Chapter 10

Rule-set-based Bi-level Decision Making

As discussed in previous chapters, bi-level decision-making problems are normally modeled by bi-level programming. Because many uncertain factors are involved in a bi-level decision-making, it is sometimes very difficult to formulate the objective functions and constraints of decision makers. When a bi-level decision problem cannot be formulated by normal bi-level programming, we can consider using rules to express the objective functions and constraints of a decision problem.

This chapter presents a *rule-set-based bi-level decision* (RSBLD)concept, approach and solution. It first introduces related theories to prove the feasibility of modeling a bi-level decision problem by a rule-set. It then proposes a *rule-set-based bi-level decision modeling approach* (i.e. Algorithm 10.1) for modeling a bi-level decision problem which contains rule generation, rule reduction and other steps. For solving a rule-set-based bi-level decision problem, this chapter also presents a *rule-set-based bi-level decision solution algorithm* (i.e., Algorithm 10.2) and a *transformation-based solution algorithm* (i.e., Algorithm 10.3) through developing *attribute-importance-degree* (AID) based rule trees (Zheng and Wang 2004). A case-based example is used to illustrate the functions and effectiveness of the proposed RSBLD modeling approach and the solution algorithms.

This chapter is organized as follows. Section 10.1 identifies the non-programming bi-level decision problem. Section 10.2 introduces the concepts and notions of information tables and rule-set, which are given as preliminaries in this chapter. Section 10.3 presents a RSBLD model. A RSBLD modeling approach is presented in Section 10.4. In Section 10.5, two RSBLD solution algorithms are provided to solve a bi-level decision problem which is modeled by rule-sets. A case-based example is shown in Section 10.6. Section 10.7 gives experiment results, and the summary is presented in Section 10.8.

10.1 Problem Identification

In general, there are two main uncertainties in modeling a bi-level decision problem. One is that the parameter values in the objective functions and constraints of the leader and the followers may be indefinite or inaccurate. Fuzzy optimization approaches can handle this issue, as discussed in previous chapters. Another type of uncertainty involves the form of the objective functions and constraints. That is, how to determine the relationships among the proposed decision variables and formulate the functions for a real decision problem. The challenge can be handled by a rule-set-based approach. Below, we give an example by way of explanation.

Example 10.1 A factory's human resource management system is distributed over two levels. The upper level is the factory executive committee and the lower level is the workshop management committee. When deciding whether a person can be recruited for a particular position, the factory executive committee principally considers the following two factors; the "age" and "education level (edulevel)" of the person. The workshop management committee is largely concerned with two other factors: the "seniority" and "health" of the person. Suppose the condition attributes in ascending order according to the importance degree are "age", "edulevel", "seniority", "health".

It is clearly difficult to express the worker selection conditions of the two committees as linear or non-linear functions, but the two committees have all necessary information about the workers. We can build rules using the information to form the selection conditions and objectives of this decision (selection) problem.

10.2 Information Tables and Rule-sets

For the convenience of describing proposed models and algorithms, we will first introduce some basic notions of information tables, formulas, rules, decision rule set functions and rule trees. In addition, we will give some related definitions and theorems which will be used in subsequent sections.

10.2.1 Information Tables

To present the definition of a rule, we first describe information table and decision table.

An information table is a knowledge-expressing system which is an important tool for representing and processing knowledge in machine learning,

data mining and many other fields. It provides a convenient way to describe a finite set of objects called the universe of discourse by a finite set of attributes (Pawlak 1991).

Definition 10.1 (Information table, Pawlak 1991**)** An information table can be formulated as a tuple:

$$S = (U, At, L, \{V_a \mid a \in At\}, \{I_a \mid a \in At\}),$$

where U is a finite non-empty set of objects, At is a finite non-empty set of attributes, L is a language defined using attributes in At, V_a is a non-empty set of values for $a \in At$, $I_a: U \to V_a$ is an information function. Each information function I_a is a total function that maps an object of U to exactly one value in V_a.

An information table can represent all the available information and knowledge about a situation. Here the objects are only perceived, observed, or measured by using a finite number of properties. We can easily extend the information function I_a to some subsets of attributes. For a subset $A \subseteq At$, the value of an object x over A is denoted by $I_A(x)$.

A decision table is a special case of information tables. A decision table is commonly viewed as a functional description, which maps inputs (conditions) to outputs (actions) without necessarily specifying the manner in which the mapping is to be implemented (Lew and Tamanaha 1976). The formal definition of a decision table is given as follows.

Definition 10.2 (Decision table, Pawlak 1991**)** A decision table is an information table for which the attributes in A are further classified into disjoint sets of condition attributes C and decision attributes D, i.e., $At = C \cup D$, $C \cap D = \emptyset$.

A decision table can be seen as a special and important knowledge expression system. It shows that, when some conditions are satisfied, decisions (actions, operations, or controls) can be made. Decision attributes in a decision table can be unique or not. In the latter case, the decision table can be converted to one with a unique decision attribute. Therefore, in this chapter, we suppose that there is only one decision attribute in a decision table.

Based on the two definitions above, we introduce the following definitions.

10.2.2 Formulas and Rules

Definition 10.3 (Formulas, Yao and Yao 2002**)** In the language L of an information table, an atomic formula is given by $a = v$, where $a \in At$ and $v \in V_a$. If ϕ and ψ are formulas, then so as $\neg\phi, \phi \wedge \psi$, and $\phi \vee \psi$.

The semantics of the language L can be defined in the Tarski style through the notions of a model and satisfiability. The model is an information table S, which provides interpretation for the symbols and formulas of L.

Table 10.1 An information table

Object	height	hair	eyes	Class
o_1	short	blond	blue	+
o_2	short	blond	brown	-
o_3	tall	dark	blue	+
o_4	tall	dark	blue	-
o_5	tall	dark	blue	-
o_6	tall	blond	blue	+
o_7	tall	dark	brown	-
o_8	short	blond	brown	-

Definition 10.4 (Satisfiability of formula, Yao and Yao 2002**)** The satisfiability of a formula ϕ by an object x, written $x \vDash_S \phi$ or in short $x \vDash \phi$ if S is defined by the following conditions:

(1) $x \vDash a = v$ if $I_a(x) = v$,
(2) $x \vDash \neg\phi$ if $x \nvDash \phi$,
(3) $x \vDash \phi \wedge \psi$ if $x \vDash \phi$ and $x \vDash \psi$.
(4) $x \vDash \phi \vee \psi$ if $x \vDash \phi$ or $x \vDash \psi$.

If ϕ is a formula, the set

$$m_s(\phi) = \{x \in U | \, x \vDash \phi\}$$

is called the meaning of the formula ϕ in S. If S is understood, we simply write $m(\phi)$. The meaning of a formula ϕ is therefore the set of all objects having the property expressed by the formula ϕ. In other words, ϕ can be viewed as the description of the set of objects $m(\phi)$. Thus, a connection between the formulas of L and subsets of U is established.

To illustrate the idea, consider an information table given by Table 10.1 (Quinlan 1983). The following expressions are some of the formulas of the language L:

$$\textbf{height} = \text{tall}, \textbf{hair} = \text{dark},$$
$$\textbf{height} = \text{tall} \wedge \textbf{hair} = \text{dark},$$
$$\textbf{height} = \text{tall} \vee \textbf{hair} = \text{dark}.$$

The meanings of the formulas are given by:
$m(\mathbf{height} = \text{tall}) = \{o_3, o_4, o_5, o_6, o_7\}$,
$m(\mathbf{hair} = \text{dark}) = \{o_4, o_5, o_7\}$,
$m(\mathbf{height} = \text{tall} \wedge \mathbf{hair} = \text{dark}) = \{o_4, o_5, o_7\}$,
$m(\mathbf{height} = \text{tall} \vee \mathbf{hair} = \text{dark}) = \{o_3, o_4, o_5, o_6, o_7\}$.

From Definition 10.1, we know that an information table records the attribute values of a set of objects and can be an object database. The aim of knowledge acquisition is to discover useful and regular knowledge from the object databases. Usually, the knowledge is expressed by rules which can be formulated as follows (Pawlak 1991; Yao and Yao 2002).

Definition 10.5 (Rules) Let $S = (U, At, L, \{V_a | a \in At\}, \{I_a | a \in At\})$ be an information table, then a rule r is a formula with the form

$$\phi \Rightarrow \psi,$$

where ϕ and ψ are formulas of information table S for any $x \in U$,

$$x \vDash \phi \Rightarrow \psi \text{ iff } x \vDash \neg\phi \vee \psi.$$

Definition 10.6 (Decision Rules) Let $S = (U, C \cup D, L, \{V_a | a \in At\}, \{I_a \mid a \in C \cup D\})$ be a decision table, where C is the set of condition attributes and D is the set of decision attributes. A decision rule dr is a rule with the form $\phi \Rightarrow \psi$, where ϕ, ψ are both conjunctions of atomic formulas, for any atomic formula $c = v$ in ϕ, $c \in C$, and for any atomic formula $d = v$ in ψ, $d \in D$.

It is evident that each object in a decision table can be expressed by a decision rule.

Definition 10.7 An object x is said to be consistent with a decision rule $dr: \phi \Rightarrow \psi$, iff $x \vDash \phi$ and $x \vDash \psi$; x is said to be conflict with dr, iff $x \vDash \phi$ and $x \vDash \neg\phi$.

Based on these definitions, we introduce decision rule set functions.

10.2.3 Decision Rule Set Function

To present a bi-level decision model based on obtaining a decision rule set, we first need to define decision rule set functions. A decision rule set function can be defined as a mapping from n objects to a decision. Given a decision table $S = (U, At, L, \{V_a | a \in At\}, \{I_a | a \in At\})$, where $At = C \cup D$ and $D = \{d\}$. Suppose x and y are two variables, where $x \in X$ and $X = V_{a1} \times \cdots \times V_{am}$,

$y \in Y$ and $Y = V_d$. V_{ai} is the set of attribute a_i's values, $a_i \in C$, $i = 1$ to m, m is the number of condition attributes. RS is a decision rule set generated from S.

Definition 10.8 (Decision rule set function) A decision rule set function rs from X to Y is a subset of the Cartesian product $X \times Y$, such that for each x in X, there is a unique y in Y generated with RS such that the ordered pair (x, y) is in rs, where RS is called a decision rule set, x is called a condition variable, y is called a decision variable, X is the definitional domain, and Y is the value domain.

Calculating the value of a decision rule set function is to make decisions for objects with a decision rule set. To present the method of calculating the value of a decision rule set function, we introduce a definition below about matching objects to decision rules.

Definition 10.9 (Matching an object to a decision rule) An object o is said to match a decision rule $\phi \Rightarrow \psi$, if $o \vDash \phi$. Given a decision rule set RS, all decision rules in RS that are matched by object o are denoted as MR_{RS}^{o}.

With the definition, a brief method of calculating the result of a decision rule set function is described as follows:

Step 1: Calculate MR_{RS}^{o};

Step 2: Select a decision rule dr from MR_{RS}^{o}, where

$$dr\colon \wedge\{(a, v_a)\} \Rightarrow (d, v_d);$$

Step 3: Set the decision value of object o to be v_d, i.e. $rs(o) = v_d$.

In Step 2, how to select a decision rule from MR_{RS}^{o} is the key task of the process. For example, there is a decision rule set RS:

(1) $(a, 1) \wedge (b, 2) \Rightarrow (d, 2)$,

(2) $(a, 2) \wedge (b, 3) \Rightarrow (d, 1)$,

(3) $(b, 4) \Rightarrow (d, 2)$,

(4) $(b, 3) \wedge (c, 2) \Rightarrow (d, 3)$,

and an undecided object:

$$o = (a, 2) \wedge (b, 3) \wedge (c, 2).$$

With Step 1, $R_{RS}^{o} = \{(a, 2) \wedge (b, 3) \Rightarrow (d, 1);\ (b, 3) \wedge (c, 2) \Rightarrow (d, 3)\}$.

With Step 2, if the final rule is selected as $(a, 2) \wedge (b, 3) \Rightarrow (d, 1)$, then with Step 3, $rs(o) = 1$; if the final rule is selected as $(b, 3) \wedge (c, 2) \Rightarrow (d, 3)$, then with Step 3, $rs(o) = 3$.

From the above example, we know that there may be more than one rule in MR_{RS}^{o}. In this case, when the decision values of these rules are different, the result can be controlled according to above method, which is called uncertainty of a decision rule set function. The method of selecting the final rule from MR_{RS}^{o} is thus very important, and is called the uncertainty solution method. In this

chapter, we use the AID-based rule tree to deal with the problem, and more details about the method will be discussed in Sections 10.4 and 10.5.

10.2.4 Rule Trees

A rule tree is a compact and efficient structure for expressing a rule set. We first introduce the definition of rule trees (Zheng and Wang 2004) as follows.

Definition 10.10 (Rule tree)

(1) A rule tree is composed of one root node, multiple leaf nodes and middle nodes;
(2) The root node represents the whole rule set;
(3) Each path from the root node to a leaf node represents a rule;
(4) Each middle node represents an attribute testing. Each possible value of an attribute in a rule set is represented by a branch. Each branch generates a new child node. If an attribute is reduced in some rules, then a special branch is needed to represent it and the value of the attribute in this rule is assumed to be"∗", which is different from any possible values of the attribute.

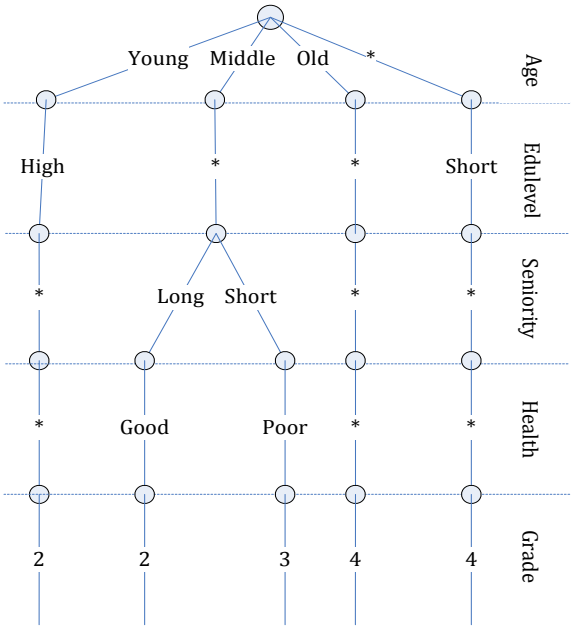

Figure 10.1 An example of a rule tree

When two nodes are connected with a branch, we call the upper node a start node and the lower node an end node.

Figure 10.1 gives an example of a rule tree, where "Age", "Educational level (Edulevel)", "Seniority", and "Health" are the names of its conditional attributes, and "Grade" is the name of its decision attribute. The values of these attributes are noted beside branches.

We define the number of nodes between a branch and the root node as the level of the branch (including the root node) in the path. For each rule tree, we have two assumptions as follows:

Assumption 10.1 The branches at the same level represent the possible values of the same attribute.

Here, an attribute is expressed by the level of a rule tree.

Assumption 10.2 If a rule tree expresses a decision rule set, the branches at the bottom level represent the possible values of the decision attribute.

Based on Definition 10.10 and the two assumptions, we can improve the rule tree structure with two constraints.

Definition 10.11 (Attribute importance degree (AID) based rule tree) An AID-based rule tree is a rule tree which satisfies the following two additional conditions:

(1) The conditional attribute expressed at the upper level is more important than that expressed at any lower level;

(2) Among the branches with the same start node, the value represented by the left branch is more important (or better) than the value represented by any right branch. Each possible value is more important (or better) than the value "*".

In the rule tree illustrated in Figure 10.1, if we suppose

- $ID(a)$ is the importance degree of attribute a, and ID (Age) $>$ ID (Edulevel) $>ID$ (Seniority) $>ID$ (Health);
- (Age, Young) is better than (Age, Middle), and (Age, Middle) is better than (Age, Old);
- (Seniority, Long) is better than (Seniority, Short), and (Health, Good) is better than (Health, Poor), then the rule tree illustrated by Figure 10.1 is an AID-based rule tree.

10.2.5 Rules Comparison

There are many reasons for uncertainty in a decision rule set function. First, the causality between two events cannot be measured easily, because there are many factors involved in the events and many different relations among these factors. When generating a decision rule set, there are some strict constraints and some less-important elements that are ignored in uncertainty analysis. Second, some empirical knowledge, especially perceptual knowledge, cannot be expressed precisely and accurately. Third, the decision rule language can induce uncertainty issues. In addition, knowledge stored in a rule base or database for a decision problem is often finite and incomplete, which is also a reason for uncertainty. More obviously, different learned objects, incomplete learning algorithms and some learning processes can cause uncertainty in a decision rule set function as well.

The uncertainty can be eliminated through a process of selection. We can select a rule correctly only when related information is known. In other words, we are said to be informed only when we can select rules accurately and definitely. In this chapter, we present a rule tree-based model to deal with the uncertainty. After the ordering of importance degrees and the possible values of attributes, a rule tree (Definition 10.10) is improved to become an AID-based rule tree (Definition 10.11). It can be proved that the following theorem holds.

Theorem 10.1 If we suppose the isomorphic trees to be the same, then there is a one-to-one correspondence between a rule set and an AID-based rule tree.

From the definitions of rule-set and AID-based rule trees, it is clear that the theorem holds.

Compared to a decision rule set, an AID-based rule tree has the following advantages:

(1) An AID-based rule tree is a more concise structure, especially when the scale of a rule set is huge;
(2) An AID-based rule tree is a more structured model which provides a number of useful properties such as confliction and repetition in an original decision rule set;
(3) It can speed up the searching and matching process on the decision rules;
(4) The rules in an AID-based rule tree are ordered, which provides a way to solve uncertainty problems in decision rule set functions.

Definition 10.12 (Comparison of rules) Suppose the condition attributes are ordered by their importance degrees as $a_1, \ldots, a_p$. Rule dr_1:$\wedge\{(a_i, v_{a1i})\} \Rightarrow (d_1, v_{d1})$ is said to be better than rule dr_2: $\wedge\{(a_i, v_{a2i})\} \Rightarrow (d_2, v_{d2})$, if $v_{a1k}$is

better than v_{a2k} or the value of a_k is deleted from rule dr_2, where $k \in \{1, \cdots, p\}$, and for each $j < k$, $v_{a1j} = v_{a2j}$.

For example, we have two rules:

dr_1: (Age, Middle) ∧ (Working Seniority, Long)⇒ 2,

dr_2: (Age, Middle) ∧ (Working Seniority, Short)⇒3, and the value "Long" of the attribute "Working Seniority" is better than the value "Short". With Definition 10.12, we know dr_1 is better than dr_2.

Theorem 10.2 In an AID-based rule tree, the rule expressed by the left branch is better than the rule expressed by the right branch.

This is evident from Definition 10.11 and Definition 10.12.

Theorem 10.3 After transformation to an AID-based rule tree, the rules in a rule set are totally in order, that is, every two rules can be compared.
It is evident that the theorem holds from Definition 10.11 and Theorem 10.2.

Example 10.2 We can order the rules expressed by the rule tree shown in Figure 10.1 as follows:

(1) $(\text{Age}, \text{Young}) \wedge (\text{Edulevel}, \text{High}) \Rightarrow 2$,
(2) $(\text{Age}, \text{Middle}) \wedge (\text{Working Seniority}, \text{Long}) \Rightarrow 2$,
(3) $(\text{Age}, \text{Middle}) \wedge (\text{Working Seniority}, \text{Short}) \Rightarrow 3$,
(4) $(\text{Age}, \text{Old}) \Rightarrow 4$,
(5) $(\text{Edulevel}, \text{Short}) \Rightarrow 4$,

where rule i is better than rule $i + 1$, $i = 1, 2, 3, 4$.

10.3 Rule-set-based Bi-level Decision Model

This section will present a RSBLD model, which uses a rule set rather than mathematic functions to express a bi-level decision problem. We will first discuss the representation of objectives and constraints in a RSBLD model, and then present the formulation of the model. Lastly, we will describe a modeling approach for solving RSBLD problems.

10.3.1 Objectives

As we have discussed before, a decision table can be used to lay out all possible situations in tabular form where a decision may be encountered, and to specify which action to take in each of these situations. Decision tables are commonly thought to be restricted in applicability to procedures involving sequencing of

tests, nested-IFs, or CASE statements. In fact, a decision table can implement any computable function. It is observed that any Turing Machine program can be "emulated" by a decision table by letting each Turing Machine instruction of the form (input, state) + (output, tape movement, state) be represented by a decision rule (or in a decision table) where (input, state) are conditions and (output, tape movement, state) are actions. From a more practical point of view, it can also be shown that all computer program flowcharts can be emulated by decision tables (Lew and Tamanaha 1976).

In principle, therefore, after emulating all possible situations in a decision domain, all objective functions can be transformed to decision tables, called *objective decision tables*. That is, the objectives of the leader and the follower in a bi-level decision problem can be transformed into a set of decision tables, where decision variables are represented by the objects in these decision tables.

Decision rule sets are general knowledge generated from decision tables and they have stronger knowledge-expressing ability than decision tables because they overcome the following disadvantages of decision tables:

(1) For complex situations, decision tables may become extremely large;

(2) The objects in decision tables lack adaptability. They are hard to adapt to new situations and one object can only record a single situation.

Thus, in the proposed model, we use a decision rule set to represent the objectives of the leader and the follower in a bi-level decision problem, whereas decision tables can be viewed as special cases of a decision rule set.

10.3.2 Constraints

Constraints (constraint conditions) can be seen as the description of the search space in a decision problem. Here, we use a rule set to represent constraints. Similar to the discussion in Section 10.3.1, after emulating all possible situations in the constraint field, the constraints can be formulated to an information table. When the information tables are too big to be processed, they can be transformed to a rule set using the methods provided by Agrawal et al. (1993) and Agrawal and Srikant (1994).

A rule set can be viewed as knowledge generated from information tables, but it has stronger knowledge-expressing ability and better adaptability than information tables. An information table can be viewed as a special case of rule sets. By using rule sets, we give the following definition about constraint conditions.

Definition 10.13 (Constraint Condition) Suppose x is a decision variable and RS is a rule set, then a constraint condition $cf(x, RS)$ is defined as

$$cf(x, RS) = \begin{cases} True, & \text{if for } \forall r \in RS, x \in m(r) \\ False, & \text{else.} \end{cases} \quad (10.1)$$

The meaning of the constraint condition $cf(x, RS)$ is whether variable x belongs to the region constrained by RS.

10.3.3 Rule-set-based Bi-level Decision Model

We can describe the objectives and constraints of the leader and the follower by rule sets, called a rule-set-based bi-level decision model, as follows.

Definition 10.14 (Rule-set-based bi-level decision model)

$$\begin{aligned} &\min_x f_L(x, y) \\ &\text{s.t. } cf(x, G_L) = True \\ &\quad \min_y f_F(x, y) \\ &\quad \text{s.t. } cf(y, G_F) = True \end{aligned} \quad (10.2)$$

where x and y are decision variables (vectors) of the leader and the follower respectively. f_L and f_F are the objective decision rule set functions of the leader and the follower respectively. cf is the constraint condition. F_L and G_L are the objective decision rule set and constraint rule set respectively of the leader, and F_L and G_L are the objective decision rule set and constraint rule set respectively of the follower.

In this model, we suppose that the decision rule set in the objectives can cover the objects in the constraints. That is, each object in the constraints can be matched by one decision rule at least in the objective decision rule set. The assumption is not too restricted, because when the objective decision tables used to generate objective decision rule set are huge and the objects in them are uniformly distributed, the resulting decision rule set usually covers most of the objects to be decided. In other cases, where some objects in the constraint fields cannot be matched by decision rules in the objective decision rule set, additional methods should be introduced, such as similarity matching, fuzzy marching, etc. In this chapter, we will discuss the models and decision methods based on the above assumption.

10.4 Rule-set-based Bi-level Decision Modeling Approach

In the following, we propose an approach for modeling a bi-level decision problem by rule sets.

Algorithm 10.1: Rule-set-based Bi-level Decision Modeling Approach

Input: A bi-level decision problem with its objectives and constraints of the leader and the follower;
Output: A rule-set-based bi-level decision model;

[Begin]
Step 1: Transform the bi-level decision problem with rule-set (information tables are as special cases);
Step 2: Pre-process F_L, such as delete reduplicate rules from the rule-set sets, eliminate noise, etc.;
Step 3: If F_L needs to be reduced, then use a reduction algorithm to reduce F_L;
Step 4: Pre-process G_L, such as delete reduplicate rules from the rule-set sets, eliminate noise, etc;
Step 5: If G_L needs to be reduced, then use a reduction algorithm to reduce G_L;
Step 6: Pre-process F_F, such as delete reduplicate rules from the rule-set sets, eliminate noise, etc.;
Step 7: If F_F needs to be reduced, then use a reduction algorithm to reduce F_F;
Step 8: Pre-process G_F, such as delete reduplicate rules from the rule-set sets, eliminate noise, etc.;
Step 9: If G_F needs to be reduced, then use a reduction algorithm to reduce G_F;
[End]

Figure 10.2 is the flow charts of the proposed rule-set-based bi-level decision problem modeling approach.

We provide explanations for the approach. Step 1 is the key step of the modeling process. Decision makers can complete the step by laying out all possible situations that is, transforming the decision problem to information tables. When the decision makers have the general knowledge (rules) for the problem, they can directly transform the problem to related rule sets. Therefore, the key in the step is decision makers' knowledge about the problem and their ability to write the knowledge into rules.

In Step 2, Step 4 and Step 6, each of the four rule sets is pre-processed. As incompleteness, noise and inconsistency are common characteristics of a huge real dataset we need to use related techniques to eliminate these problems before using the data to make a bi-level decision (Han and Kamber 2001).

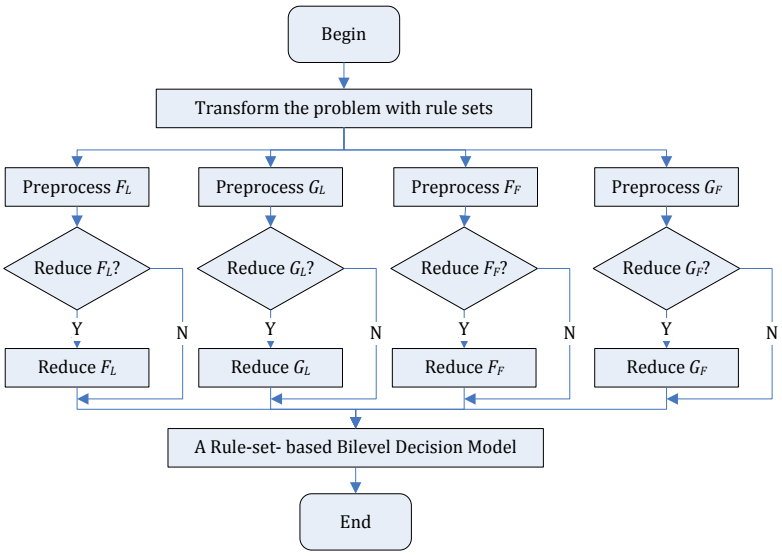

Figure 10.2 Flow chart of the RSBLD modeling approach for modeling rule-set-based bi-level decision problems (Algorithm 10.1)

In Step 5, Step 7, and Step 9 of this algorithm, related rule sets are reduced by applying a reduction algorithm. This is for at least one of the following three reasons:

(1) When modeling a real-world bi-level decision problem, the rule set in the model is often on a large scale, which is not convenient to process, and cannot be easily interpreted and understood by decision makers.

(2) The rules in the rule set lack adaptability. In this case, the rule set cannot adapt to new situations well, so it is unable or has poor ability to support decision making.

(3) The rule sets in the model are original data sets, so the patterns in the data sets need to be extracted, and the results are more general rules.

To reduce the size of a decision rule set or to extract decision rules from a decision table, the rough set-based methods from literature are efficient. Many rough set-based decision rule extraction and reduction algorithms, called value reduction algorithms, have been developed (Pawlak 1991; Hu and Cercone 1995; Mollestad and Skowron 1996; Wang 2001; Wang and He 2003; Zheng and Wang 2004). These algorithms have been applied successfully in many fields. Some rough set-based systems, such as ROSETTA, RIDAS (Wang et al. 2002), and RSES (Jan et al. 2002), can be used to reduce the size of a decision rule set or extract a decision rule set from a decision table. Therefore, we use a rough set-based method to handle the issue.

To reduce the size of the constraint rule set or to generate constraint rules from information tables, some effective methods have been developed, such as Apriori algorithm (Agrawal et al. 1993), Fast algorithms for mining association rules (Agrawal and Srikant 1994), and FP tree algorithm (Han et al. 2000). We can select one of them to complete this task. Also, we have found that in many cases, when the constraint rules are obvious or already known, the rule generation process can be passed over.

10.5 Rule-set-based Bi-level Decision Solution Algorithms

In this section, we present two algorithms to solve rule-set-based bi-level decision models, in which a key technique is that an AID-based rule tree is used to express a rule set.

10.5.1 Concepts and Properties

Based on Bard (1998), we have the following definition.

Definition 10.15

(a) Constraint region of a bi-level decision problem:

$$S = \{(x, y) \mid cf(x, G_L) = True, cf(y, G_F) = True\} \tag{10.3}$$

(b) Feasible set for the follower for each fixed x:

$$S(x) = \{y \mid (x, y) \in S\} \tag{10.4}$$

(c) Projection of S onto the leader's decision space:

$$S(x) = \{x \mid \exists y, (x, y) \in S\} \tag{10.5}$$

(d) Follower's rational reaction set for $x \in S(X)$:

$$P(x) = \left\{y \mid y \in \arg\min_{y}\{f_F(x, y') \mid y' \in S(x)\}\right\} \tag{10.6}$$

(e) Inducible region:

$$IR = \{(x, y) \mid (x, y) \in S, y \in P(x)\} \tag{10.7}$$

To ensure that (10.2) is well posed it is common to assume that S is nonempty and compact, and that for all decisions taken by the leader, the follower has some room to respond, i.e. $P(x) \neq \emptyset$. The rational reaction set $P(x)$ defines the response while the inducible region IR represents the set over which the leader may optimize its objective. Thus in terms of the above notation, the bi-level decision problem can be written as

$$\min\{f_L(x,y) \mid (x,y) \in IR\} \tag{10.8}$$

From the features of the bi-level decision, it is clear that once the leader selects an x, the first term in the follower's objective function becomes a constant and can be removed from the problem. In this case, we replace $f_F(x,y)$ with $f_F(y)$.

To begin, let $(x_{[1]}, y_{[1]}), (x_{[2]}, y_{[2]}), \ldots, (x_{[N]}, y_{[N]})$ denote the N ordered feasible solutions to the rule-set-based one level one objective problem

$$\min_x\{f_L(x,y) \mid (x,y) \in S\} \tag{10.9}$$

such that

$$f_L(x_{[i]}, y_{[i]}) \leq f_L(x_{[i+1]}, y_{[i+1]}) \tag{10.10}$$

and $i = 1, \ldots, N-1$. Thus, to solve (10.9) is equivalent to finding the global optimum $(x_{[k]}, y_{[k]})$.

10.5.2 Rule-based-based Solution Algorithm

We present an algorithm for solving a rule-set-based bi-level decision problem. The main picture of the algorithm is to repeatedly solve two rule-set-based one-level decision problems. One is for the leader in all of the variables x and a subset of the variables y associated with an optimal basis to the follower's problem, and the other is for the follower with all the variables x fixed. The leader first makes his decision, and the decision will influence the objective rule set function and the constraint rule set function of the follower. In a systematic way, the algorithm explores the optimal solution of the follower's problem for x fixed and then returns to the leader's problem with the corresponding variables y. If the variables y are not an optimal solution of the leader's decision problem, then the leader modifies his decision and engages in the procedures repeatedly.

Algorithm 10.2: Rule-set-based Solution Algorithm

Input: The objective decision rule set $F_L = \{dr_{L1}, \dots, dr_{Lp}\}$ and the constraint rule set G_L of the leader, the objective decision rule set $F_F = \{dr_{F1}, \dots, dr_{Fq}\}$ and the constraint rule set G_F of the follower;
Output: The final decision of the leader and the follower $(x_{[i]}, y_{[i]})$.

[Begin]
Step 1: Construct the objective rule tree FT_L of the leader by F_L;
Step 1.1: Arrange the condition attributes in ascending order according to the importance degrees. Let the attributes be the discernible attributes of levels from the top to the bottom of the tree;
Step 1.2: Initialize FT_L to an empty AID-based rule tree;
Step 1.3: For each rule R of the decision rule set F_L;
Step 1.3.1: Let$CN \leftarrow$ root node of the rule tree FT_L;
Step 1.3.2: For$i = 1$ to m; /*m is the number of levels in the rule tree */
If there is a branch of CN representing the ith discernible attribute value of rule R, then
let$CN \leftarrow$node *I*; /*node *I* is the node generated by the branch*/
else {Create a branch of *CN* to represent the *i*th distensibility attribute value;
According to the value order of the *i*th discernible attribute, put the created branch to the right place;
Let CN←node *J*. /*node *J* is the end node of the branch*/
Step 2: Construct the objective rule tree FT_F of the follower byF_F;
The detail of Step 2 is similar to that of Step 1. It is necessary to replace FT_LwithFT_F and replace F_L with F_F in the sub-steps of Step 1.
Step 3: Solve problem (10.9) to obtain its optimal solution $(x_{[i]}, y_{[i]})$, and initialize $i = 1$;
Step 3.1: Initialize $FT_L^{'}$ to an empty AID-based rule tree, where $FT_L^{'}$ represents the objective rule tree of the follower pruned by the constraint rule set;
Step 3.2: Use the constraint rule tree sets G_L andG_F to prune FT_L;
For each rule dr in G_L andG_F,
Add the rules in FT_L that are consistent with dr to$FT_L^{'}$;
Delete the rules in FT_L and$FT_L^{'}$ that are conflict with dr;
Step 3.3: Search for the rules with the minimal decision value in $FT_L^{'}$ and the result rule set is $RS = \{dr_1, \dots, dr_m\}$, where dr_1 to dr_m are the rules from left to right in $FT_L^{'}$;
Step 3.4: Let dr be the first rule in RS;
Step 3.5: $RS = RS \setminus \{dr\}$;
$OS = \{o | o$ is the objects consistent with dr and the constraint rule sets$\}$;

Step 3.6: Order the objects in OS so that the ith object in OS is better than the $(i+1)$th object in OS according to Definition 10.12;

Step 3.7: The solution of the problem (10.9) is the first object (Definition 10. 12) in OS.

Step 4: Let $FT_F^{'}$ be the objective rule tree of the follower pruned by the constraint RULE SET of the leader and the follower, and $W = \psi$;

Step 5: Prune the rules from $FT_F^{'}$, which are not consistent with rule

$$True \Rightarrow x_{[i]}$$

and suppose the result is a rule tree $FT_F^{''}$;

Step 6: Solve the follower's rule-set-based decision problem:

$$\min_{y}\{f_F(x_{[i]}, y): y \in P(x_{[i]})\} \tag{10.11}$$

and get the optimal solution $(x_{[i]}, y)$;

Step 6.1: Search for the rules with the minimal decision value in $FT_F^{''}$ and the result rule set is $RS^{'} = \{dr_1^{'}, \ldots, dr_m^{'}\}$;

Step 6.2: Let $dr^{'}$ be the first rule in $RS^{'}$;

Step 6.3: $RS^{'} = RS^{'} \setminus \{dr^{'}\}$; $OS^{'} = \{o^{'} \mid o^{'}$ is the objects consistent with $dr^{'}$ and the constraint rule set$\}$;

Step 6.4: Order the objects in $OS^{'}$ so that the ith object in $OS^{'}$ is better than the $(i+1)$th object in $OS^{'}$ according to Definition 10.12;

Step 6.5: The solution of the follower's problem is the first object (Definition 10.12) in $OS^{'}$.

Step 7: If $y = y_{[i]}$, then {

The optimal solution set is obtained, which is $(x_{[i]}, y_{[i]})$; End;}

Else {

Go to Step 8;

};

Step 8: Select another solution for the follower;

Step 8.1: $OS^{'} = OS^{'} \setminus \{(x_{[i]}, y)\}$;

Step 8.2: If OS$^{'}$ is null, then {

If $RS^{'}$ is null, then {

Go to Step 9;}

else {

Let $dr^{'}$ be the first rule in RS;

$RS^{'} = RS^{'} \setminus \{dr^{'}\}$;

$OS^{'} = \{o^{'} \mid o^{'};$ is the objects consistent with $dr^{'}$ and the constraint rule-set$\}$;

Step 8.3: Order the objects in OS' and the next solution of the follower's problem is the first object $(x_{[i]}, y)$ (Definition 10.12) in OS'.

Step 8.4: Go to Step 7.

Step 9: Select another solution for the leader;

Step 9.1: $OS = OS \setminus \{(x_{[i]}, y_{[i]})\}$;

Step 9.2: $W = W \cup \{(x_{[i]}, y_{[i]})\}$

Step 9.3: If OS is null, then {

If RS is null, then {

There is no optimal solution for the problem;

End;}

Else {

Let dr be the first rule in RS;

$RS = RS \setminus \{dr\}$;

$OS = \{o | o$ is the objects consistent with dr and the constraint rule set;}

Let $(x_{[i+1]}, y_{[i+1]})$ be the first object in OS;

$i = i + 1$;

Go to Step 5;}.

[End]

Figure 10.3 shows the flow charts of Algorithm 10.2. We will use an example to illustrate the algorithm.

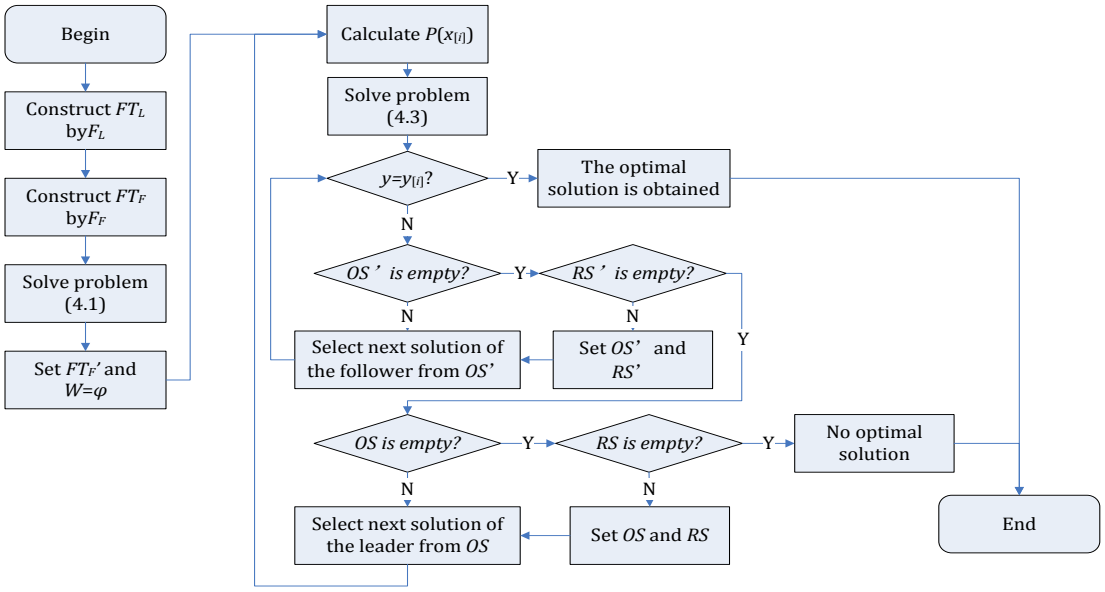

Figure 10.3 Flow charts of the approach for solving rule-set based bi-level decision model

10.5.3 Transformation-based Solution Algorithm

This section presents an alternative algorithm for solving a rule-set-based bi-level decision problem. The main idea of this algorithm is to transform two level rule-sets to one level first, then to solve this one-level decision. Before developing the algorithm, a transformation theorem will be proposed to show the solution equivalence for the two problems before and after transformation.

First, we give a definition below.

Definition 10.16 (Combination rule of two decision rules) Suppose dr_1: $\phi_1 \Rightarrow (d_1, v_1)$ and dr_2: $\phi_2 \Rightarrow (d_2, v_2)$ are two decision rules and they are not in conflict, then their combination rule is denoted as $dr_1 \bigcap dr_2$ with the form

$$\phi_1 \wedge \phi_2 \Rightarrow (d, (v_1, v_2)),$$

where d_1, d_2, and d are the decision attributes of dr_1, dr_2 and dr respectively, v_1, v_2 and (v_1, v_2) are the decision values of dr_1 and dr_2 and dr respectively.

Here, v_1, v_2 are called the leader decision and the follower decision of dr respectively. For example, suppose

dr_1: (Age, Young) $\Rightarrow$ 2,

dr_2: (Working Seniority, Long) $\Rightarrow$ 2,

then the combination of the two rules is

dddr: (Age, Young) $\wedge$ (Working Seniority, Long) $\Rightarrow$ (*d*, (2, 2)).

Suppose the objective rule sets are expressed by AID-based rule trees, then the transformation process can be presented as follows.

[Begin]

Step 1: (Initialization): Let *CT* be an empty attribute importance degree-based rule tree;

Step 2: (Construct a new rule tree):

For each rule dr_L in FT_L

For each decision rule dr_F in FT_F

{If dr_L are not conflict with dr_F, then

Add rule $dr_L \bigcap dr_F$ to *CT*;}

[End]

Suppose the combined rule set is noted as *F*, then the single level rule-sets based decision problem can be formulated as:

$$\min_{x,y} f(x, y)$$

$$\text{s.t. } cf(x, G_L)=\text{True},$$

$$cf(y, G_F)=\text{True}, \tag{10.12}$$

where x and y are variables of the leader and the follower respectively; f is the objective decision rule-set function; cf is the constraint function; F, G_L, G_F are the objective decision rule set, leader's constraint rule set and follower's constraint rule set respectively.

With the following theorem, we can prove the solution equivalence of the original problems and the transformed problem.

Theorem 10.1 The RSBLD model presented in Equation 10.2 has an optimal solution (x, y), iff (x, y) is an optimal solution of its corresponding single level decision model presented in Equation 10.12.

Proof: Suppose x and y are variables of the leader and the follower respectively, f_L and f_F are the objective rule-set functions of the leader and the follower respectively in Equation 10.2, and f is the objective rule set function in Equation 10.12. F_L and F_F are the objective rule sets of the leader and the follower in the RSBLD model, and F is the objective rule set in the single level decision model.

If the optimal solution of the RSBLD model presented in Equation 10.2 is (x, y), and

$$f_L(x, y)=v_L \text{ and } f_F(x, y)=v_F.$$

Suppose the final matching rules (Section 10.2) of (x, y) in rule sets F_L and F_F are dr_L and dr_F respectively. Then, from the process of transformation, we know the rule $dr_L \bigcap dr_F$ belongs to the combined rule set F.

Because (x, y) is the optimal solution of the RSBLD model, dr_L and dr_F must be the best rules, having the minimal decision values in F_L and F_F respectively. Thus, $dr=dr_L \bigcap dr_F$ must be the best rules matched by (x, y) in F. Besides, because (x, y) is the best object satisfying both dr_L and dr_F , (x, y) is the best object satisfying dr. Thus, (x, y) is the optimal solution of the single level decision model presented in Equation 10.12.

The sufficient condition of the theorem is proved.

If the optimal solution of the single level decision model presented in Equation 10.12 is (x, y), and

$$f(x, y)=(d, (v_{Ld}, v_{Fd})).$$

Suppose the final matching rule of (x, y) in rule set F is dr, then from the process of transformation, there must be two decision rules dr_L in F_L and dr_F in F_F that $dr=dr_L \bigcap dr_F$. If there is more than one rule pair dr_L and dr_F satisfying that $dr=dr_L \bigcap dr_F$, then select the best one among them.

Because (x, y) is the optimal solution of the single level decision model, dr must be the best rules having the minimal decision value in F. Thus, dr_L and dr_F must be the best rules matched by (x, y) in F_L and F_F respectively. Besides,

because (x, y) is the best object satisfying dr, (x, y) is the best object satisfying both dr_L and dr_F both, so (x, y) is the optimal solution of the bi-level decision model.

Thus, the necessary condition of the theorem is proved.

From Theorem 10.1, it is known that the optimal solution of the RSBLD problem presented in Equation 10.2 and its transformation problem shown in Equation 10.12 are equivalent. Therefore, any RSBLD problem can be transformed into a single level decision problem. Furthermore, a solution is achieved by solving the single level decision problem. Note that although the original bi-level decision problem and the transformed one level problem have the same optimal solution, they are not equivalent. Some information of the leader and the follower unrelated to the acquiring of the optimal solution is reduced during the transformation process, because the aim of transformation is only to generate a model which can be easily solved but has the same optimal solution as the original bi-level decision model.

Based on the transformation theorem proposed, we will develop a transformation-based solution algorithm for RSBLD problems in the following sections.

Algorithm 10.3: Rule-set-based Transformation Solution Algorithm

Input: The objective decision rule set $F_L = \{dr_{L1}, \dots, dr_{Lp}\}$ and the constraint rule set G_L of the leader, the objective decision rule set $F_F = \{dr_{F1}, \dots, dr_{Fq}\}$ and the constraint rule set G_F of the follower;

Output: An optimal solution of the RSBLD problem (ob).

[Begin]

Step 1: Construct the objective rule tree FT_L of the leader by F_L;

Step 1.1: Arrange the condition attributes in ascending order according to the importance degrees. Let the attributes be the discernible attributes of levels from the top to the bottom of the tree;

Step 1.2: Initialize $FT_L$to an empty AID-based rule tree;

Step 1.3: For each rule R of the decision rule set F_L {

Step 1.3.1: Let CN ←root node of the rule tree FT_L;

Step 1.3.2: For $i = 1$ to m /*m *is the number of levels in the rule tree**/

If there is a branch of CN representing the ith discernible attribute value of rule R, then

let CN ←node I; /*node I is the node generated by the branch*/

else {Create a branch of CN to represent the *i*th discernible attribute value;

According to the value order of the *i*th discernible attribute, put the created branch to the right place;

Let *CN*←node *J*. /*node *J* is the end node of the branch*/

Step 2: Construct the objective rule tree FT_F of the follower byF_F;

The detail of Step 2 is similar to that of Step 1. It is necessary to replace FT_L with FT_F and to replace F_L with F_F in the sub-steps of Step 1.

Step 3: Transform the bi-level decision problem to a single level one, and the resultant objective rule tree is CT;

Step 4: Use the constraint rule set of both the leader and the follower to prune CT;

Step 4.1: Generate an empty new AID-based rule tree CT';

Step 4.2: For each rule dr in G_L and G_F;

Add the rules in CT to CT' that are consistent with dr to FT_L';

Delete the rules in CT and CT'that conflict with dr;

Step 4.3: Let $CT = CT'$;

Step 5: Search for the leftmost rule dr in CT whose leader decision and follower decision are both minimal;

Step 6: If dr does not exist, then there is no optimal solution for the problem and go to end;

Step 7: $OB = \{ob | ob \vDash dr \text{and for} \, r, G_L, G_F, ob \vDash r\}$;

Step 8: If there is more than one object in OB, then according to Definition 10.12, select the best or most important object ob; else ob=the object in OB;

Step 9: ob is the optimal solution of the RSBLD problem;

[End]

The flow chart of the algorithm is illustrated in Figure 10.4. By this algorithm, we can obtain a solution for a rule-set-based bi-level decision problem through solving the transformed single level problem.

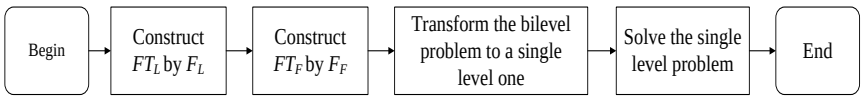

Figure 10.4 Flow chart of Algorithm 10.3

10.6 A Case Study

In this section, we use the RSBLD approach and algorithms introduced to handle the recruit decision problem given in Example 10.1.

10.6.1 Problem Modeling

Now, we use Algorithm 10.1 to transform the recruit problem to a RSBLD model:

Step 1 transforms the recruit problem with rule sets (information tables as special cases).

Table 10.2 and Table 10.3 represent the objective rule set of the leader, and the follower respectively.

Table 10.2 Objective rule set of the leader

Age	Edulevel	Seniority	Health	Grade
Young	High	Middle	Good	2
Middle	High	Long	Middle	2
Young	Short	Short	Poor	4
Young	Middle	Middle	Middle	2
Middle	Middle	Short	Middle	3
Middle	Middle	Long	Middle	2
Old	High	Long	Middle	3
Young	Short	Middle	Poor	2
Middle	Short	Short	Middle	4
Old	Short	Middle	Poor	4
Middle	Short	Long	Good	3
Middle	Short	Long	Middle	2
Old	High	Middle	Poor	3
Old	High	Long	Good	2
Old	Short	Long	Good	4
Young	High	Long	Good	4
Young	Short	Long	Middle	3

Equations 10.12 and 10.13 give the constraint rule sets of the leader and the follower respectively.

The constraint rule set of the leader is

$$G_L = \{\text{True} \Rightarrow (\text{Age, Young}) \vee (\text{Age, Middle})\}. \quad (10.12)$$

The constraint rule set of the follower is

$$G_F = \{True \Rightarrow (\text{Seniority, Long}) \vee (\text{Seniority, Middle})\}. \quad (10.13)$$

Because the scale of the data is very small, the pre-process steps (Steps 2, 4, 6 and 8) are passed over. Besides, the constraint rule set of the leader and the

follower are brief enough, so the reduction steps of G_L and G_F (Step 5 and Step 9) can be ignored.

Step 3 and Step 7 reduce the objective rule set of the leader and the follower.

Table 10.3 Objective rule set of the follower

Age	Edulevel	Seniority	Health	Grade
Young	High	Long	Good	2
Old	Short	Short	Good	4
Young	High	Short	Good	2
Old	High	Long	Middle	3
Young	Short	Long	Middle	4
Middle	High	Middle	Poor	3
Middle	Short	Short	Poor	4
Old	Short	Short	Poor	4
Old	High	Long	Good	2
Young	Short	Long	Good	2
Young	Short	Middle	Middle	3
Middle	Short	Middle	Good	3
Old	High	Long	Good	2
Middle	High	Long	Good	2
Middle	High	Short	Poor	4

After reducing the decision tables based on the rough set theory given in Section 10.2.5, we can obtain the reduced objective rule set of the leader and the follower as shown in Equations 10.14 and 10.15. Here, we use the decision matrices-based value reduction algorithm (Ziarko et al. 1996) in the RIDAS system (Wang et al. 2002).

The reduced objective rule set of the leader:

$F_L = \{$

$(\text{Age, Young}) \wedge (\text{Seniority, Middle}) \Rightarrow (\text{Grade, 2}),$

$(\text{Age, Middle}) \wedge (\text{Edulevel, High}) \Rightarrow (\text{Grade, 2}),$

$(\text{Edulevel, Short}) \wedge (\text{Seniority, Short}) \Rightarrow (\text{Grade, 4}),$

(Edulevel, Middle)∧(Seniority, Short)⇒(Grade, 3),

(Edulevel, Middle)∧(Seniority, Long)⇒(Grade, 2),

(Age, Old)∧(Health, Middle)⇒(Grade, 3),

(Age, Old)∧(Edulevel, Short)⇒(Grade, 4),

(Age, Middle)∧(Health, Good)⇒(Grade, 3),

(Age, Middle)∧(Seniority, Long)∧(Health, Middle)⇒(Grade, 2),

(Age, Old)∧(Edulevel, High)∧(Health, Good)⇒(Grade, 2),

(Edulevel, High)∧(Health, Poor)⇒(Grade, 3),

(Age, Young)∧(Edulevel, High)∧(Seniority, Long)⇒(Grade, 4),

(Age, Young)∧(Edulevel, Short)∧(Seniority, Long)⇒(Grade, 3)

}. (10.14)

The reduced objective rule set of the follower:

$F_F = \{$

(Edulevel, High) ∧ (Health, Good) $\Rightarrow$ (Grade, 2)

(Edulevel, Short) ∧ (Seniority, Short) $\Rightarrow$ (Grade, 4)

(Age, Old)∧ (Health, Middle) $\Rightarrow$ (Grade, 3)

(Age, Young)∧ (Seniority, Long) ∧ (Health, Middle) $\Rightarrow$ (Grade, 4)

(Seniority, Middle) $\Rightarrow$ (Grade, 3)

(Seniority, Long) ∧ (Health, Good) $\Rightarrow$ (Grade, 2)

(Seniority, Short) ∧ (Health, Poor) $\Rightarrow$ (Grade, 4)

}. (10.15)

With the above steps, we get the RSBLD model of the decision problem, as follows:

$$\min_x f_L(x, y)$$

$$\text{s.t. } cf(x, G_L) = True,$$

$$\min_y f_F(x, y)$$

$$\text{s.t. } cf(y, G_F) = Ture$$

where f_L, f_F are the corresponding decision rule set functions of F_L, F_F respectively.

10.6.2 Solution

Now, we use Algorithm 10.2 to solve the bi-level decision problem given in Section 10.6.1. We suppose that the four condition attributes are ordered by attribute importance degrees as "age", "edulevel", "seniority", "health".

Step 1: Construct the objective rule tree FT_L of the leader by F_L and the result is as shown in Figure 10.4;

Step 2: Construct the objective rule tree FT_F of the follower by F_F and the result is as shown in Figure 10.5;

Step 3: Solve problem (10.9), and initialize $i = 1$.

Step 3.1: Let $FT_L^{'}$ be the objective rule tree of the follower pruned by the constraint rule sets, and initialize $FT_L^{'}$ to an empty AID-based rule tree;

Step 3.2: Use the constraint rule tree GT_L to prune FT_L and the result is $FT_L^{'}$ as Figure 10.7;

Step 3.3: Search for the rules with the minimal decision value in $FT_L^{'}$ and the result rule set is

$RS = \{$

(Age, Young)∧ (Seniority, Middle) $\Rightarrow$ (Grade, 2);

(Age, Middle)∧ (Edulevel, High) $\Rightarrow$ (Grade, 2)

(Age, Middle)∧ (Seniority, Long)∧ (Health, Middle) $\Rightarrow$ (Grade, 2)

(Edulevel, Middle)∧ (Seniority, Long) $\Rightarrow$ (Grade, 2)

};

Step 3.4:

dr: (Age, Young)∧ (Seniority, Middle) $\Rightarrow$ (Grade, 2)

Steps 3.5-3.6:

$RS = \{$

(Age, Middle)∧ (Edulevel, High) $\Rightarrow$ (Grade, 2)

(Age, Middle)∧ (Seniority, Long)∧ (Health, Middle) $\Rightarrow$ (Grade, 2)

(Edulevel, Middle)∧ (Seniority, Long) $\Rightarrow$ (Grade, 2)

};

$OS = \{$

(Age, Young)∧(Edulevel, High)∧(Seniority, Middle)∧(Health, Poor),

(Age, Young)∧(Edulevel, Middle)∧(Seniority, Middle)∧(Health, Good),

(Age, Young)∧(Edulevel, Middle)∧(Seniority, Middle)∧(Health, Middle),

(Age, Young)∧(Edulevel, Middle)∧(Seniority, Middle)∧(Health, Poor),

(Age, Young)∧(Edulevel, Poor)∧(Seniority, Middle)∧(Health, Good),

(Age, Young)∧(Edulevel, Middle)∧(Seniority, Middle)∧(Health, Middle),

(Age, Young)∧(Edulevel, Middle)∧(Seniority, Middle)∧(Health, Poor)

}

Step 3.7: The solution of problem (4.1) is the first object in OS, that is

o = (Age, Young)∧ (Edulevel, High)∧ (Seniority, Middle)∧ (Health, Good)

Step 4: Let $FT_F^{'}$ be the objective rule tree of the follower pruned by the constraint rule set and $FT_F^{'}$ is as Figure 10.8. $W = \psi$;

Step 5: $x_{[1]}$ = (Age, Young)∧(Edulevel, High). Prune the rules from $FT_F^{'}$, which are not consistent with

$$True \Rightarrow x_{[1]}$$

and suppose the result is the rule tree $FT_F^{''}$ as shown in Figure 10.9;

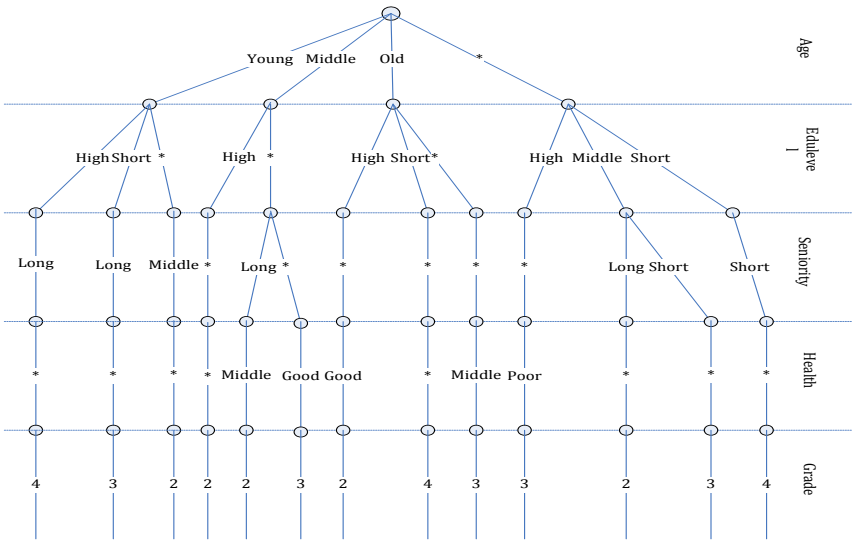

Figure 10.5 Rule tree of the leader's objective rule set

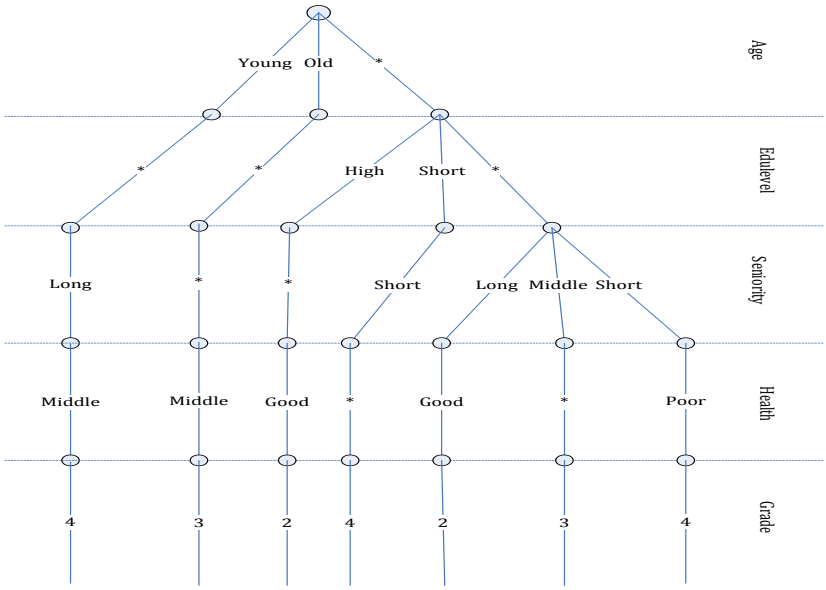

Figure 10.6 Rule tree of the follower's objective rule set

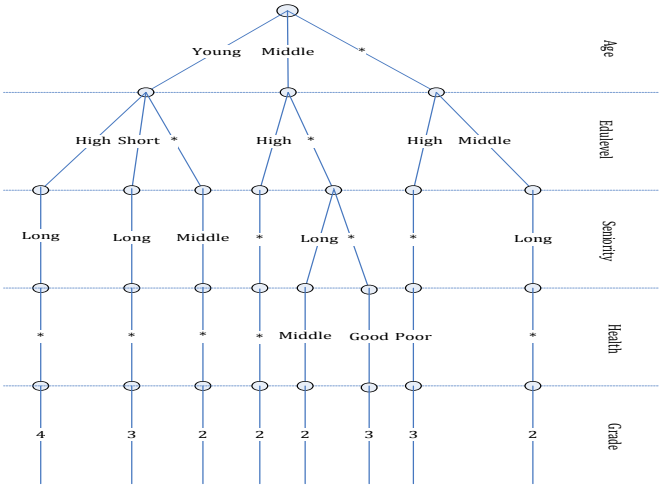

Figure 10.7 Rule tree of the leader's objective rule set after cutting by constraint rule set

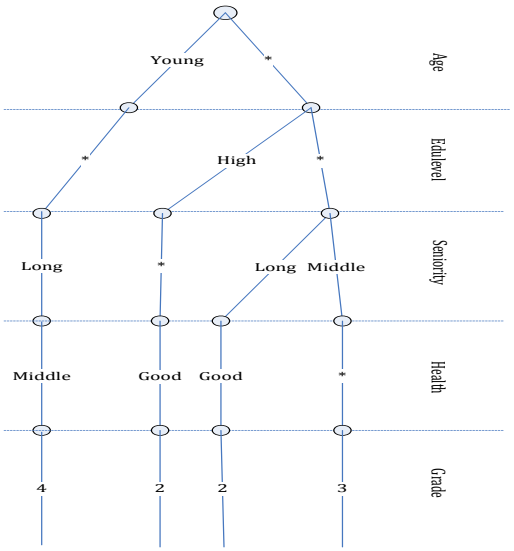

Figure 10.8 The Rule tree of the follower's objective rule set after cutting by constraint rule set

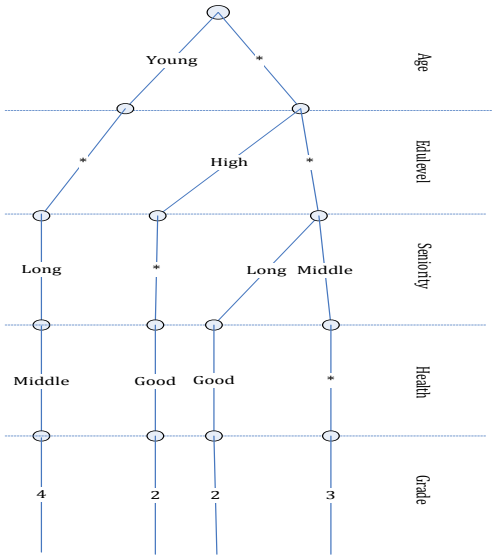

Figure 10.9 Rule tree of the follower's objective rule set after pruning the rules which are not consistent with $x_{[i]}$

Step 6: Solve the follower's rule-set-based decision problem below.

$$\min_{y}\{f_F(x_{[i]}, y): y \in P(x_{[i]})\}$$

Step 6.1: Search for the rules with the minimal decision value in $F_T^{''}$ and the result rule set is

$RS^{'}$ = {

(Edulevel, High) ∧ (Health, Good) ⇒ (Grade, 2)

(Seniority, Long) ∧ (Health, Good) ⇒ (Grade, 2)

};

Step 6.2:

$dr^{'}$: (Edulevel, High) ∧ (Health, Good) ⇒ (Grade, 2)

Steps 6.3-6.4:

$RS^{'}$ = {(Seniority, Long) ∧ (Health, Good) ⇒ (Grade, 2)};

$OS^{'}$ =

(Age, Young)∧ (Edulevel, High)∧ (Seniority, Long)∧ (Health, Good),

(Age, Young)∧(Edulevel, High)∧(Seniority, Middle)∧(Health, Good),

(Age, Middle)∧(Edulevel, High)∧(Seniority, Long)∧(Health, Good),

(Age, Middle)∧(Edulevel, High)∧(Seniority, Middle)∧(Health, Good)

}

Step 6.5: The solution of the follower's problem is

(Age, Young)∧ (Edulevel, High)∧ (Seniority, Long)∧ (Health, Good).

Step 7: Because $y \neq y_{[i]}$, Go to Step 8;

Step 8:

Step 8.1-8.2:

$OS' = \{$

(Age, Young)∧ (Edulevel, High)∧ (Seniority, Middle)∧ (Health, Good),

(Age, Middle)∧ (Edulevel, High)∧ (Seniority, Long)∧ (Health, Good),

(Age, Middle)∧ (Edulevel, High)∧(Seniority, Middle)∧(Health, Good)

},

Step8.3: The next solution of the follower's problem is

$o^{'}$ = (Age, Young)∧ (Edulevel, High)∧ (Seniority, Middle)∧ (Health, Good)

Step 8.4: Go to Step 7;

Step 7: Because $y = y_{[i]}$, the optimalsolution for the bi-level decision problem is

(Age, Young)∧ (Edulevel, High)∧ (Seniority, Middle)∧ (Health, Good).

[End]

The solution with the variables of both the factory executive committee and the workshop management committee will be used in the factory's decision in recruiting new workers. It will maximize the ability to satisfy the objectives of decision making at the two levels.

Now, we use the Algorithm 10.3 to solve the RSBLD problem. We suppose that the four condition attributes are ordered as "age", "edulevel", "seniority", and "health".

Step 1: Construct the objective rule tree FT_L of the leader by F_L, and the result is illustrated by Figure 10.10;

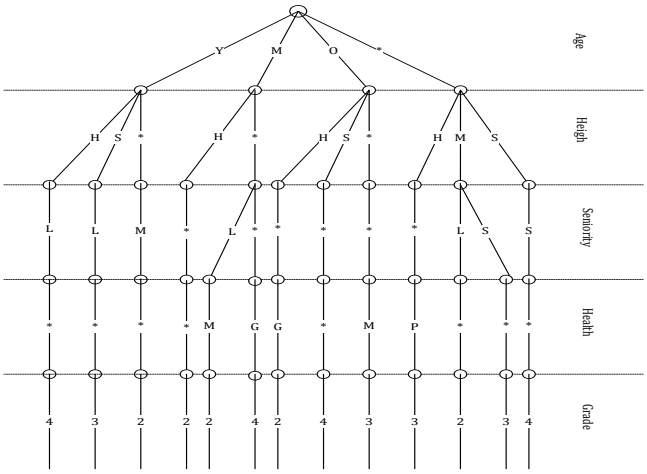

Figure 10.10 Rule tree of the leader's objective rule set

Step 2: Construct the objective rule tree FT_F of the follower by F_F, and the result is illustrated by Figure 10.11;

Step 3: Transform the RSBLD problem to a single level one, and the resulting objective rule tree CT is illustrated by Figure 10.12;

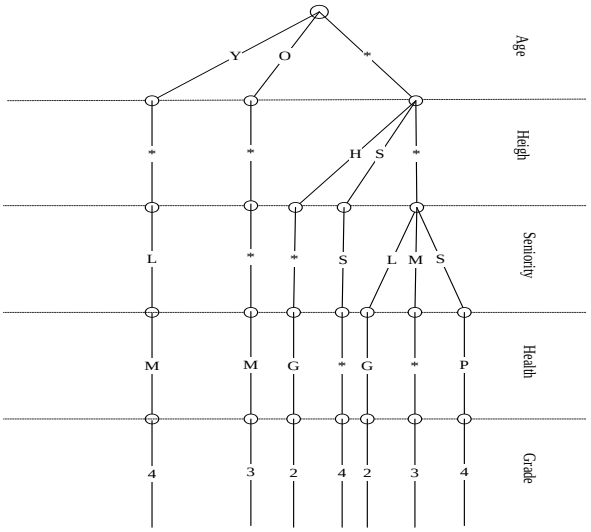

Figure 10.11 Rule tree of the follower's objective rule set

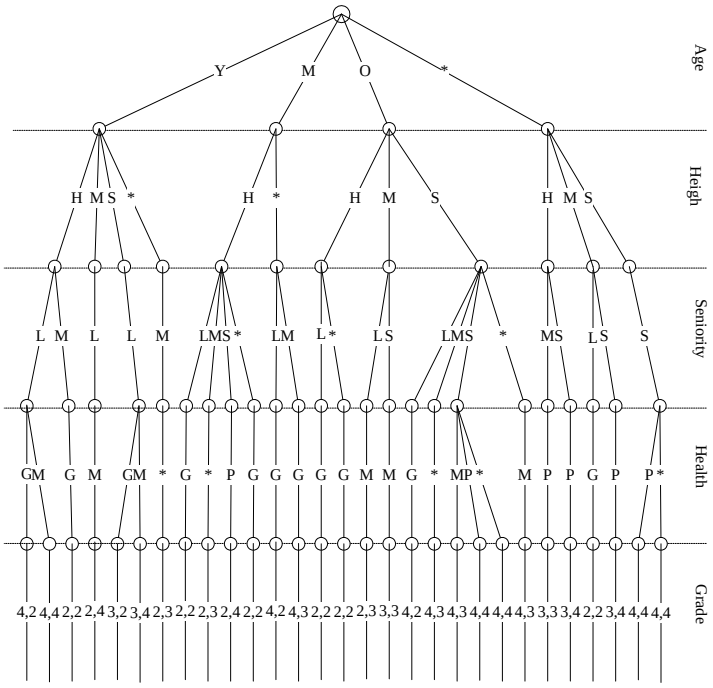

Figure 10.12 Transformation result of the objective rule trees

Step 4: Use the constraint rule set of both the leader and follower to prune CT, and the result is illustrated by Figure 10.13;

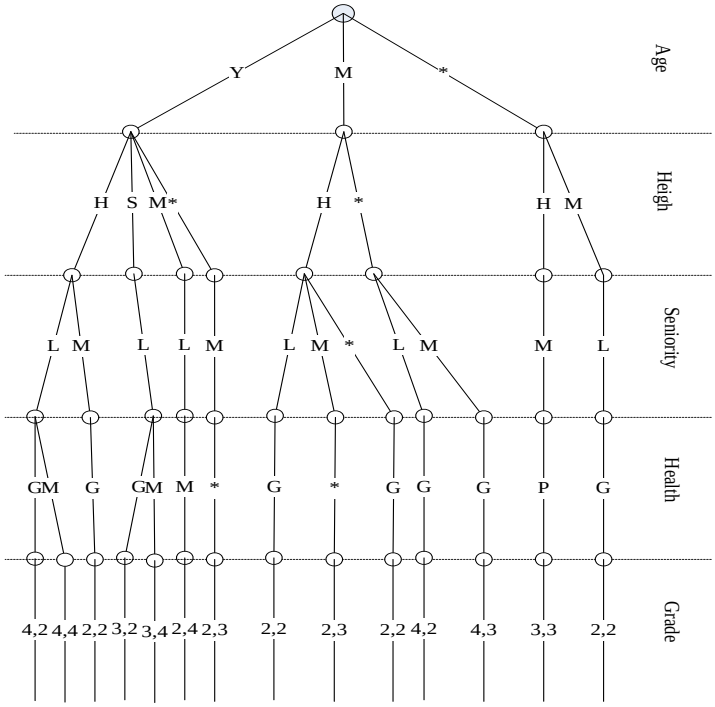

Figure 10.13 Combined objective rule trees after pruning by the constraint rules

Step 5: Search for the leftmost rule dr in CT whose leader decision and follower decision are both minimal, and the result is
dr: (Age, Young) (Edulevel, High) (Seniority, Middle) (Health, Good) (d, (2, 2));

Step 6: $OB = \{ob | ob$ is the object satisfying:
(Age, Young) (Edulevel, High) ((Seniority, Middle) (Health, Good) };

Step 7: $ob =$ (Age, Young) (Edulevel, High) (Seniority, Middle) (Health, Good);

Step 8: ob is the final solution of the RSBLD problem.

In Figures 10.10-10.13, these attribute values are represented by its first letter.

10.7 Experiments and Analysis

To test the effectiveness of the proposed RSBLD problem modeling algorithm (Algorithm 10.1) and solution algorithms (Algorithms 10.2 and 10.3), we

implemented these algorithms in Matlab 6.5. We then used classical data sets from the UCI database to test them in a set of experiments. The UCI database (http://www.ics.uci.edu/~mlearn/MLRepository.html) consists of many data sets that can be used by the decision systems and machine learning communities for the empirical analysis of algorithms.

For each data set we chose, we first selected half of the data set as the original objective rule set of the leader and the other half as the original objective rule set of the follower. We assumed no constraints, meaning that all objects consistent with the objective rule set were in the constraint region. We also supposed that the first half of the condition attributes were for the leader and the remainder for the follower. The importance degrees of the condition attributes are in descending order from the first condition attribute to the last condition attribute. The two experiments are processed on a computer with 2.33GHz CPU and 2G memory space. Here, we describe only these two experiments, as follows.

Experiment 10.1 Testing of Algorithm 10.1 with the data sets in the UCI database.

Step 1: Randomly choose 50% of the objects from the data set to be the original objective decision rule set of the leader, and the remaining 50% of the objects to be the original objective decision rule set of the follower;

Step 2: Apply Algorithm 10.1 to construct a rule-set-based bi-level decision model by using the chosen rule sets. Here, we use the decision matrices-based value reduction algorithm (Ziarko et al. 1996) in the RIDAS system (Wang et al. 2002) to reduce the size of the original rule sets.

Experiment 10.2 Testing of Algorithm 10.2 with the data sets in the UCI database.

Following Steps 1 and 2 in Experiment 10.1, we have

Step 3: Apply Algorithm 10.2 to get a solution from the generated rule-set-based bi-level decision model in Experiment 10.1.

The complexity of the two algorithms (Algorithms 10.1 and 10.2) is also tested by conducting these two experiments. As shown in Table 10.4, p_{OL} and p_{OF} are the numbers of objects in the original decision rules of the leader and the follower respectively; m_L and mF are the condition attribute numbers of the leader and the follower respectively; n_{OL} and n_{OF} are the numbers of the rules in the reduced objective decision rule set of the leader and the follower respectively; t_1 and t_2 are the processing times of Algorithms 10.1 and 10.2 respectively.

Table 10.4 Testing results of Algorithms 10.1 and 10.2

Data Sets	p_{OL}	p_{OF}	m_L	m_F	n_{OL}	n_{OF}	Algorithm 10.1	Algorithm 10.2
							t_1(sec.)	t_2 (sec.)
LENSES	12	12	2	3	6	3	<0.01	0.03
HAYES-ROTH	50	50	2	3	21	24	<0.01	0.09
AUTO-MPG	199	199	4	4	80	76	0.08	0.39
BUPA	172	172	3	3	159	126	0.06	3.10
PROCESSED_ CLEVELAND	151	151	6	7	115	127	0.28	5.20
BREAST-CANCER-WISCONSIN	349	349	5	5	47	47	0.51	0.63

From the results shown in Table 10.4 we find that

(1) The processing time of Algorithm 10.1 highly relates to the number of the rules in the original objective decision rule set and the condition attribute numbers of the leader and the follower respectively, expressed by the symbols p_{OL}, p_{OF}, m_L and m_F.

(2) The processing time of Algorithm 10.2 highly relates to the numbers of the rules in the reduced objective decision rule set and the condition attribute numbers of the leader and the follower respectively, expressed by n_{OL}, n_{OF}, m_L and m_F.

These are consistent with our complexity analysis results in Sections 10.4 and 10.5.

10.8 Summary

In the traditional bi-level decision-making models discussed in previous chapters, objectives and constraints are expressed by linear or nonlinear functions, and bi-level programming or genetic approaches can be effectively used to obtain solutions. However, some real-world bi-level decision problems cannot be easily formulated as linear or non-linear programs. This chapter uses rule sets to handle the issue. It presents how to use rule sets to model non-programming bi-level decision problems, and also develops two algorithms to solve rule-set-based bi-level decision problems.