A Two-stage Rule Base Optimization Method Based on Combination of Classical Algorithm and GA for Intelligent Decision Support System

Hongqin Wei

Glorious Sun Business & Management School, Donghua University, Shanghai 200051, P. R. China

Abstract

Intelligent decision support system performance depends on knowledge base quality. With the decision problem and knowledge base becoming more complicated, it is necessary to provide effective method to optimize the management of knowledge base. Based on detailed analyzing of running characteristics and potential detects of rule base in IDSS, a novel two-stage optimization method is proposed. Conventional optimization approach and genetic algorithm are combined to recognize and eliminate kinds of defects in rule base. Exact defect definitions and optimization operation details are given. The sample shows that the method is feasible in practice.

Keywords: Rule base optimization, Rule base maintenance, Intelligent decision support system, Genetic algorithm, Generative knowledge system

1. Introduction

By adding knowledge base to traditional DSS, intelligent decision support system provides an effective solution for unstructured and semi-structured decision problems, which can't be supported well by common DSS for large volumes, bad structures or complexity earlier[1]-[4]. All the activities of intelligent decision-making are supported by knowledge system. The knowledge quality and reasoning efficiency of knowledge base decide the system performance and decision quality. In the early period, the knowledge base is small and simple, and the maintenance can be implemented manually. With system and problem is becoming more and more complicated, there are large quantity rules in the base with complicated relationship between rules, and manual method can't give satisfying maintenance results. Many researches on rule base optimization have been done in the past decade[5]-[9], but most of them is about one or few knowledge defects and can't be implemented automatically. In the paper, an automatic optimization method is given based on conventional and genetic algorithm for rule base, which can be realized easily using common OO tools with little experts' help.

2. GA based optimization

The combinatorial optimization problems of knowledge engineering field are mostly nonlinear, which can't be solved effectively by conventional optimization technologies. Genetic algorithm is a kind of adaptive heuristic global search algorithm based on probability iterative, and has many superior properties such as robustness, global optimality, not depending on problem models, ie. GA has been used widely in many fields[10]-[14].

The general structure of GA can be described as Fig.1. GA begins the search process with one group of initial solutions called "Population". An individual in population is one solution of problem, which is called as chromosome. Chromosome is a series of symbols, such as binary string. The chromosomes will be evolved continually during iterations between generations, which is called as genetic process. In every generation, fitness is used to evaluate the chromosome quality and to create next generation chromosome, that is, offspring, by crossover or mutation operation. Chromosomes with higher fitness will be selected. After times of iteration, operation will converge to best chromosomes, which is mostly the optimal or suboptimal solution of problem [15].

In the paper, the genetic algorithm will be amended partly to meet the special needs of rule base optimization according to the running characteristics of knowledge system. To overcome shortness of GA, some conventional methods are used together.

3. Rule base elements analysis and representation

To facilitate the automatic defects recognition and optimization in system, it is necessary to extract all the component elements of rule base and give each of them a represent symbol, which can make rules and other basic components to be located and recognized accurately. Decision rule base is a set of domain knowledge for special decision problem. In addition to auxiliary attributes, the core components of each rule are assumption and conclusion. Assumption and

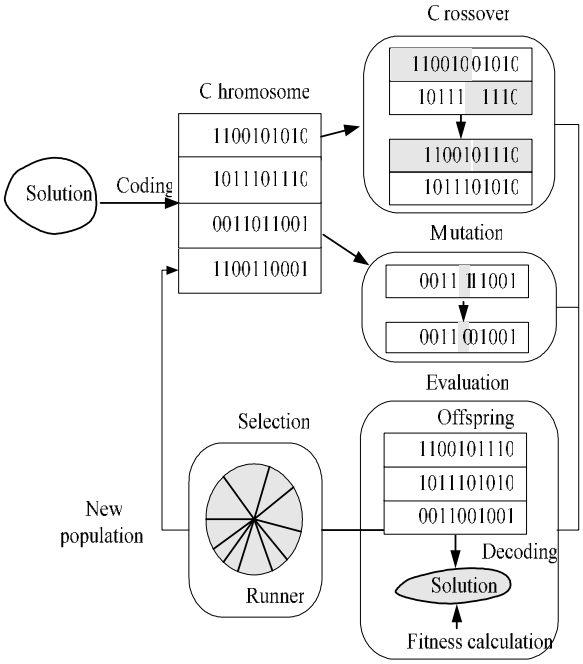

Fig 1: Genetic algorithm general structure

conclusion can contain one or some sub-clauses respectively. Every decision domain can be abstracted as a set of objects, and every object have their attributes set, in which every attribute have several feasible values and form one value field. Every sub-clause can be described as "the value of attribute A of object O is V". Rule base elements can be represented as follows.

1. rule set can be represented as $R=\{R_1, R_2,\ldots,R_i,\ldots\}$, in which R_i is the *i*th rule in rule set;
2. assumption set $A=\{A_1,A_2,\ldots,A_i, \ldots\}$, in which A_i is the assumption component of R_i;
3. Conclusion set $C= \{C_1,C_2,\ldots, C_i, \ldots\}$, in which C_i is the conclusion component of R_i;
4. sub-clause set $M=\{M_1, M_2, \ldots, M_i, \ldots\}$, in which M_i is the *i*th sub-clause in all assumption and conclusion;
5. reasoning element set $B=\{B_1,B_2,\ldots,B_i, \ldots\}$, in which B is the combination result set of A and C without redundancy;
6. domain knowledge object set $O=\{O_1, O_2,\ldots,O_i,\ldots\}$, in which is the *i*th object in the decision-making domain
7. object attribute set $P=\{P_{11},P_{12},\ldots,P_{21}, P_{22}, \ldots, P_{ij}, \ldots\}$, in which P_{ij} is the jth attribute of the *i*th object;
8. value range of object attribute $S=\{ S_{111}, S_{112},\ldots, S_{121}, \ldots, S_{ijk,\ldots}\}$, in which S_{ijk} is the *k*th feasible value of P_{ij}

Optimization algorithm can be realized by object-oriented language[16]. For one rule base, all the above elements can be saved in relational database, the smallest Storage Granularity is the attributes of object and their values. For every rule, the detailed description of the sub-clauses composing its assumption and conclusion can be gain from the database.

4. Two-stage rule base optimization

4.1. Initial optimization of rule base

Genetic algorithm doesn't perform well always for all rule base defects recognition. For example, GA doesn't detect conflict rules more quickly than conventional methods. And when there exist circulation rules in rule base, the optimization program will fall into a death cycle. On the basis of detail analysis of defect characteristics, a two-stage optimization solution is adopted to realize the defect recognition and performance optimization in the paper. In the first stage, conventional method is used to eliminate six kinds of basic defects, namely, equivalent rules, conflict rules, contained rules, omitted rules, circulation rules and unreachable objectives.

Some works need to be done before initial optimization. The first step is to get sub-clause set of rule base, that is, $M=\{M_1, M_2, \ldots, M_i, \ldots\}$, and the M_i can be described as $P_{jk}=S_{jkl}$. Different combination of sub-clauses forms the assumption and conclusion parts of rules. One rule R can be described as $(M_i \cap M_j \cap \ldots) \rightarrow (M_l \cap M_n \cap \ldots)$. Based on sub-clause M analysis, the next step is to build up assumption set A and conclusion set C. Finally reasoning elements set B is gotten by merging A and C.

4.1.1 Equivalent rules detection and elimination

Definition 1 If two rules have same assumptions and conclusions, then they are equivalent rules. That is, $M_i \cap M_j \rightarrow C_k$ and $M_j \cap M_i \rightarrow C_k$ are equivalent rules.

For Equivalent rules, the optimization algorithm is as follows.

1) get every sub-clause in M and their serial number n;
2) For every rule, rearrange the sub-clause components in assumption and conclusion to make their sequence is in accordance to the sub-clause number;
3) recognize rules with same A and C as equivalent rules
4) Remain one of the equivalent rule set and delete

others.

4.1.2 Conflict rules detection and elimination

Definition 2 If two rules have same assumption and conflict conclusion, then they are conflict rules, ie. $A_i \rightarrow C_j$ and $A_i \rightarrow \neg C_j$ are conflict with each other.

For conflict rules, the optimization algorithm is as follows.

1) recognize rules with same assumption, that is , $A_i=A_j=B_k$, and get one rule subset to be detected;
2) check the conclusion sub-clauses of rules in each subset to find in two rules whether one object attribute is given opposite values respectively. If it is , then the corresponding rules are conflict rules;
3) Mark conflict rules and provide them to experts, then keep the correct one;

4.1.3 Contained rules detection and elimination

Definition 3 If two rule R_i and R_j have same conclusions, but R_j have additional constrains in assumption component, then R_i is the contained rule of R_j, that is , rule $M_i \rightarrow C_k$ is contained rule of rule $M_i \cap M_j \rightarrow C_k$.

For contained rules, the optimization algorithm is as follows.

1) recognize rule subsets with same conclusion;
2) check the assumption component of rules in subsets, abstract their sub-clause sets, if one sub-clause set can be contained by another, there are contains rules.
3) provide them to expert and check which one is correct. If there is no wrong rule, then the contained rule is kept and others are discards.

4.1.4 Circulation rules detection and elimination

Definition 4 If the rules in one set can form a close chain, then the rule set is Circulation rule set, rules in the set are circulation rules, which can be marked as $A_i \rightarrow C_j, C_j \rightarrow C_k, C_k \rightarrow C_l, C_l \rightarrow A_i$. For a knowledge based system, reasoning will fall into a death cycle when circulation rule chain exists.

For Circulation rules, the optimization algorithm is as follows.

1) design complete test case set for the decision-making problem;
2) test rule set using the test case one by one by forward reasoning; mark every reasoning path by $B_i \rightarrow B_j \rightarrow B_m \rightarrow \ldots$, and get the reasoning element number in every step;
3) judge whether the current reasoning element has occurred in the reasoning path, If it is, stop the reasoning.
4) mark the path, and provide to expert to amend.

4.1.5 Omitted rule detection and elimination

Definition 5 For a rule set, if one or some feasible values of an object attribute can't be covered by any rule assumption, then there is omitted rule in the rule set.

The existing of partly covered attribute will prevent one conclusion to be reached or make system running successfully.

The optimization algorithm for omitted rule is as follows.

1) carry out every tasks in test case set using rules in the rule base by forward reasoning;
2) If there is empty reasoning result, then there exist omitted rules in knowledge base. Mark the decision problem.
3) Under the guidance of expert, insert new rules and continue to test until all tasks are solved.

4.1.6 Unreachable objectives detection and elimination

Definition 6 In generative system driven by objective, it is need that there is a conclusion matching with one of objective in the rule set. If not, the objective is called unreachable objective. That is, for objective set $T=\{T_1, T_2, \ldots, T_i, \ldots\}$, if there is no $C_j=T_i$ in the conclusion set C, then T_i is unreachable objective.

The optimization algorithm for defect of unreachable objective is as follows.

1) carry out backward reasoning for every objective in objective set;
2) If the rule with conclusion matching with the objective can't be found, one unreachable objective exist;
3) Add new rules to rule base, redo 1) until all objectives can be reached.

4.2. GA based advanced optimization of rule base

After initial optimization, most defects in rule base have been eliminated. The correctness is improved. But the running efficiency of system isn't necessary high because there are still many redundant knowledge in the knowledge base. For example, there maybe exist two reasoning path with same initial assumption and finial conclusion. In the longer path, apart from two beginning and ending rules the others are called redundant rules. Redundant rules make rule base bigger and difficult to be managed. In the same time, reasoning efficiency will be slow. In addition to

redundant rules, there maybe exist void rules in rule base. These rules can't be used for ever during decision-making. The above two problems in initially optimized rule base can be solved further by Genetic Algorithm.

During realization of GA, the first task is to design coding method. The usual code system is binary code. But the rule quantity of rule base is very large and some rules are very complex. In the assumption and conclusion part, there are several sub-clauses. The code length will become very large using binary coding method. In the paper, rules are selected as chromosome and the sub-clauses composing the assumption and conclusion are genes. A rule base forms one original population to be optimized.

During implementing the advanced optimization for knowledge base by GA, the steps are as follows.

(1) copy rules to two-level optimization rule base from initial optimization rule base, and make the initial fitness values of each rule equals 0;

(2) Design complete test case set for decision-making task. Every test case includes known decision-making constrains and correct reasoning results. With the assumption of test case selected as known realities, carry out forward reasoning using rules in two-level optimization rule base by reasoning machine and calculate the fitness values;

(3) For objective-driven reasoning system, carry out backward test reasoning using the objectives in objective set O, and calculate the fitness values;

Fitness of rules is measured by reasoning path length, utilization probability, Solvability of problem, reasoning correctness, the calculating principles are as follows.

- For several reasoning paths having complete one task successfully, fitness value of rules in the longest path will be plus 1, while rules in other paths will be plus 2 or 3….. The shorter the path, more increases the fitness value of rules in the path will be given;
- For one wrong reasoning results, fitness values of the rules in the path will minus 1;
- rule fitness will increase when rule is activated and could complete reasoning task well, while fitness of rules not activated will unchanged;
- Fitness of rules which can activate other rules after having been activated, will increase more;

Create next generation rule set based on the fitness value of rules using following genetic operations.

(a) selection: select rules with higher fitness value and delete ones with lower fitness, the quantity of deleted rules should make sure that all tasks can be reasoned by the rest rules;

(b) copy: copy the selected rules to new generation optimization rule set;

(c) Mutation: amend the assumption or conclusions of wrong rules and insert them to the optimization rule base; or insert new rules according to expert's guidance. Having eliminate reasoning mistakes, lest changes should be make to avoid new knowledge defects;

(4) Redo (2)and (3) until the fitness of rules in the rule base don't change any more, then the best optimization rule base is gotten.

By genetic algorithm useless rules will be recognized for their fitness will remain 0. And redundant rules will be eliminated completely during the replacement of rules base generations gradually.

The complete flowchart of decision rule base optimization is shown as Fig.2.

The rules in optimized rule base are saved in natural and random condition. To shorten knowledge searching time and accelerate the reasoning speed, rules used frequently should be find more easily and rules close to each other in logic should be put together. So the next work after optimization is to reorganize the knowledge base structure. The best reasoning path for every task should be find and collect all the rules together.

Case study

In the paper, the problem of law firm task assignment is selected to test the validity of the proposed optimization method. There are 30 rules in original knowledge base. After initial optimization, 20 rules is remained and copied to two-level optimization rule base, that is, $R=\{R_1, R_2, \ldots, R_{20}\}$. Set the fitness values of all rules equal 0, and run the test tasks to evaluate the rules.

Based on the fitness values, get new generation rules by selection, copy and mutation operations with expert's support. The optimization results are shown as Table1. The fitness values move forward stability after 16 generations iteration. The final optimized rule set is $R'=\{R_1, R_2, R_6, R_7, R_{11}, R_{12}, R_{14}, R_{15}, R_{17}, R_{18}, R_{19}, R_{21}, R_{23}, R_{24}\}$, in which R_{21}, R_{23} and R_{24} are new rules inserted by expert.

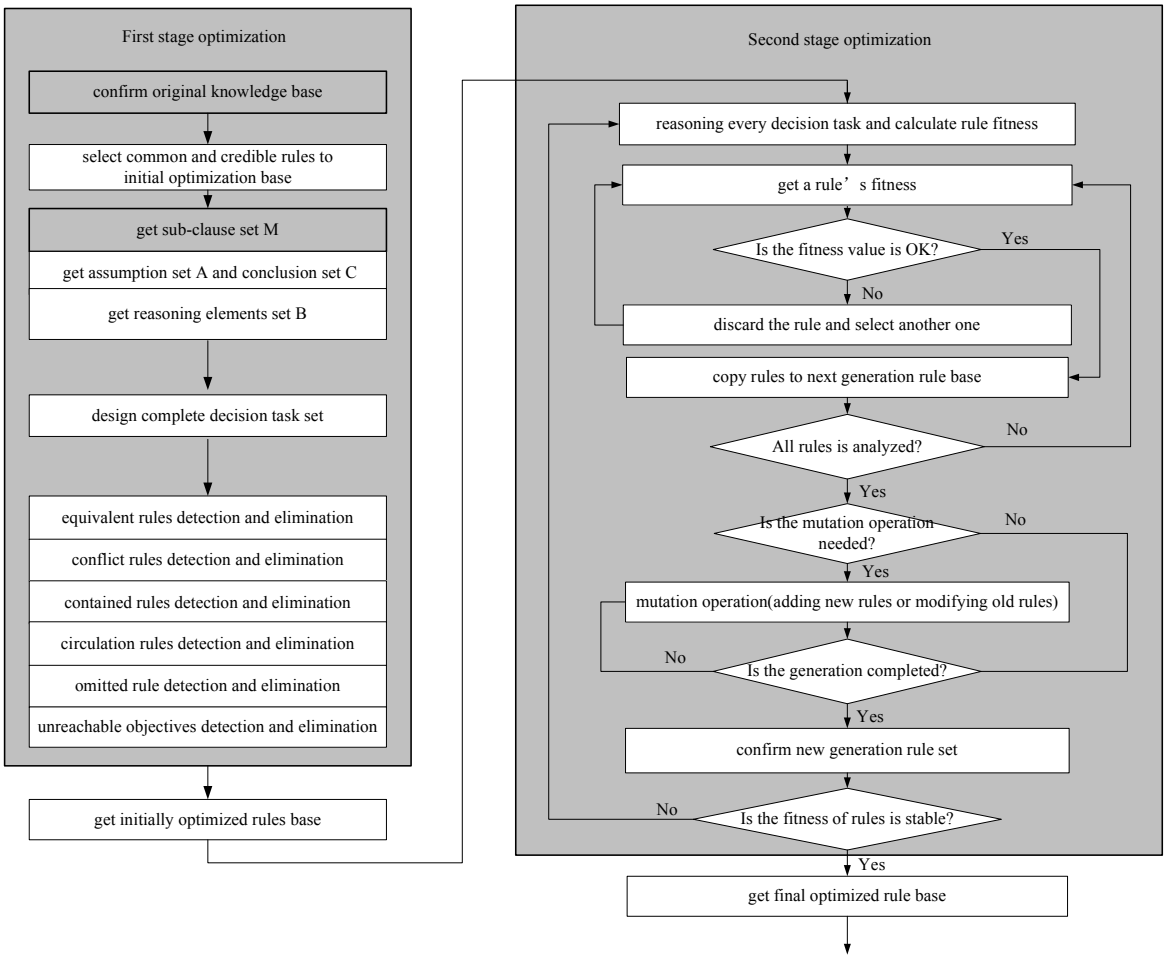

Fig.2 GA-based two-stage rule base optimization flowchart

Generations	Rule set and fitness values																				
	Rule set	R_1	R_2	R_3	R_4	R_5	R_6	R_7	R_8	R_9	R_{10}	R_{11}	R_{12}	R_{13}	R_{14}	R_{15}	R_{16}	R_{17}	R_{18}	R_{19}	R_{20}
1	Rule	7	5	3	-1	3	4	2	1	2	0	4	6	1	5	5	1	6	6	7	2
	Rule	R_1	R_2	R_3	R_5	R_6	R_7	R_8	R_9	R_{11}	R_{12}	R_{13}	R_{14}	R_{15}	R_{16}	R_{17}	R_{18}	R_{19}	R_{20}	R_{21}	
2	Rule	7	5	0	4	4	2	2	3	4	6	2	5	5	2	6	6	8	1	6	
	Rule	R_1	R_2	R_5	R_6	R_7	R_8	R_9	R_{11}	R_{12}	R_{13}	R_{14}	R_{15}	R_{16}	R_{17}	R_{18}	R_{19}	R_{21}	R_{22}	R_{23}	R_{24}
3	Rule	6	5	4	4	3	1	2	4	5	2	5	5	1	6	5	8	6	6	6	4
…	…																				
	Rule	R_1	R_2	R_6	R_7	R_{11}	R_{12}	R_{14}	R_{15}	R_{17}	R_{18}	R_{19}	R_{21}	R_{23}	R_{24}						
15	Rule	6	5	5	4	6	5	5	5	6	5	7	4	6	5						
	Rule	R_1	R_2	R_6	R_7	R_{11}	R_{12}	R_{14}	R_{15}	R_{17}	R_{18}	R_{19}	R_{21}	R_{23}	R_{24}						
16	Rule	6	5	5	4	6	5	5	5	6	5	7	4	6	5						

Table1 1: GA optimization results of law firm task assignment rule base.

5. Conclusions

The performance of intelligent decision support system and decision-making quality depend on knowledge base content and running condition. With the system and decision problem being more complex, the knowledge base is becoming bigger and more complex. There are maybe kinds of potential defects and will impact the

system effectiveness and efficiency seriously. The maintenance and optimization works are necessary during the process of knowledge base establishing, renewing and using. Conventional methods and nonclassical algorithm have their own advantages in different applicants. In the paper, based on analysis of decision-making rule base characteristics, a two-stage optimization method is proposed. The method combines conventional algorithm and genetic algorithm, and amends the genetic algorithm to fit the need of knowledge base reasoning.

Acknowledgement

This work is partially supported by National Nature Science Foundation of China (Grant No. 70271002).

References

[1] W. Wen, W. K. Wang and T. H. Wang, A hybrid knowledge-based decision support system for enterprise mergers and acquisitions, *Expert Systems with Applications*, 28: 569-582, 2005.

[2] María N. Moreno García, Luis A. Miguel Quintales, Francisco J. García Peñalvo and M. José Polo Martín, Building knowledge discovery-driven models for decision support in project management, *Decision Support Systems*, 38:305-317, 2004.

[3] Solomon Antony, Dinesh Batra and Radhika Santhanaml, The use of a knowledge-based system in conceptual data modeling, *Decision Support Systems*, 41: 176-188, 2005.

[4] Liang-Tsung Huang, M. Michael Gromiha, Shiow-Fen Hwang and Shinn-Ying Ho. Knowledge acquisition and development of accurate rules for predicting protein stability changes, *Computational Biology and Chemistry*, 30: 408-415, 2006.

[5] S. Liu and M. Sun, Research on knowledge base maintenance technology, *Harbin Polytechnic University Journal*, 2: 33-36, 1997 (in Chinese).

[6] Y. Sun and R. Bie, Research on generative rule base refinement, *Beijing Normal University Journal : Natural Science* ,39: 435-443, 2003(in Chinese).

[7] Sang C. Park, Selwyn Piramuthu and Michael J. Shaw, Dynamic rule refinement in knowledge-based data mining systems, *Decision Support Systems*, 31: 205-222, 2001.

[8] W. Mark, D. Sleeman and P. Tim, Inventory management using constraint satisfaction and knowledge refinement techniques, *Knowledge-Based Systems*, 11: 293-300, 1998.

[9] Christopher W. Zobel, Loren Paul Rees and Terry R. Rakes, Automated merging of conficting knowledge bases, using a consistent, majority-rule approach with knowledge-form maintenance, *Computers & Operations Research*, 32:1809–1829, 2005.

[10] S. S. Ana and Pedro M. Vilarinho, A genetic algorithm based approach to the mixed-model assembly line balancing problem of type II, *Computers & Industrial Engineering*, 47: 391-407, 2004.

[11] F. Roberto, G. Giorgio and Fulvia B. Quagliotti, Flight control system design and optimisation with a genetic algorithm, *Aerospace Science and Technology*, 9: 73-80, 2005.

[12] Ramón Quiza Sardiñas, Pedro Reis and J. Paulo Davim, Multi-objective optimization of cutting parameters for drilling laminate composite materials by using genetic algorithms, *Composites Science and Technology*, 66: 3083-3088 2006.

[13] Hamed Hasheminia and Seyed Taghi Akhavan Niaki, A genetic algorithm approach to find the best regression/econometric model among the candidates, *Applied Mathematics and Computation*, 183: 337-349 , 2006.

[14] J. Miller, W. Potter, R. Gandham and C. Lapena, An evaluation of local improvement operators for genetic algorithms, *IEEE Transaction on Systems , Man, and Cybernetics*, 1993, 23: 1340-1351.

[15] G. Xuan and R. Cheng, *Genetic algorithm and engineering design*, Science press, 2000(in Chinese).

[16] Yao Tsung Lin, S. S. Tseng and Chi-Feng Tsai, Design and implementation of new object-oriented rule base management system, *Expert Systems with Applications*, 25: 369-385, 2003.