the monotonicity:

- If a set of items S is frequent (appears in at least fraction s of the baskets), then every subset of S is also frequent.
- Conversely, a set S cannot be frequent unless all its subsets are.

The following ideas can be used to find the basket's frequent itemsets in the basket analysis (Ullman, 2000).

1. Proceed levelwise, finding first the frequent items (sets of size 1), then the frequent pairs, the frequent triples, etc. In our discussion, we concentrate on finding frequent pairs because:
   (a). Often pairs are enough.
   (b). In many data sets, the hardest part is finding the pairs; proceeding to higher levels takes less time than finding frequent pairs.
2. Find all maximal frequent itemsets (i.e., sets S of any size, such that no proper superset of S is frequent) in one pass or a few passes.

The following is taken from Agrawal (1993, 1994). The algorithm is called apriori algorithm:

1. Given support threshold s, in the first pass we find the items that appear in at least fraction s of the baskets. This set is called L_1, the frequent items. Presumably, there is enough main memory to count occurrences of each item.
2. Pairs of items in L_1 become the candidate pairs C_2 for the second pass. We hope that the size of C_2 is not so large that there is not room in the main memory for an integer count per candidate pair. The pairs in C_2 whose count reaches s are the frequent pairs, L_2.
3. The candidate triples, C_3 are those sets $\{A, B, G\}$ such that all of $\{A, B\}$, $\{A, C\}$, and $\{B, C\}$ are in L_2. On the third pass, count the occurrences of triples in C_3; those with a count of at least s are the frequent triples, L_3.
4. Proceed as far as you like (or the sets become empty). L_i is the frequent sets of size i; C_{i+1} is the set of sets of size $i + 1$ such that each subset of size i is in L_i.

In the apriori algorithm, the candidate of the association rule is evaluated by using two values—support value and confidence value—while searching for the association rules. The support value of association rule R ($sup(R)$) is defined in eq. (2). The confidence value of association rule R ($sup(R)$) is defined in eq. (3).

$$sup(R : B \Rightarrow H) = \frac{n(B \cup H)}{N}. \tag{2}$$

$n(B \cup H)$ means a number of cases containing both B and H items, N means a number of all cases.

$$conf(R : B \Rightarrow H) = \frac{n(B \cup H)}{n(B)}. \tag{3}$$

$n(B \cup H)$ means a number of cases containing both B and H items, $n(B)$ means a number of cases containing B items.

The apriori algorithm, for support value and confidence value, uses minimum support value and minimum confidence value as those thresholds. If a candidate of association rule cannot fill these minimum values, we assume that its rule expresses low association. Then, its rule excludes evaluation to reduce search space consecutively. Therefore, the apriori algorithm can search faster than the other association rule analysis techniques.

By operating each minimum value, the candidate number of association rules can be increased, or the range of the search space can be reduced.

However, it is possible that an unexpected rule cannot be extracted by reducing the range of the search space. Moreover, the load of the expert who analyzes the rule increases when there are a lot of association rule candidates, and it is possible that it becomes difficult to search for a useful rule.

GENETIC PROGRAMMING

Genetic programming learning method based on the natural theory of evolution, and the flow of the algorithm is similar to genetic algorithm (GA). The difference between GP and GA is that GP has extended its chromosome to allow structural expression using function nodes and terminal nodes (Koza 1992, 1994). Using function nodes and terminal nodes, GP can search hierarchical computer programs by undergoing adaptation.

Genetic programming can be used for a search problem if its problem can be expressed as computer programs. Genetic programming's search space depends on function nodes and terminal nodes.

In using GP for a problem, it has five major preparatory steps. The five steps follow:

1. the set of terminals,
2. the set of primitive functions,
3. the fitness measure,
4. the parameters for controlling the run,
5. the method for designating a result and criterion for terminating a run.

The decision tree construction with GP is given by the following procedures:

1. An initial population is generated from a random grammar of the function nodes and terminal nodes defined for each problem domain.
2. The fitness value, which relates to the problem-solving ability for each individual of the GP population, is calculated.
3. The next generation is generated by genetic operations.
   (a). The individual is copied by fitness value (reproduction).
   (b). A new individual is generated by intersection (crossover).
   (c). A new individual is generated by random change (mutation).
4. If the termination condition is met, then the process exits. Otherwise, the process repeats from the calculation of fitness value in step 2.

In this article, the tree structure is used to express the decision tree. Therefore, we express the decision tree by using each attribute value and the class name as the terminal node and the condition sentence as the

```
RPB:
  (IFLTE A "T"
    (IFEQ B "T"
      (IFEQ C "F" N P) ADF0) P) P)

ADF0:
  (IFEQ D "F" P N)

ADF1:
  C
```

FIGURE 1. Expression of GPs chromosome.

function node (Figure 1). We defined some expressions for decision trees. In Figure 1, the decision tree is expressed like LISP-code. "RPB" is defined as the main GP tree. Both "ADF0" and "ADF1" are defined as each ADF tree. "IFLTE" and "IFEQ" are function nodes. These functions require four arguments, *arg*1, *arg*2, *arg*3, and *arg*4. Their definitions are as follows:

(IFLTE *arg*1, *arg*2, *arg*3, *arg*4) if *arg*1 is less than or equal to ($\leq$) *arg*2 then evaluate *arg*3, else then evaluate *arg*4
(IFEQ *arg*1, *arg*2, *arg*3, *arg*4) if *arg*1 is equal to (=) *arg*2 then evaluate *arg*3, else then evaluate *arg*4.

A, B, C, and D express the attribute values from database. "T" and "F" express attribute value, and "N" and "P" express class name.

Using these expressions, GP can use database attributes more easily. Moreover, GP can apply continuous attributes. However, in general, database attributes are a lot of numbers, so GPs learning speed may become slower. The definition for continuous attributes may be the cause of slow learning speed. It has a trade-off relation between learning speed and expression compatibility.

Ordinarily, there is no method of adequately controlling the growth of the tree, because GP does not evaluate the size of the tree. Therefore, during the search process, the tree may become overly deep and complex, or settle to a too simple tree. There has been research on methods to have the program define functions itself for efficient use. One of the approaches is automatic function definition (or automatically defined function (ADF)), and this is achieved by adding the gene expression for the function definition to normal GP (Koza, 1994). By implementing ADF, a more compact program can be produced and the number of generation cycles can be reduced. More than one ADF can be defined in one individual.

In addition, the growth of the tree is controlled by evaluating the size of the tree. For example, an approach based on minimum description length (MDL) has been proposed concerning the evaluation of the size of the tree. In this article, we used the classification success ratio ($f_{hits(n)}$) and the number of composed nodes ($f_{nodes(n)}$) to define the fitness of the individual ($f_{fitness(n)}$):

$$f_{fitness(n)} = \alpha f_{hits(n)} + (1 - \alpha) f_{nodes(n)}$$

α is weight defined by $(0 \leqq \alpha \leqq 1)$.

Here, two things are expected. One is that accuracy of the decision tree is raised while avoiding overtraining in GP. And the other is that the decision tree generated can be made comparatively compact. The fitness value and rule size of the best individual is influenced by weighting the success rate or node size. We set weight α by pre-experiment.

APPROACH OF PROPOSED COMBINED LEARNING

The apriori algorithm can be applied to a large database. But it is difficult to define its parameters without experience. On the other hand, a high classification ability is obtained by GP through training structural information, but more learning time is needed as the degree of learning freedom increases.

It is observed that the discursive accuracy of classification becomes highly improved by the emergent property of interaction between two combined methods.

To make up for the advantages and disadvantages of the apriori algorithm and GP, we propose a rule discovery technique which combines GP with the apriori algorithm. By combining each technique, searching for flexible rules from a large database is expected. An outline of our proposed technique is shown in Figure 2.

The following steps are proposed for the rule discovery technique:

1. The apriori algorithm generates the association rule.
2. The generated association rules are converted into decision trees, which are taken in as initial individuals of GP. The decision trees are trained by GP learning.
3. The final decision tree is converted into classification rules.

This allows effective schema to be contained in the initial individuals of GP. As a result, improvement of the GPs learning speed and its classification accuracy is expected.

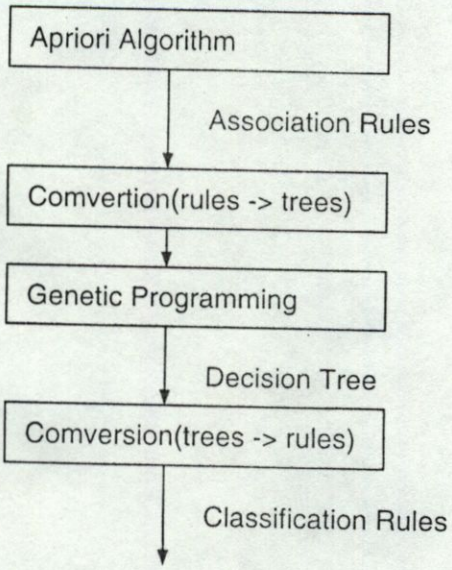

FIGURE 2. Flow chart of approach of proposed combined learning.

For conversion from the association rules into decision trees, we use the following procedures:

1. For the first process, the route of the decision tree is constructed, assuming the conditions of the association rule as the attribute-based tests of the decision tree.
2. In the next process, the conclusions of the association rule is appended on the terminal node of this route.
3. Finally, the terminal nodes, which are not defined by the association rule, are assigned candidate nodes at random.

In this conversion, one decision tree is converted by one rule. When the decision tree is taken into the initial population of GP, it is necessary that variety in the initial population is not upheld (Niimi & Tazaki, 1999a). Because an apriori algorithm can make a lot of rules, it is easy to maintain the variety of an initial population of GP. In later GP learning, the accuracy of the decision tree need not be so high, though it is possible to convert the decision tree with higher accuracy from more rules. Moreover, because the rules only have to be able to be extracted for the GPs initial population, the number of rules is not so necessary.

In this GP, the classification system is expressed by a decision tree. If you need decision rules, it is easy to convert from trees to rules by the process proposed by Quinlan (1993).

The key ideas of Quinlan's conversion process follow:

- Every path from the root of an unpruned' tree to a leaf gives one initial rule. The left-hand side of the rule contains all the conditions established by the path, and the right-hand side specifies the class at the leaf.
- Each such rule is simplified by removing conditions that do not seem helpful for discriminating the nominated class from other classes, using a pessimistic estimate of the accuracy of the rule.
- For each class in turn, all the simplified rules for that class are sifted to remove rules that do not contribute to the accuracy of the set of rules as a whole.
- The sets of rules for the classes are then ordered to minimize false positive errors and a default class are chosen.

In the proposed technique, a large number of rules made from an apriori algorithm can be brought together by GP. The rules are made from an apriori algorithm, the decision tree is learned by GP, and the result is converted into the rules. This algorithm seems to cause the overhead in this operation seemingly by a lot of conversions. But a decision tree can express the features of all data. On the other hand, it is difficult to express the features of all the data by using a rule with the exception of using rule sets. Genetic programming is evaluated by all training data checking as one element of fitness function. One GPs individual need to express the classification rule set of all data, then we use GP as a decision tree. If you need classification rules, it is easy to use GP where classification rules are converted from a decision tree after a GP decision tree learning. Moreover, because GP is learned after the rules are extracted, the rule that contains a continuous value can be easily learned. The rule that contains a continuous value is learned by the change of the threshold in the function node of GP.

The conversion from GPs tree to rules is an effective approach as post-processing of data mining. Genetic programming's tree often contains meaningless rules or useless rules because GPs structure expression has high flexibility and main GP operations are based on probability operations. For this problem, many approaches have been proposed, for example, a modified scheme of GPs individual expression, expanded genetic operation, pruning operation on GP learning (Niimi & Tazaki, 1999c). But these techniques may become overhead operation on GP learning. In our proposed method, GPs initial population is improved and unless search is reduced, by taking apriori algorithm results into GPs initial population. In addition, conversion from a decision tree to classification rules as post-processing can remove unable expressed rules and unable explained rules.

For conversion from a decision tree to classification rules as post-processing, we propose the following steps:

1. prune meaningless subtree in decision tree by GP,
2. convert from tree to rules using C4.5 rule conversion method,
3. remove meaningless rules or unable explained rules,
4. check the rules by expert.

Some examples of step 1 are "*ifA = Athen . . .*," "*ifA = BthenCelseC.*" An example of step 2 is comparative equation which is overflowed input data range.

The other consideration is calculation increasing by combined method. Genetic programming is a most heavy calculation in the following four calculations: apriori algorithm, conversion from rules to GP, GP, conversion from GP to rules. The GPs calculation is far heavier than the other calculations. For this reason, the increase of calculation in the combined method is capable of being actually ignored by GP calculation. For implementation of GP, both the set of terminals and the set of functions can be automatically defined by the apriori algorithm's result. For fitness function, general definition of an evaluation tree can be used. By this way, our proposed method makes GPs implementation more easily, and the implementation times can be reduced.

APPLY TO DECISION TREE CONSTRUCTION FROM DATABASE

To verify the validity of the proposed method, we applied it to the house-votes data from the UCI machine learning respository (Blake & Merz, 1998), and medical database for occurrence of hypertension (Ichimura, Oba, et al. 1997; Niimi & Tazaki, 1999b). From here on, all occurrence of GP uses automatically defined function genetic programming (ADF-GP) (Koza, 1994) including the proposed method. In the proposed method, we took the association rule generated by apriori algorithm as initial individuals of GP. We compared the results of the proposed method against GP. We did not compare GP with the apriori algorithm because of the difference of the expression such as rules and decision trees. We use house votes database as a small test database expressed by discrete values, and a hypertension database as large test expressed by continuous values. Validity of the expert (doctor) had to be evaluated to the result concerning the experiment on hypertension.

Application to House Votes Data

For evaluation, we used house votes data from the UCI machine learning respository (Blake & Merz, 1998). We compared the results of the proposed method with GP. The evaluation data contains 16 attributes and two classes.

The attributes are, for example "handicapped-infants" and "water-project-cost-sharing," etc. They are expressed by three values: "y," "n," and "?." And the two classes are "democrat" and "republican." Fifty cases out of the total 435 data of house votes were used for training data.

We extracted the association rule from the database by the apriori algorithm. We applied the apriori algorithm to a data set excluding data with the "?" value, because "?" value means "others." In the following experiment, we used minimum support value ($=30$) and minimum confidence value ($=90$). As a result of the experiment, 75 rules were generated.

Next, the above generated 75 rules were taken into the initial individual. Table 1 was used concerning the parameter of GP in each experience. In 1, GP makes "y" and "democrat," "n" and "republican" to the same treatment as dividing attributes because of mounting. The result of the evaluation of each fitness value is shown in Figure 3, and the results of the best individual is shown in Table 2. The best decision tree of each method is shown by figures 4 and 5. These tables do not contain useless branches using post-processing.

By using GP, fitness value did not improve rapidly. However, the proposed method showed fast learning and achieved high accuracy. Comparing the best individual results, the proposed method showed better results than GP, except for accuracy against the training data. Concerning the results of training data, GP may have shown overfitting. Furthermore, in Figure 3, GPs fitness value did not change at 100–300 generations. We consider this no-learning term was caused by overfitting. But proof could not be obtained by this result only.

The rules were converted from the constructed decision tree removing invalid and meaningless rules. The rules' total accuracy was 94.8%.

TABLE 1 Parameters of GP (House Votes)

GP population	500
Reproduction probability	0.1
Intersection probability	0.8
Mutation probability	0.1
Selection method	Tournament method
Function node	IFEQ, ADF0, ADF1
Terminal node	Attribute value of database (16 attributes), y, n, ?, democrat, republican
Number of training data	democrat:31, republican:19
Number of test data	435

IFEQ: if equal to ($=$).

ADF0, ADF1: the function definition gene expanded by ADF.

y, n, ?: yes(y), no(n), others(?).

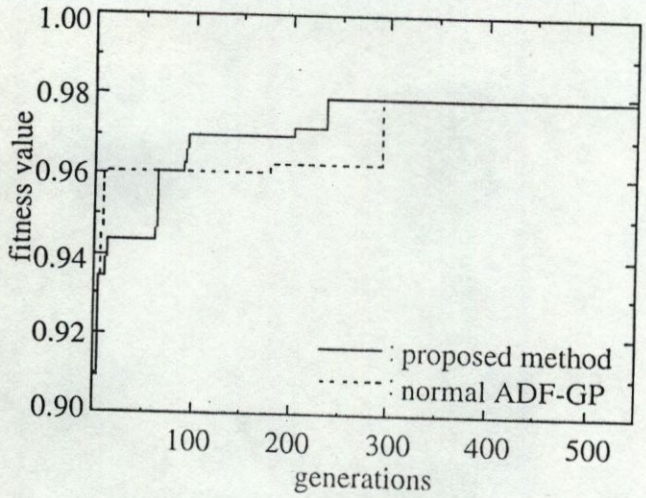

FIGURE 3. Evaluation of each fitness value.

TABLE 2 Experiment Best Individuals Result (House Votes)

	Training (%)	All Data (%)	Nodes	Depths	Generations
ADF-GP	100.0	86.0	11	3	586
Apriori + ADF-GP	98.0	92.9	9	2	235

```
RPB:
    (IFEQ y physician-free-freeze
        (IFEQ physician-fee-freeze mx-missile
            ADF0 adoption-of-the-budget-resoltion)
        y)
ADF0:
    handicapped-infants
ADF1:
    el-salvador-aid
```

FIGURE 4. The result from ADF-GP (house votes).

```
RPB:
    (IFEQ mx-missile physician-free-freeze synfuels-corporation-cutback
        (IFEQ mx-missile y
            mx-missile adoption-of-the-budget-resolution)

ADF0:

    handicapped-infants

ADF1:

    democrat
```

FIGURE 5. The result from apriori + ADF-GP (house votes).

Application to Medical Database

We applied a medical diagnostic system for the occurrence of hypertension. We compared the results of a proposed method with GP. Most of the data values are expressed as continuous values, and the size of the database is larger than the house votes database. The domain expert gave the evaluation and comments on the results.

The occurrence of hypertension database contains 15 input terms and one output term. There are two kinds of intermediate assumptions between the input terms and the output term (Ichimura et al., 1997; Niimi & Tazaki, 1999b). Among the input terms, 10 terms are categorized into a biochemical test related to the measurement of blood pressure for past 5 years, and the other terms are "Sex," "Age," "Obesity Index," "γ-GTP," and "Volume of Alcohol Consumption." One output term represents whether the patient has an attack of hypertension for the input record. The database has 1024 patient records. In this article, we selected 100 occurrence data and 100 no-occurrence data by random sampling, and this was used as the training data.

The association rule has been extracted from the database by the apriori algorithm. The apriori algorithm was used after these attributes had been converted into binary attributes using the average of each data because the continuous value attributes were included in this database. To search for the relationship between the minimum support value and the minimum confidence value and the number of rules, we experimented with the threshold patterns (refer to the results in Table 3). In the following experiment, we used minimum support value ($=30$) and minimum confidence value ($=90$).

Next, the 33 rules generated by the apriori algorithm were taken into the initial individual. The continuous values are learned at the same time by the change of the threshold of the function node while GP learning. Table 4 was used concerning the parameter of GP in each experience. "P" and "1," "N" and "0" do the same treatment by convenience in mounting in GP. The result of the best individual is shown in Table 5. The best decision tree of each

TABLE 3 Relations Between Thresholds and Number of Rules

Minimum Support Value	Minimum confidence Value	Rules
25	75	396
30	75	125
25	90	187
30	90	33

TABLE 4 Parameters of GP (Hypertension)

GP population	500
Reproduction probability	0.1
Intersection probability	0.8
Mutation probability	0.1
Selection method	Tournament method Elitist strategy
Function node	IFLTH, IFEQ, *, /, +, −, ADF0, ADF1
Terminal node	Attribute value of database such as SEX and AGE (15 attributes), P, N, R
Number of training data	Occurrence (100), No-occurrence (100)
Number of test data	1024
Maximum generations	20000

IFLTH, IFEQ: if less than or equal to (≤), if equal to (=).
ADF0, ADF1: the function definition gene expanded by ADF.
R: randomly generated constant.
P, N: occurrence (P), no-occurrence (N).

TABLE 5 Experiment Best Individuals Result (Hypertension)

	Training (%)	All Data (%)	Nodes	Depths	Generations
ADF-GP	89.5	66.3	41	6	18553
Apriori + ADF-GP	89.5	74.9	49	4	671

method is shown by Figures 6 and 7. These tables do not contain useless branches using post-processing.

By using GP, fitness value did not improve rapidly. However, the proposed method showed fast learning and achieved high accuracy.

When the rules were converted from the decision tree, invalid and meaningless rules were removed. Each ratio of the number of effective rules to generation rules was 37.5% (by GP) and 50.0% (by proposed method). Table 6 shows three rules generated by each technique, chosen by the highest support value.)

```
RPB:
    (/ N
       (- ADF1 Kaku2))
ADF0:
    (/ 152 Sex)
ADF1:
    (IFLTE (IFEQ Syu4 Syu2 P Kaku3)
           (/ (- 120 Age) ADF0) Syu2
           (IFLTE Kaku4
                  (- 120 Age)
                  (IFLTE (- Syu2 Kaku4) Kaku4 Syu2
                         (IFLTE Kaku4
                                (/ (- Syu2 Kaku4) ADF0) Syu4 Kaku2)) Kaku2))
```

FIGURE 6. The result from ADF-GP (hypertension).

```
RPB:
    (IFLTE 45 Age
           (IFLTE 76 Kaku4 P
                  (IFLTE 75 Kaku5
                         (IFLTE 75 Kaku2 P N) N))
           (IFLTE (- (IFEQ ADF1 Sex Age Kaku1)
                     (/ Kaku2 Sake)) Kaku5
                  (IFLTE (IFLTE 76 Kaku4 P 90) Kaku5
                         (IFLTE 75 Kaku2 P N) N) N))
ADF0:
    (/ Syu4 Kaku2)
ADF1:
    (/ Kaku3
       (+ Kaku5 P))
```

FIGURE 7. The result from apriori + ADF-GP (hypertension).

TABLE 6 Comparison of Generated Rules (Size and Fitness Value)

Technique	Size	Support Value (%)	Wrong (%)
ADF-GP	4	41.4	48.4
	2	36.0	24.1
	4	22.6	8.2
Apriori + ADF-GP	2	17.7	39.2
	4	15.5	13.2
	4	13.8	19.2

By using GP, many invalid rules, and many rules that were difficult to interpret were generated. Compared to GP, the proposed method showed a decrease in the support value and improvement in accuracy. The proposed technique improved the ratio of effective rules and accuracy.

Discussion for the Results

In both of the two experiments, the accuracy of the decision tree, the size, and the learning speed of our proposed method obtained better results than normal GPs. The table of the results shows only the generation cycle of GP, but the calculation time of apriori algorithm and the time for conversion process from apriori algorithm to GP are shorter than the calculation time for one generation of GP. Because of the results of this experiment, an increasing calculation by the combination was not seen.

When the rules were extracted from the results of both experiments generated decision trees, it was very difficult to extract as post-processing. It is thought that the reason is that a lot of rules with a difficult interpretation are included by the probability operation of GP. However, the threshold to which the proposed technique is more significant than normal GP, is oftentimes learned in the learning of continuous values in the experiment on hypertension. Automatically defined function was not used too effectively concerning its use in the experiment on house votes. The reason is that the small decision tree had enough size for the problem because the problem was too easy to make the decision tree.

For experimental results in hypertension the expert comment is "appropriateable result." Another comment is that more interesting rules were contained by normal GPs results than the proposed methods. This comment is expressed because GP rules include some interesting attributes (such as sex). A expert does not consider such attributes (sex, etc.) in the diagnosis, but it can be considered as background knowledge. The result is that the idea of an unexpected rule comes out easily by GPs probability operations. However, it seems that GPs probability operations could not present the effect by the influence from the initial individual taken from an apriori algorithm in the proposed method. In general, unexpected rules can be applied only in a small number of data. Therefore, it is thought that our fitness function cannot express unexpected rules well. The accuracy of an unexpected rule and the decision tree becomes the relation of the trade-off. It is necessary to design fitness function, which can strongly express the unexpectedness to generate an unexpected rule by using the proposed method.

In our proposed method, it is not necessary to consider the inside of each algorithm. We think of an apriori algorithm as the technique to extract the association rules with the database and GP as the technique which can take the decision tree and construct it with the database. Therefore, even if

other association rules extracting techniques are used instead of an apriori algorithm, the proposed method can be applied. It is believed that only considering the method combination without considering each internal algorithm, it is possible to construct the combination learning method by using the proposed method's analogy.

CONCLUSIONS

In this article, we proposed the rule discovery technique from the database using GP combined with apriori algorithms. The proposed method has three steps: (1) using an apriori algorithm, extract association rules; (2) Generate the GP population which includes initial individuals converted from the association rules; train the genetic program to construct the classification system.

The proposed method has some advantages The first advantage is that the proposed method can improve the decision tree's accuracy. The second advantage is that its learning speed becomes faster than normal GPs. The third advantage is that the design of GP becomes easy. And it is possible to correspond also to datasets with a lot of numbers of data and attributes which contain the continuous value that is not treated easily by normal GP. It can be considered that method combination makes calculation heavier.

We experimented by using two kinds of datasets to verify the proposed method. For one experiment, the domain expert gave the evaluation and comments on the results. Two problems were discussed. One is intended for dataset with small test database expressed by discrete values. The other is intended for dataset which contained a lot of continuous values. We compared the results of these methods using ADF-GP only, and using combined apriori algorithm and ADF-GP using the proposed method.

In the proposed method, the decision tree has been improved compared with GP and the apriori algorithm. The number of generations until the best individual has been decreased. The combination overhead was disregarded because of the speed improvement of GP. Though the result of suppressing the overfitting was seen, we did not consider the analysis of its mechanism.

In the future, we will research the following four topics: apply the method to other verifications (Niimi & Tazaki, 2000a, 2000b, 2000c); discuss the conversion algorithm from the association rule to a decision tree with high accuracy; extend the proposed method to multivalue classification problems; study a theoretical analysis about the mechanism of the overfitting.

REFERENCES

Agrawal, R., T. Imielinski, and A. Swami. 1993. Mining associations between sets of items in massive databases. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp 207–216, Washington, DC.

Agrawal, R., and R. Srikant. 1994. Fast algorithms for mining association rules. In *Proceedings 20th International Conference on Very Large Databases*, Santiago, Chile.

Berry, M. J. A., and G. S. Linoff. 1997. *Data mining techniques: For marketing, sales, and customer support*. John Wiley & Sons.

Blake, C. L., and C. J. Merz. 1998. UCI Repository of machine learning databases. http://www.ics.uci.edu/~mlearn/MLRepository.html. Irvine, CA: University of California, Department of Information and Computer Science.

Ichimura, T., K. Oba, E. Tazaki, H. Takahashi, and K. Yoshida. 1997. Knowledge based approach to structure level adaptation of neural networks. In *Proceedings of IEEE International Conference on Systems, Man, and Cybernetics (SMC'97)*, 548–553, Orlando, Florida.

Kitsuregawa, M. 1997. Mining algorithms for association rules. *Journal of Japanese Society for Artificial Intelligence* 12(4):513–520.

Koza, J. R. 1992. *Genetic programming*. Cambridge: MIT Press.

Koza, J. R. 1994. *Genetic programming II*. Cambridge: MIT Press.

Koza, J. R. 1994. Scalable learning in genetic programming using automatic function definition. Advances in genetic programming, 99–117.

Liu, H., and H. Motoda. 1998a. *Feature selection for knowledge discovery and data mining*. Kluwer Academic Publishers.

Liu, H., and H. Motoda. 1998b. *Feature extraction, construction and selection: A data mining perspective*. Kluwer Academic Publishers.

Niimi, A., and E. Tazaki. 1999a. Combined learning method of decision tree and ADF GP by object oriented approach. In *Proceedings of 42nd Knowledge Based System Meeting*, Japan AI Society, 75–82.

Niimi, A., and E. Tazaki. 1999b. Object oriented approach to combined learning of decision tree and ADF GP. In *International Joint Conference on Neural Networks*, Washington D.C.

Niimi, A., and E. Tazaki. 1999c. Extended genetic programming using reinforcement learning operation. In *Proceedings of IEEE International Conference on Systems, Man, and Cybernetics (SMC'99)*, 596–600, Tokyo, Japan.

Niimi, A., and E. Tazaki. 2000a. Genetic programming combined with association rule algorithm for decision tree construction. In *Proceedings of the Fourth International Conference on Knowledge-Based Intelligent Engineering System & Allied Technologies*, Volume 2, 746–749.

Niimi, A., and E. Tazaki. 2000b. Knowledge discovery using extended genetic programming from biochemical data. In *Proceedings of 49th Knowledge Based System Meeting*, Japan AI Society, SIG-KBS-A002, 45–49.

Niimi, A., and E. Tazaki. 2000c. Rule discovery technique using genetic programming combined with apriori algorithm. In *Proceedings of the Third International Conference on Discovery Science*, 273–277, Kyoto, Japan.

Quinlan, J. R. 1993. *C4.5: Programs for machine learning*. San Mateo, CA: Morgan Kaufman Publishers.

Terabe, M., O. Katai, T. Sawaragi, T. Washio, and H. Motoda. 2000. Attribute generation based on association rules. *Journal of Japanese Society for Artificial Intelligence* 15(1):187–197.

Ullman, J. D. 2000. A survey of association-rule mining. In *Proceedings of the Third International Conference on Discovery Science*, 1–14, Kyoto, Japan.