Developing a GP-based Framework for Knowledge Integration

[1] Chan-Sheng Kuo, [2]Tzung-Pei Hong, [3]Chuen-Lung Chen
*[1], *Department of Information Management, Kainan University, Taoyuan, 338, Taiwan, R.O.C., ecskuo@gmail.com*
[2], *Department of Computer Science and Information Engineering, National University of Kaohsiung, Kaohsiung, 811, Taiwan, R.O.C., tphong@nuk.edu.tw*
[3], *Department of Management Information Systems, National Chengchi University, Taipei, 116, Taiwan, R.O.C., chencl@mis.nccu.edu.tw*

Abstract

Knowledge integration is one of the important tasks for applying knowledge management in an organization to improve organizational performance and competitive competence. In this paper, we have proposed a GP-based knowledge-integration framework that automatically combines multiple rule sets into one integrated knowledge base. The proposed framework consists of three phases: knowledge collection and translation, knowledge integration, and knowledge output. In the collection and translation phase, each knowledge source is obtained and expressed as a rule set and then translated as a classification tree. In the integration phase, the genetic programming technique is used to generate a nearly optimal classification tree. In the output phase, the final derived classification tree is transferred as a rule set, then output to the knowledge base to facilitate the inference of new data. Two new genetic operators, abridgement and compromise, are designed in the proposed approach to remove redundancy, subsumption and contradiction, thus producing more accurate and concise classification rules than that without using them. Experimental results from diagnosis of breast cancer also show the feasibility of the proposed algorithm.

Keywords: *Genetic Programming, Knowledge Integration, Knowledge Base, Genetic Operator*

1. Introduction

Many organizations increasingly consider that knowledge is an essential asset and an important source of the core competence. Knowledge management can be used for effective management of organizational knowledge to raise organizational performance through knowledge management processes. A knowledge management system (KMS) plays a role of information technique and is mainly adopted to support creation, integration, dissemination and application of knowledge in an organization [3][17]. Construction of a complete and consistent knowledge base is significantly needed for developing an efficient KMS. The knowledge required to build a knowledge base for an application field is often distributed among different expert groups or cross-functional project teams [7][11][18]. Especially, the needed knowledge is generally distributed among multiple sources for some complex application problems. Deriving and integrating multiple knowledge sources from some experts or by various data mining techniques [2][6][13][21][22][23] is thus a critical task in building a complete knowledge base for a KMS to support daily operations of an organization. The benefits through integrating multiple knowledge sources for a knowledge base are described below [18].

(a) The resulting knowledge base acquires more complete and consistent quality;
(b) The acquired knowledge has better validity and comprehension;
(c) The Integrated knowledge can properly deal with more complex situations and problems than individual knowledge.

Knowledge integration is accordingly one of the key issues in knowledge management. It has been often viewed as a critical source of innovative competence or competitive advantage for an organization in a dynamic market environment.

Recently, genetic programming has been very popular and widely used in many application domains. It is an evolutionary approach and can also be used to discover useful classification rules as well as others [5]. In the past, Kuo et al. proposed a learning algorithm based on genetic programming to automatically find an appropriate knowledge base [16]. In this paper, we expand that approach to knowledge integration. We propose a GP-based knowledge-integration framework that can automatically combine multiple rule sets into one integrated knowledge base for a KMS to support practical applications in an enterprise. The proposed approach adds a mechanism for knowledge acquisition before the evolutionary process begins. Besides, two new genetic operators, abridgement and compromise, are also designed in the proposed approach for removing redundancy, subsumption and contradiction among the rules.

The remaining parts of this paper are organized as follows. Related research is first briefly reviewed in Section 2. A GP-based knowledge integration framework is designed in Section 3. An algorithm based on GP for knowledge integration is proposed in Section 4. The details of the proposed algorithm are described in Section 5. Experiments to demonstrate the performance of the proposed algorithm are reported in Section 6. Conclusions are given in Section 7.

2. Related research

Knowledge management (KM) is the modern method to facilitate an enterprise for managing important knowledge and provide useful knowledge needed for workers at the right moment to raise their productivity as well as creativeness. The intent of KM is to emphasize knowledge flows and main processes of acquisition, integration, storage/categorization, dissemination, and application[3][17][27]. One of the key processes in KM is knowledge integration.

The valuable knowledge undeveloped in an enterprise can come from multiple sources including worker's experience ,business processes, operating rules, cases, expert's knowledge, cross-functional teams and external information. The knowledge sources might be directly acquired through an interview, a knowledge acquisition tool or a data mining method. [2][6][7][12][13][21][22][23] Knowledge integration can be then used to combine multiple knowledge sources into one integrated knowledge source and maintains consistency of knowledge in an organization. As enterprises face decision making situations that require knowledge come from several major sources, they can utilize knowledge integration through information technology to solve redundancy and contradiction problems among them, thus producing more consistent and complete organizational knowledge to help decision making [4][11][27].

Some studied have used extant information technologies to develop knowledge integration such as ontology, genetic algorithm, intelligent agent, data warehouse, etc. [10][20][24][26] The Ontology can be used to provide accepted terms for different users and represent acquired knowledge during knowledge integration [10]. Wang et al. developed a self-integrating knowledge-based brain-tumor diagnostic system and proposed a knowledge-integration strategy that automatically integrates multiple rule sets by genetic algorithm [25][26]. Furthermore, intelligent agents can be used to analyze the relationships of knowledge and combining different knowledge to form the appropriate summaries for users [24].

Genetic programming (GP) was firstly proposed by Koza [14] and adopted some genetic operations as its basis for creating computer programs. It is an extension to genetic algorithms by using variable-length trees instead of fixed-sized chromosomes. An individual in GP is a tree structure consisting of functions and terminals and can be viewed as a potential solution to a problem.

There are five major preparatory steps before GP is applied to solving a problem [14]. They include the preparation of the following: (a) a set of terminals, (b) a set of functions, (c) a fitness measure, (d) parameters for controlling the process, and, (e) criteria for terminating the process. Moreover, there are three fundamental genetic operators, which are reproduction, crossover and mutation in GP[15]. GP is an evolutionary technique and begins with a population that is composed of a set of random created or given trees by users. It chooses suitable trees for genetic operations, thus gradually creating good offspring. The offspring then undergo repetitive evolution until the termination criterion is met. Eventually, a good tree, which represents a good solution for the problem, can be found.

In addition, users can modify basic operations or create new operations according to the problem domain to steer the evolving results closer to those desired [15]. For example, the new decimation operation is designed to remove individuals which have a low-fitness value for a percentage within a population. This, however, will usually take more time for the evolving process than using only the basic operators.

Genetic programming can extract valuable relationships and has been applied to several applications such as finance forecasting, economic models, symbolic regression, medical applications, software engineering, hand written digit recognition, and among others [1][5][9][14][19]. Due to a rule set can be represented by a classification tree, GP is thus appropriate for rule-based integration problems. In this paper, we will try to develop an extended GP strategy for knowledge integration of rule sets from multiple knowledge sources.

3. A GP-based knowledge integration framework

The proposed knowledge integration framework based on genetic programming is shown in Figure 1. It integrates multiple rule sets into one integrated knowledge base with reduced intervention of domain experts for practical applications. In this paper, we assume all knowledge sources are represented by rules. The individual knowledge from each source might be directly acquired by an expert group using a knowledge acquisition tool [7][13], or derived by a data mining method [2][6][21][22]. Genetic programming is used to maintain a population of classification trees with each representing a possible rule set and searches for the best integrated tree.

The proposed framework consists of three phases: knowledge collection and translation, knowledge integration, and knowledge output. In the knowledge collection and translation phase, each knowledge source is obtained and expressed as a rule set and then translated as a classification tree. The combined classification trees from multiple knowledge sources form an initial knowledge population, which is then ready for integration. In the knowledge integration phase, the genetic programming technique is used to generate a nearly optimal classification tree. In the knowledge output phase, the final best classification tree is transferred as a rule set, and then output to one centralized knowledge base to support decision making and facilitate daily operations in an organization.

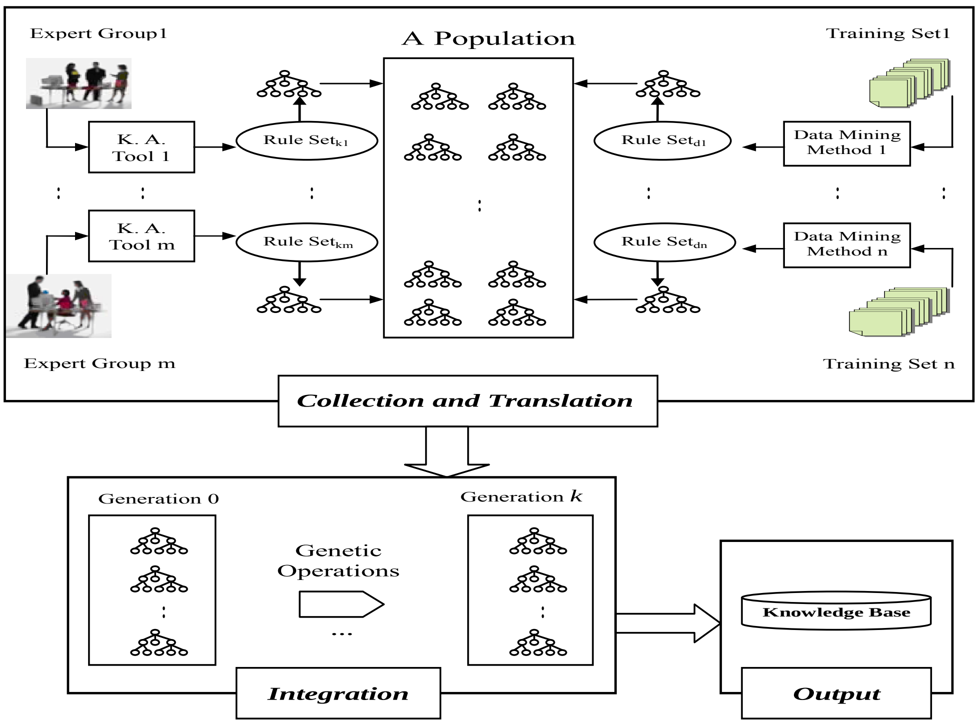

Figure 1. A GP-based knowledge integration framework

4. The proposed GP algorithm for knowledge integration

The proposed GP algorithm for knowledge integration (GPKI) is stated in this section to show how an integrated classification tree can be found from individual knowledge sources. It first chooses suitable classification trees for genetic operations, thus gradually creating favorable offspring. Two new genetic operators, abridgement and compromise, are also designed in the proposed algorithm to remove redundancy, subsumption, and contradiction. The offspring then undergoes repetitive evolution until the termination criterion is met. The proposed algorithm is described as follows.

```
The proposed GPKI algorithm:
Input: k knowledge sources and a set of instances to be classified.
Output: one integrated knowledge base.

Knowledge collection and translation phase:
  Step 1: Collect k rule sets from different knowledge sources.
  Step 2: Translate each rule set among different knowledge sources into a classification tree that
          will act as an individual in the initial population.

Knowledge integration phase:
  Step 1: Evaluate the fitness value of each classification tree by the predefined evaluation function
          and the set of instances.
  Step 2: Select suitable classification trees according to the evaluation results to perform the
          following genetic operations:
    Step 2.1: Perform crossover operations on parent trees to generate offspring trees;
    Step 2.2: Perform mutation operations on parent trees to generate offspring trees;
    Step 2.3: Perform abridgement operations on offspring trees to remove redundancy and
              subsumption;
    Step 2.4: Perform compromise operations on offspring trees to remove contradiction.
  Step 3: Evaluate the fitness value of each resulting offspring tree by the evaluation function and
          the set of instances.
  Step 4: If the termination criterion is not satisfied, then go to Step 2; otherwise, proceed to the next
          step.
  Step 5: Select the best classification tree with the highest fitness value from the population as the
          final classification tree.

Knowledge output phase:
  Step 1: Transfer the final classification tree into a rule set.
  Step 2: Feed the rule set into the centralized knowledge base of an organization.
```

The two new genetic operators, abridgement and compromise, can reduce the tree complexity and raise its accuracy. It will, however, take more execution time than using only the original operators. Trade-off thus exists between the tree concision and the time complexity. Some details about the execution of the algorithm are stated below.

5. Algorithm details

5.1. The initial population

The technique of genetic programming requires a population of feasible solutions to be initialized and updated during the evolving process. Each individual within the population is a classification tree consisting of functions and terminals in our approach. The functions used in this paper are as follows:

(a) Boolean Operations: {And, Or, Not, >, <, ≧, ≦}, and
(b) Conditional Operators: {If-Then, If-Then-Else}.

In this paper, the needed knowledge of the k rule sets comes from multiple knowledge sources. Each rule set is then translated as a classification tree. The combined classification trees from the k rule sets form an initial knowledge population. Below, a simple example is given to illustrate the concept.

Example 1. Assume in applying for a consumer loan, two classes {rejection, approval} represented as (R {R0, R1}), are to be distinguished by the four features {Annual income, Seniority, Home ownership, Marital status}. Assume the feature of Annual income has four possible values {< 200 thousand (NT dollars), 200-500 thousand, 501-800 thousand, >800 thousand} represented as (I {I1, I2, I3, I4}), the feature of Seniority has four possible values {< 1 year, 1-3 years, 4-8 years, > 8 years) represented as (S {S1, S2, S3, S4}), the feature of Home ownership has three possible values {rented, owned and mortgaged, owned} represented as (H {H1, H2, H3}), and the feature of Marital status has two possible values {single, married} represented as (M {M1, M2}). To illustrate the tree representation clearly and simply, assume that a rule set RS_k has only the following two rules:

Rule 1: If (Annual income = 501-800 thousand) and (Seniority = 4-8 years) and (Marital status = married) then Class is approval;
Rule 2: If (Annual income < 200 thousand) and (Seniority = 1-3 years) and (Home ownership = rented) then Class is rejection.

The two rules are equivalent to the following representation:

Rule 1': If I3 and S3 and M2 then R1;
Rule 2': If I1 and S2 and H1 then R0.

The two rules can be represented as a classification tree shown in Figure 2. The internal nodes of the classification tree are labeled with operations or functions (e.g. And and IF); the leaf nodes are labeled with the feature values (e.g. I3, S3, M2, I1, S2, and H1) and the classes (e.g. R1 and R0). The symbol E represents the end of the rule set.

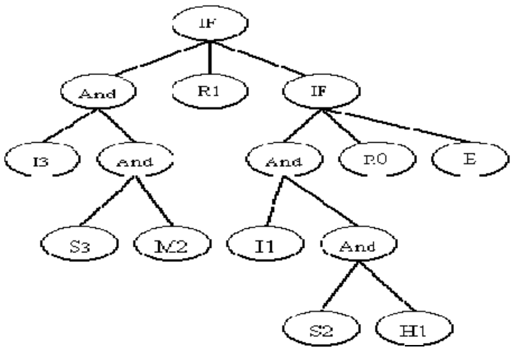

Figure 2. A classification tree representation for the example

5.2. The fitness function

An appropriate fitness function is very critical for genetic programming to work. It is used to reward good individuals or punish bad ones. In order to develop a good knowledge base from an initial population of classification trees, the genetic programming approach selects parent classification trees with high fitness values for mating. An objective evaluation function and a set of training instances are used to qualify the classification trees. In the past, Wang et al. combined two important factors, accuracy and complexity, in evaluating rule sets by genetic algorithms [26]. In this paper, we adopt and modify the evaluation function to evaluate a classification tree (CT) by genetic programming. The accuracy of a classification tree CT_q is evaluated by the training instances and is defined as follows:

$$\text{Accuracy}(CT_q) = Cor_q \ / \ n \ ,$$

where Cor_q is the number of training instances correctly matched by CT_q, and n is the number of training instances. The higher the Accuracy(CT_q), the better the CT_q.

The complexity of a classification tree CT_q is the ratio of nodes used, which is defined as follows:

$$\text{Complexity}(CT_q) = Un_q \; / \; Ave ,$$

where Un_q is the number of nodes used within CT_q and Ave is the average number of nodes of an individual within the population. A higher Complexity(CT_q) value means the tree is more complex (with more nodes).

Accuracy and complexity are combined to evaluate the fitness value of a classification tree CT_q. The evaluation function is thus defined as follows:

$$\text{Fitness}(CT_q) = \text{Accuracy}(CT_q) \; / \; \text{Complexity}(CT_q)^{\mu},$$

where μ is a control parameter representing a tradeoff between accuracy and complexity. The fitness value of CT_q can thus represent the suitable degree for the integrating goal. As Fitness(CT_q) has a higher value, CT_q is better. In the paper, we will find a good integrated classification tree for multiple rule sets by GP based on the above criterion.

5.3. Genetic operators

In this paper, two fundamental genetic operators, crossover and mutation, and two new operators, abridgement and compromise, are used in the proposed algorithm. The abridgement and compromise operators are designed to solve the problems such as redundancy, subsumption, and contradiction [8].

5.3.1. Crossover

In the crossover operation, two parent trees are partially exchanged to form two new offspring trees. The crossover point may occur within a rule or on rule boundaries.

5.3.2. Mutation

The mutation operation is designed to help the evolving process escape from local optimum. It begins by selecting a parent tree from the population at random. A node within the tree is randomly selected. The mutation operation then replaces the selected node (including its child nodes) with a randomly generated subtree in its place.

5.3.3. Abridgement

The abridgement operation can solve the redundancy and subsumption problems. The redundancy problem means that two or more rules with the same feature values and class labels appear in a classification tree. The abridgement operation will remove redundant rules and can thus reduce the tree complexity. Below, an example is given to illustrate how the abridgement operator deals with redundancy.

Example 2. Assume the classification tree CT_r in Figure 3 includes two redundant rules $Rule_{ri}$ and $Rule_{rj}$. To solve the redundancy problem, the abridgement operation is used on the tree CT_r, with the deeper redundant rule $Rule_{rj}$ removed. The resulting classification tree CT'_r is shown in the right part of Figure 3.

Rule subsumption means one rule subsumes the other in a classification tree. The subsumed rule may be removed from the tree to reduce tree complexity. Since redundancy is a special case of subsumption, the abridgement operation can be extended to solve the subsumption problem in a classification tree. Below, an example is given to illustrate how the abridgement operator deals with subsumption.

Example 3. Assume in Figure 4, Rule_{si} and Rule_{sj} are included in the classification tree CT_s. There is a subsumption relation between Rule_{si} and Rule_{sj}. To solve the subsumption problem, the abridgement operator is used on the tree CT_s. Rule_{sj} is thus removed from the tree since it has more feature values than Rule_{si}. The resulting classification tree CT'_s is shown in the right part of Figure 4.

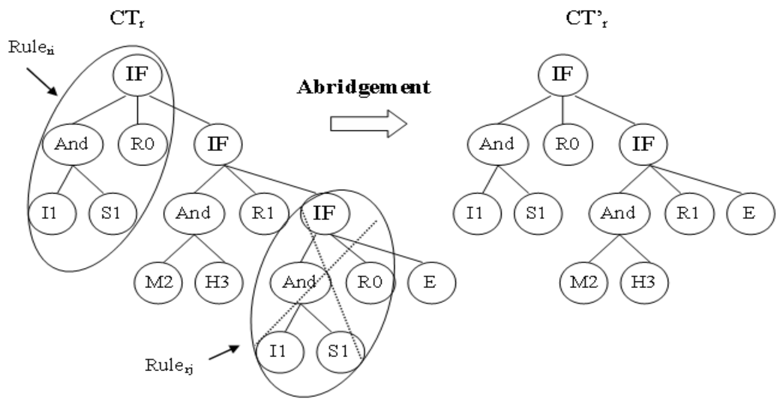

Figure 3. Using the abridgement operation for removing redundancy

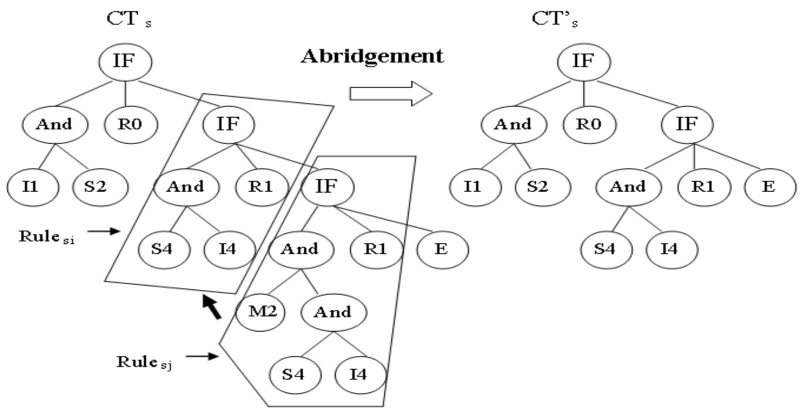

Figure 4. Using the abridgement operation for removing subsumption

5.3.4. Compromise

The compromise operation solves the contradiction problem, in which two rules with the same feature values conclude to different classes in a classification tree. It removes contradictive rules in a classification tree, such that the tree can become more practical and predictable. It can thus produce the consistent classification rules and raise the accuracy of a CT. Let the classification tree to be processed be CT_c with z rules. The steps for the compromise operation are shown as follows.

(1) Set $i = 1$ and $j = i+1$, where i and j are used to represent the indexes of the current two selected rules for checking.
(2) Compare Rule_{ci} with Rule_{cj} for their feature values and classes.
(3) If the two rules have the same feature values but different classes, split CT_c into two subtrees (CT_{c1}, CT_{c2}) to exclude the contradictive rule (Rule_{ci} or Rule_{cj}) respectively from CT_c, calculate their accuracy values and select the subtree with the higher value or select the first subtree by the equal value as the individual.
(4) Set $j = j+1$.
(5) If $j > z$, then do the next step; otherwise, go to step (2).
(6) Set $j = i+2$ and $i = i+1$.
(7) If $i = z$, then exit the searching process; otherwise, go to step (2).

Below, an example is given to illustrate how the compromise operator deals with contradiction.

Example 4. Assume the classification tree CT_c in Figure 5 includes two contradictive rules $Rule_{ci}$ and $Rule_{cj}$. To solve the contradictive problem, the compromise operator is used to split CT_c into two subtrees (CT_{c1}, CT_{c2}). Then it calculates the accuracy values of the two subtrees (assume CT_{c1}=83%, CT_{c2}=78%). CT_{c1} is then selected as the next individual since CT_{c1} has a higher value than CT_{c2}. The resulting classification tree CT_{c1} is shown at the right part of Figure 5.

5.4. The final classification tree

After the knowledge integration phase, the classification tree with the highest fitness value from the population is selected and transferred as a rule set.

Assume that a rule set RS_t has the following rules:

Rule 1: If (Annual income >800 thousand) and (Seniority = 4-8 years) and (Home ownership = owned) then Class is approval;
Rule 2: If (Annual income < 200 thousand) and (Seniority < 1 year) and (Marital status = single) then Class is rejection;
Rule 3: If (Annual income = 501-800 thousand) and (Seniority > 8 years) and (Marital status = married) then Class is approval.

…

These rules for a consumer loan can then be fed into a knowledge base that will support a manager to examine a consumer's condition and decide whether they can be approved or not in a bank.

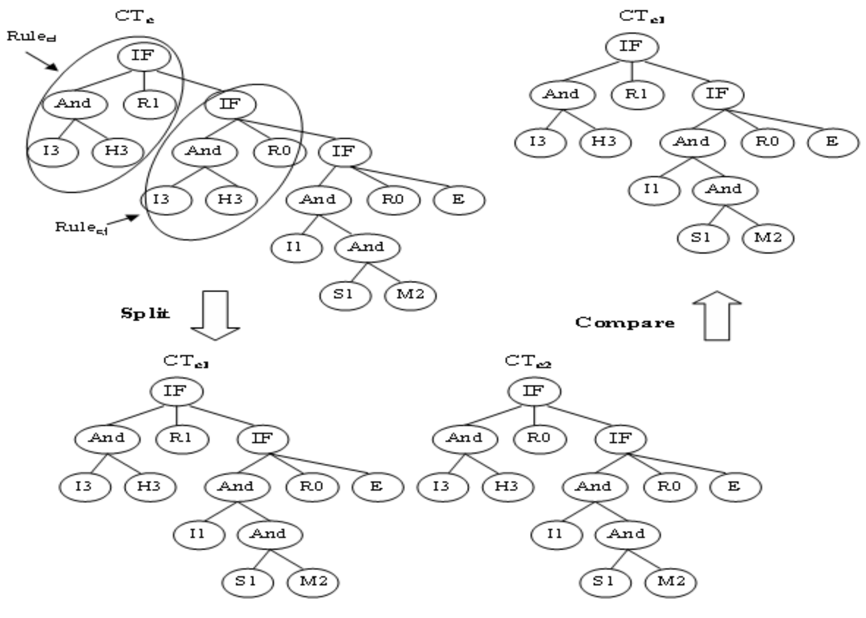

Figure 5. An example of compromise

6. Experimental results

In this section, experiments were made to show the performance of the proposed methods (GPKI), which were implemented in C++ on a personal computer. A set of data for breast cancer diagnosis was obtained for the experiments from a hospital in Taipei, Taiwan. A total number of 455 cases were collected in the data set. The cases consisted of thirteen features and two classes (Cancer and Noncancer). The goal was to find an integrated classification tree which could be converted into a set of rules to help identify one of the two classes.

The data set was first split into a training set and a test set. The training set was used to evaluate the fitness values of classification trees during the integration process and the test set was used as input cases to test the resulting classification trees. In each run, 70% of the breast cancer cases were randomly selected for training, and the remaining 30% were used for testing. The percentage of correct predictions was recorded. The initial population included ten rule sets, which were obtained from different groups of experts in the hospital or derived from data mining methods.

In the experiments for GPKI, the rates for crossover, mutation, abridgement and compromise operations were set at 0.9, 0.01, 0.8 and 0.8, respectively. The parameter μ in the fitness function was used to adjust a tradeoff between accuracy and complexity. If the μ value was small, the fitness function emphasized the accuracy; if the μ value was large, the fitness function focused on the complexity. The parameter μ was set at 0.125 in the experiments.

Experiments were made to compare GPKI with GPO method (without combining a mechanism for knowledge acquisition). Five different sets of samples were employed to test the accuracy rate. The results are shown in Table 1. The proposed approach obtained an accuracy rate of 82.04% for GPKI after 1000 generations, averaged over 5 runs. It can easily be seen that the proposed approach had a higher accuracy than GPO.

Table 1 The accuracy of the proposed method and GPO

Method	**Accuracy (%)**
Our approach	82.04
GPO	79.12

7. Conclusions

In this paper, we have proposed a knowledge-integration framework based on the technique of genetic programming to automatically integrate multiple knowledge sources into an integrated knowledge base. The proposed approach searches for an integrated classification tree according to the criteria of accuracy and complexity. It tries to find a good classification tree by genetic operators and improves its accuracy via the fitness function. Experimental results have also shown that the proposed approach could find an integrated classification tree with a higher accuracy than GPO. The final classification tree obtained can also be transferred back into a rule set, which can be further fed into a knowledge base to support decision-making processes in an organization.

The proposed approach has designed two new genetic operators, abridgement and compromise, to take domain-specific characteristics into consideration, thus reducing tree complexity, raising accuracy, and getting results closer to those desired. This, however, takes more execution time than using only the original operators. Our approach solves three commonly seen problems in rule-based systems. They are redundancy, subsumption and contradiction due to the integration. In the future, we may further deal with other knowledge verification problems. Besides, fuzzy knowledge is often seen and handled in the real world. We may thus also study how to deal with fuzzy knowledge integration through the proposed framework.

8. References

[1] W. Afzal, R. Torkar, “On the application of genetic programming for software engineering predictive modeling: a systematic review”, Expert Systems with Applications, vol. 38, pp.11984-11997, 2011.

[2] R. Agrawal, T. Imielinksi, A. Swami, “Mining association rules between sets of items in large database”, The 1993 ACM SIGMOD Conference, pp.207-216, 1993.

[3] M. Alavi, D. E. Leidner, “Review: knowledge management and knowledge management systems: conceptual foundations and research issues”, MIS Quarterly, vol. 25, no. 1, pp.107-136, 2001.

[4] L. K. J. Baartman, E. D. Bruijn, “Integrating knowledge, skills and attitudes: conceptualising learning processes towards vocational competence”, Educational Research Review, vol. 6, pp.125-134, 2011.

[5] S. H. Chen, T. W. Kuo, "Evolutionary computation in economics and finance: a bibliography," Evolutionary Computation in Economics and Finance, Physica-Verlag, USA, pp.419-455, 2002.
[6] R. O. Duda, P. E. Hart, D. G. Stork, Pattern Classification, Wiley Interscience, USA, 2001.
[7] B. R. Gaines, M. L. Shaw, "Eliciting knowledge and transferring it effectively to a knowledge-based system", IEEE Transactions on Knowledge and Data Engineering, vol. 5, no. 1, pp.4–14, 1993.
[8] J. Giarratano, G. Riley, Expert System Principles and Programming, MA: PWS, USA, 1993.
[9] L. Gusel, M. Brezocnik, "Application of genetic programming for modelling of material characteristics" , Expert Systems with Applications, vol. 38, pp.15014-15019, 2011.
[10] N. Huang, S. Diao, "Ontology-based enterprise knowledge integration", Robotics and Computer-Integrated Manufacturing, vol. 24, pp.562-571, 2008.
[11] J. C. Huangc, S. Newell, "Knowledge integration processes and dynamics within the context of cross-functional projects", International Journal of Project Management, vol. 21, pp.167–176, 2003.
[12] G. J. Hwang, S. S. Tseng, "EMCUD: A knowledge acquisition method which captures embedded meanings under uncertainty", International Journal of Man-Machine Studies, vol. 33, pp.431-451, 1990.
[13] G. A. Kelly, The Psychology of Personal Constructs, Norton, USA, 1955.
[14] J. R. Koza, Genetic Programming: On the Programming of Computers by Means of Natural Selection, MIT Press, USA, 1992.
[15] J. R. Koza, Genetic Programming III: Darwinian Invention and Problem Solving, Morgan Kaufmann, USA, 1999.
[16] C. S. Kuo, T. P. Hong, C. L. Chen, "Applying Genetic Programming Technique in Classification Trees", Soft Computing, vol. 11, no. 12, pp. 1165-1172, 2007.
[17] M. M. Kwan, P. Balasubramanian, "Knowledgescope: managing knowledge in context", Decision Support Systems, vol. 35, pp.467-486, 2003.
[18] L. Medsker, M. Tan, E. Turban, "Knowledge acquisition from multiple experts: problems and issues", Expert Systems with Applications, vol. 9, pp.35-40, 1995.
[19] C. Neely, P. Weller, R. Dittmar, "Is technical analysis in foreign exchange market profitable? A genetic programming approach", Journal of Financial and Quantitative Analysis, vol. 32, no. 4, pp.405-426, 1997.
[20] D. Pan, "An Integrative Framework for Continuous Knowledge Discovery", Journal of Convergence Information Technology, vol. 5, no. 3, pp.46-53, 2010.
[21] J. R. Quinlan, "Induction of decision trees", Machine Learning, vol. 1, pp.81– 106, 1986.
[22] J. R. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufmann, USA, 1993.
[23] L. Sun, J. Xu, Y. Song, "Information Quantity-based Decision Rule Acquisition from Decision Tables", Journal of Convergence Information Technology, vol. 7, no. 2, pp.57-67, 2012.
[24] H. Wang, "Intelligent agent-assisted decision support systems: integration of knowledge discovery, knowledge analysis, and group decision support", Expert Systems with Applications, vol. 12, no. 3, pp.323-335, 1997.
[25] C. H. Wang, T. P. Hong, S. S. Tseng, "Self-integrating knowledge-based brain tumor diagnostic system", Expert Systems with Applications, vol. 11, no. 3, pp.351-360, 1996.
[26] C. H. Wang, T. P. Hong, S. S. Tseng, C. M. Liao, "Automatically integrating multiple rule sets in a distributed-knowledge environment", IEEE Transactions on Systems, Man, and Cybernetics, vol. 28, no. 3, pp.471-476, 1998.
[27] Y. Xu, A. Bernard, "Quantifying the value of knowledge within the context of product development", Knowledge-Based Systems, vol. 24, pp.166-175, 2011.