Journal of Computer Science and Cybernetics, V.33, N.2 (2017), 131–142
DOI 10.15625/1813-9663/33/2/9281

SOME ALGORITHMS RELATED TO CONSISTENT DECISION TABLE

HOANG MINH QUANG[1], VU DUC THI[2], NGUYEN NGOC SAN[3]

[1] *Institute of Information Technology, Vietnam Academy of Science and Technology*
[2] *The Information Technology Institute (ITI), Vietnam National University, Hanoi*
[3] *Posts and Telecommunications Institute of Technology*
[1] *hoangquang@ioit.ac.vn*

Abstract. Rough set theory is a useful mathematical tool developed to deal with vagueness and uncertainty. As an important concept of rough set theory, an attribute reduct is a subset of attributes that are jointly sufficient and individually necessary for preserving a particular property of the given information table. Rough set theory is also the most popular for generating decision rules from decision table. In this paper, we propose an algorithm finding object reducts of consistent decsion table. On the other hand, we also show an algorithm to find some attribute reducts and the correctness of our algorithms is proof-theoretical. These algorithms of ours have polynomial time complexity. Our finding object reduct helps other algorithms of finding attribute reducts become more effective, especially as working with huge consistent decision table.

Keywords. Object reduct, attribute reduct, rough set theory, decision table, decision rules.

1. INTRODUCTION

Rough set theory, introduced by Zdzislaw Pawlak in the early 1980s [3], is a new methamatical tool to deal with vagueness and uncertainty. This approach seems to be of fundamental importance to artificial intelligence and cognitive sciences, especially in the areas of machine learning, knowledge acquisition, decision analysis, knowldege discovery from databases, expert systems, decision support systems, in ductive reasoning, and pattern recognition. The information [4] about the real world is given in the form of an information table (sometimes called a decision table). Thus, the information table represents input data, gathered from any domain, such as medicine, finance, or the military. Rows of a decision table are called objects (examples, entities). Properties, columns of a decision table, of objects are perceived through assigning values to some variables. We distinguish between two kinds of variables: *attributes* (sometimes called condition attributes) and *decisions* (sometimes called decision attributes). Usually a single decision is all that is required. For example, if the information table describes a hospital, the objects may be patients; the attributes, symptoms and tests; and the decisions, diseases. Each patient is characterized by the results of tests and symptoms and is classified by the physicians (experts) as being on some level of disease severity. If the information table describes an industrial process, the objects may represent samples of a process taken at some specific moments in time; attributes, the parameters of the process; and decisions, actions taken by the operators (experts).

The main concept of rough set theory [4] is an *indiscernibility relation*, normally associated with a set of attributes. Obviously, the indiscernibility relation is an equivalence relation. The redundant (or dispensable) attributes are defined due to the concept of indiscernibility relation. If a set of attributes and its superset define the same indiscernibility relation, then any attribute that belongs to the superset and not to the set is redundant. Such a set of attributes, with no redundant attribute, is called minimal (or independent). The set P of attributes is the *reduct* (or covering) of another set Q of attributes if P is minimal and the indiscernibility relations which defined by P and Q are the same.

In [5], the problems of generating minimal (relative) reducts and of generating minimal dependencies are NP-hard. In this paper, we propose a novel method to eliminate redundant objects of consistent decision table while the problem of finding some attribute reducts is preserved. Our methods run in polynomial time complexity. An object reduct of consistent decision table is a reductive table from the original one that satisfies both these two conditions: firstly that table has to preserve all attribute reducts of the original one; and secondly that table has to have minimal number of objects. For example, a consistent decision table which has one million objects, has ten attribute reducts totally; after applying the object reduct algorithm it still keeps ten attribute reducts but only has ten thousand objects and that number of objects is minimal so that if we reduce one more objects, we can not preseve the ten attribute reducts. It is clear that our algorithm is useful in reducing number of objects and size of data storage. Because computational time complexity of almost attribute reduct algorithms depends on number of objects, then obviously our algorithm helps these algorithms become more effective and efficient, especially for tables that contain huge number of objects.

2. BASIC CONCEPTS

Some basic concepts of relational database theory and of rough set theory can be found in [1, 2, 6] and [3, 4, 5], respectively. We consider a decision table in rough set theory as relational table in relational database theory because they include rows and columns and have some similar concepts about objects and attributes. Based on results in relational database theory [6], we can apply these results into decision table [7]. Some definitions and combinations that use in this paper are given as follows:

Pawlak denotes *information system* and *decision table* in [3]. The definitions about *relation* and *functional dependency* (FD) are in [2].

Definition 1. Let *decision table* $DS = (U, C \cup \{d\}, V, f)$, $U = \{u_1, ..., u_m\}$ be a *relation* over $C \cup \{d\}$. A decision table DS is *consistent* if and only if the *functional dependency* $C \to \{d\}$ is true; it means that for any $x, y \in U$ if $C(x) = C(y)$ then $d(x) = d(y)$. Conversely, DS is inconsistent.

Example 2. Given a consistent decision table $DS = (U, C \cup \{d\}, V, f)$
where $U = \{u_1, ..., u_6\}$ (or $\{1, 2, 3, 4, 5, 6\}$), $\{d\}$ is decision attribute "Flu" (denoted as $\{f\}$)
$C = \{Headache, Muscle\_pain, Temperature\}$ (denoted as $\{h, m, t\}$ or $\{hmt\}$),
$R = C \cup \{d\} = \{hmtf\}$.
$V_{Headache} = \{yes, no\}$,

$V_{Muscle_pain} = \{yes, no\}$,
$V_{Temperature} = \{normal, high, very\_high\}$,
$V_d = \{no, yes\}$, $V = V_{Headache} \cup V_{Muscle\_pain} \cup V_{Temperature} \cup V_d$,
and function $f : U \times C \cup \{d\} \rightarrow \bigcup\limits_{a \in C} V_a$ as Table 1

Table 1. Consistent decision table

STT	Headache	Muscle_pain	Temperature	Flu
1	yes	yes	normal	no
2	yes	yes	high	yes
3	yes	yes	very_high	yes
4	no	yes	normal	no
5	no	no	high	no
6	no	yes	very_high	yes

Definition 3. Every attribute subset $P \subseteq C \cup D$ determines an *indiscernibility relation* $IND(P) = \{(u, v) \in U \times U | \forall a \in P, f(u, a) = f(v, a)\}$,
$IND(P)$ determines a *partition* of U which is denoted by U/P.
Any element $[u]_P = \{v \in U | (u, v) \in IND(P)\}$ in U/P is called an *equivalent class.*

Example 4. (continue Example 2)

$$\begin{aligned}
U/h &= \{\{1, 2, 3\}, \{4, 5, 6\}\} \\
U/m &= \{\{1, 2, 3, 4, 6\}, \{5\}\} \\
U/t &= \{\{1, 4\}, \{2, 5\}, \{3, 6\}\} \\
U/f &= \{\{1, 4, 5\}, \{2, 3, 6\}\} \\
U/hm &= \{\{1, 2, 3\}, \{4, 6\}, \{5\}\}\} \\
U/ht &= \{\{1\}, \{2\}, \{3\}, \{4\}, \{5\}, \{6\}\} \\
U/mt &= \{\{1, 4\}, \{2\}, \{3, 6\}, \{5\}\} \\
U/hmt &= \{\{1\}, \{2\}, \{3\}, \{4\}, \{5\}, \{6\}\}.
\end{aligned}$$

Definition 5. Pawlak [3] define lower approximation, upper approximation and positive region based on equivalent class as follows:

- *B-upper approximation* of X is the set $\overline{B}X = \{u \in U | [u]_B \cap X \neq \emptyset\}$,
- *B-lower approximation* of X is the set $\underline{B}X = \{u \in U | [u]_B \subseteq X\}$ with $B \subseteq C$, $X \subseteq U$,
- *B-boundary* is the set $BN_B(X) = \overline{B}X \backslash \underline{B}X$,
- *B-positive region* of D is the set $POS_B(D) = \bigcup\limits_{X \in U/D} (\underline{B}X)$

Example 6. (continue Example 4) Since Definition 5 we obtain

$$\begin{aligned}
POS_h(\{f\}) &= \{\emptyset\}\\
POS_m(\{f\}) &= \{\{5\}\}\\
POS_t(\{f\}) &= \{\{1,4\},\{3,6\}\}\\
POS_{hm}(\{f\}) &= \{\{5\}\}\\
POS_{ht}(\{f\}) &= \{\{1\},\{2\},\{3\},\{4\},\{5\},\{6\}\}\\
POS_{mt}(\{f\}) &= \{\{1\},\{2\},\{3\},\{4\},\{5\},\{6\}\}.
\end{aligned}$$

Definition 7. A pair $s = \langle R, F\rangle$, where R is a set of attributes and F is a set of FDs on R, is called a *relation schema.* For any $A \subseteq R$, the set $A^+ = \{a : A \to \{a\} \in F^+\}$ is called the *closure* of A on s. It is clear that $A \to B \in F^+$ if and only if $B \subseteq A^+$. Similarly, $A_r^+ = \{a : A \to \{a\} \in F^+\}$ is called the closure of A on relation r.

Definition 8. Let r be a relation, $s = \langle R, F\rangle$ be a relation scheme and $A \subseteq R$. Then A is a *key* of r (a key of s) if $A \to R$ ($A \to R \in F^+$). A is a *minimal key* of r (s) if A is a key of r (s) and any proper subset of A is not key of r (s). The set of all minimal keys of r (s) is denoted by K_r (K_s). A family $K \subseteq P(R)$ is a *Sperner-system* on R if for any $A, B \in K$ implies $A \not\subset B$. It is clear that K_r (K_s) are Sperner-systems.

Definition 9. Let K be a Sperner-system over R as the set of all minimal keys of s. We defined the set of *antikeys* of K, denoted by K^{-1}, as follows

$$K^{-1} = \{A \subset R : (B \in K) \Rightarrow (B \not\subset A) \text{ and if } (A \subset C) \Rightarrow (\exists B \in K)(B \subseteq C)\}.$$

It is easy to see that K^{-1} is the set of subsets of R, which does not contain the element of K and which is maximal for this property. They are the maximal non-keys. Clearly, K^{-1} is also a Sperner-system.

Definition 10. Let r be a relation over R. Denote $E_r = \{E_{ij} : 1 \le i \le j \le |r|\}$, where $E_{ij} = \{a \in R : h_i(a) = h_j(a)\}$. Then E_r is called an *equality set* of r. For $A_r \in R, A_r^+ = \cap E_{ij}$, if there exists $E_{ij} \in E_r : A \subseteq E_{ij}$, otherwise $A_r^+ = R$.

Example 11. (continue Example 2) Let U be a relation over $C \cup \{d\}$, $1 \le i < j \le |U|$. For each pair of rows (i, j), we construct the sets E_{ij}. We have:

$E_{1,2} = \{hm\}$, $E_{1,3} = \{hm\}$. By doing the same thing with pairs (1, 4), ..., (1, 6), (2, 3), (2, 4), ..., (5, 6) we obtain the set E_r containing sets A_i as follows

$$\begin{aligned}
A_1 &= \{hm\} = E_{1,2} = E_{1,3} = E_{4,6}\\
A_2 &= \{mtf\} = E_{1,4} = E_{3,6}\\
A_3 &= \{f\} = E_{1,5}\\
A_4 &= \{m\} = E_{1,6} = E_{2,4} = E_{3,4}\\
A_5 &= \{hmf\} = E_{2,3}\\
A_6 &= \{t\} = E_{2,5}
\end{aligned}$$

$$A_7 = \{mf\} = E_{2,6}$$
$$A_8 = \{hf\} = E_{4,5}$$
$$A_9 = \{h\} = E_{5,6}$$

$$E_r = \{A_1, ..., A_9\} = E_r^U.$$

Definition 12. Let $r = \{h_1, ..., h_m\}$ be a relation over R, E_r is the equality set of r. Let

$$M_r = \{E_{ij} \in E_r : \forall E_{st} \in E_r : E_{ij} \subseteq E_{st}, E_{ij} \neq E_{st}\}$$

where $1 \leq i < j \leq m$, $1 \leq s < t \leq m$. M_r is called the *maximal equality system* of r.

$$M_d = \{A \in E_r : d \notin A, \nexists B \in E_r : d \notin B, A \subset B\}$$

is called the *maximal equality system* of r with respect to decision attribute d of consistent decision table DS.

Example 13. (continue Examples 2 and 11) We construct the set M_d^U being the maximal equality system of E_r that do not have decision attribute d of consistent decision table DS. It means that:

$$M_d = M_d^U = \{hm, t\} = \{B_1, B_2\}.$$

Definition 14. Let $s = \langle R, F\rangle$ be a relation scheme over R and $a \in R$. The set

$$K_a^s = \{A \subseteq R : A \to \{a\}, \nexists B : (B \to \{a\})(B \subset A)\}$$

is called a family of minimal sets of the attribute a over s. Similarly, the set

$$K_a^r = \{A \subseteq R : A \to \{a\}, \nexists B \subseteq R : (B \to \{a\})(B \subset A)\}$$

is called a family of minimal sets of the attribute a over r.

Definition 15. If K is a Sperner-system over R as the family of minimal sets of the attribute a over r (or s); in other words $K = K^r$ (or $K = K^s$), then $K^{-1} = \{K_a^r\}^{-1}$ (or $K^{-1} = \{K_a^s\}^{-1}$) is the family of maximal subsets of R which are not the family of minimal sets of the attribute a, defined as

$$\{K_a^r\}^{-1} = \{A \subseteq R : A \to \{a\} \notin R_r^+, A \subset B \Rightarrow B \to \{a\} \in F_r^+\},$$
$$\{K_a^s\}^{-1} = \{A \subseteq R : A \to \{a\} \notin R_r^+, A \subset B \Rightarrow B \to \{a\} \in F_r^+\}.$$

It is clear that $R \notin K_a^s$, $R \notin K_a^r$, $\{a\} \in K_a^s$, $\{a\} \in K_a^r$ and K_a^s, K_a^r are Sperner-systems over R.

Definition 16. Let $DS = (U, C \cup \{d\}, V, f)$ be a decision table. If $B \subseteq C$ satisfies
1) $POS_B(D) = POS_C(D)$
2) $\forall b \in B, POS_{B-\{b\}}(D) \neq POS_C(D)$
then B is called *reduct* of C.

If DS is a consistent decision table, B is an *attribute reduct* of C if B satifies $B \to \{d\}$ and $\forall B' \subset B$, $B' \not\to \{d\}$. Let $RED(C)$ be the set of all reducts of C. From Definition 16 and formula K_a^r in Definition 14 we have $RED(C) = K_d^r - \{d\}$ where K_d^r is the family of all minimal set of the attribute $\{d\}$ over $r = \langle U, C \cup \{d\}\rangle$.

Example 17. (continue Example 6) Based on Definition 16, we test $POS_B(\{f\})$ where $B \in \{\{h\}, \{m\}, \{t\}, \{hm\}, \{ht\}, \{mt\}\}$ then we have $B \in \{\{ht\}, \{mt\}\}$ satifies $POS_B(\{f\}) = POS_C(\{f\})$. Thus, $RED_U(C) = \{ht, mt\}$.

3. ALGORITHMS RELATED TO CONSISTENT DECISION TABLE

In this section, we construct two algorithms to find an object reduct and an attribute reduct run in polynomial time complexity. The first algorithm will preserve process that find some attribute reducts according to positive region reducts [3]. This is a significant result in data mining process specially with respect to large consistent decision table.

The objective of almost algorithms related to consistent decision table is to find all attribute reducts. While computational time complexity of these algorithms depends not only on number of attributes but also number of objects, there are many redundant objects making the process slower. Therefore, in this paper we introduce a method to remove these redundant objects, but still preserve all attribute reducts. This method bases on database theory and rough set theory. In rough set theory, we have the definition of $RED(C)$. In database theory, we have the definition of K_d^r. According to above definition, $RED(C) = K_d^r - \{d\}$, the formula means that set of all attribute reducts of consistent decision table equals set of all minimal keys of relation U over $C \cup \{d\}$ of $DS = (U, C \cup \{d\}, V, f)$ minus decision attribute $\{d\}$. On another hand, we know that if there exists a set of all minimal keys, then there also exists a set of anti keys, they define each other. We have Lemma 18, $M_d = (K_d^r)^{-1}$. If we gradually temporarily remove an object of consistent decision table so that M_d does not change, then $(K_d^r)^{-1}$ also does not change, thus set of all minimal keys does not change and $RED(C)$ does not change, then we conclude that this object is redundant, then we permanently remove that object; and vice versa. At the end, we get an object reduct which is a consistent decision table that already rejects all redundant objects.

Lemma 18. *Let $DS = (U, C \cup \{d\}, V, f)$ be a consistent decision table where $C = \{c_1, c_2, ..., c_n\}$, $U = \{u_1, u_2, ..., u_n\}$. Let us consider $r = \{u_1, u_2, ..., u_m\}$ on the attribute set $R = C \cup \{d\}$.*
We set $E_r = \{E_{ij} : 1 \leq i < j \leq m\}$ where $E_{ij} = \{a \in R : a(u_i) = a(u_j)\}$.
We set $M_d = \{A \in E_r : d \notin A, \nexists B \in E_r : d \notin B, A \subset B\}$.
Then we have $M_d = (K_d^r)^{-1}$ where K_d^r is called a family of minimal sets of the attribute $\{d\}$ over relation r.

The Lemma 18 is proved in [7].

Definition 19. An object reduct of consistent decision table $DS = (U, C \cup \{d\}, V, f)$ is a consistent decision table $DS' = (U', C \cup \{d\}, V, f)$, where $RED(C) = RED_U(C)$ and:
1) $U' \subseteq U$,
2) $RED_U(C) = RED_{U'}(C)$,
3) $RED_U(C) \neq RED_{U'-\{u\}}(C)$, $\forall u \in U'$.

Algorithm 1: Finding an object reduct over consistent decision table

```
Input  : $DS = (U, C \cup \{d\}, V, f)$
Output: $DS' = (U', C \cup \{d\}, V, f)$
Step 1: Compute $E_r = \{A_1, ..., A_t\}$;
Step 2: Compute $M_d^U = \{A \in E_r : d \notin A, \nexists B \in E_r : d \notin B, A \subset B\}$.;
Step 3: Set $T(0) = U = \{u_1, ..., u_m\}$;
Step 4: Set
    $T(i+1) = \begin{cases} T(i) - u_{i+1}, & \text{if } M_d^{T(i)-u_{i+1}} = M_d^U \\ T(i), & \text{otherwise} \end{cases}$
Then we set $U' = T(m)$.
```

Theorem 20. *$T(m)$ satisfies the three conditions 1), 2) and 3) in Definition 19.*

Proof. We prove the theorem by induction. At basis step $T(0) = U$, clearly, $U' = U$, $RED_{U'}(C) = RED_U(C)$ thus the two conditions 1), 2) are satisfied. At inductive step, assume that we have $T(i) = U(i)$ satisfies two conditions 1), 2) in Definition 19. We have to prove that $T(i+1) = U(i+1)$ satisfies the two conditions.

- In the first case: If $T(i+1) = T(i)$ then it is obvious that

  $$U(i+1) = U(i),\ RED_{U(i+1)}(C) = RED_{U(i)}(C) = RED(C)$$

  by induction hypothesis. Thus, $T(i+1)$ satisfies the two conditions 1), 2) in Definition 19.

- In the second case: If $T(i+1) = T(i) - \{u_{i+1}\}$ then $M_d^U = M_d^{U(i+1)}$.

  By Lemma 18, $M_d^U = \left(K_d^U\right)^{-1}$ where

  $$(U = \{u_1, ..., u_m\}) \Rightarrow M_d^{U(i+1)} = \left(K_d^{U(i+1)}\right)^{-1} \Rightarrow \left(K_d^U\right)^{-1} = \left(K_d^{U(i+1)}\right)^{-1}.$$

  By Definition 9 and 15 (K and K^1 are uniquely determined by one another), it can be seen that $\left(K_d^U\right) = \left(K_d^{U(i+1)}\right)$. From Definition 16 and the result of Definition 16, we have

  $$RED_U(C) = \left(K_d^U\right) - \{d\}$$

  and

  $$RED_{U(i+1)}(C) = \left(K_d^{U(i+1)}\right) - \{d\} \Rightarrow (ii1) RED_U(C) = RED_{U(i+1)}(C).$$

  From induction hypothesis, we have (ii2) $RED_U(C) = RED_{U(i)}(C)$. From (ii1), (ii2) we obtain $RED_U(C) = RED_{U(i)}(C) = RED_{U(i+1)}(C)$. Because $RED_U(C) = RED(C)$ is a Sperner-system (by definition K_d^U is a Sperner-system $\Rightarrow K_d^U - \{d\}$ is a Sperner-system), $RED_{U(i)}(C)$ and $RED_{U(i+1)}(C)$ are Sperner-systems. Finally, the two conditions in Definition 19 are satisfied at step $i+1$ as follows

1) $U(i+1) \subseteq U(i)$,

2) $RED_{U(i+1)}(C) = RED_{U(i)}(C) = ... = RED_U(C) = RED(C)$.

When $i+1 = m$ then Algorithm 1 stops. Now we need show that $U(m)$ satisfies the condition 3) in Definition 19 which means that $RED_{U(m)-u}(C) \neq RED_U(C)$, where $\forall u \in U(m)$. Assume that there exists $u = u_{i+1}$, $u \in U(m)$ such that $RED_{U(m)-u_{i+1}}(C) = RED_U(C)$ $(ii3)$. By Definition 16, $RED_{U(m)-u_{i+1}}(C) = K_d^{U(m)-u_{i+1}} - \{d\}$ and $RED_U(C) = K_d^U - \{d\}$, thus

$$(ii3) \Leftrightarrow K_d^{U(m)-u_{i+1}} - \{d\} = K_d^U - \{d\} \Leftrightarrow K_d^{U(m)-u_{i+1}} = K_d^U\,(ii4).$$

By Definition 9, 15 and Lemma 18 (K and K^{-1} are uniquely determined by one another), it means that

$$(ii4) \Leftrightarrow \left(K_d^{U(m)-u_{i+1}}\right)^{-1} = \left(K_d^U\right)^{-1} \Leftrightarrow M_d^{U(m)-u_{i+1}} = M_d^U\,(ii5).$$

By the above induction, if $M_d^{U(m)-u_{i+1}} = M_d^U$ then u_{i+1} will be removed, thus $u_{i+1} \notin U(m)$ which contradicts with hypothesis $u = u_{i+1} \in U(m)$. Hence, the condition 3) in Definition 19 is satisfied. The theorem is proved. ■

It is clear that the number of steps computing E_r by Definition 10 is less than $|U|^2$. The number of steps computing M_d is less than $|E_r|^2$ and $|E_r| \leq \frac{|U|(|U|-1)}{2}$. Thus, the worst-case time complexity of Algorithm 1 is not greater than $O(|U|^5)$. If we change order of universe set U, we can find another object reduct.

Example 21. (continue example 13) In step 3 and step 4 of Algorithm 1 we have:
$T(0) = U = \{1,2,3,4,5,6\}$, by using formula M_d in Definition 10 we compute

$$\begin{aligned}
E_r^{T(0)-\{1\}} &= \{A_1, A_2, A_4, A_5, A_6, A_7, A_8, A_9\}\\
M_d^{T(0)-\{1\}} &= \{hm, t\} = M_d^U \Rightarrow T(1) = T(0) - \{1\}\\
E_r^{T(1)-\{2\}} &= \{A_1, A_2, A_4, A_8, A_9\}\\
M_d^{T(1)-\{2\}} &= \{hm\} \neq M_d^U \Rightarrow T(2) = T(1)\\
E_r^{T(2)-\{3\}} &= \{A_1, A_4, A_6, A_7, A_8, A_9\}\\
M_d^{T(2)-\{3\}} &= \{hm, t\} = M_d^U \Rightarrow T(3) = T(2) - \{3\}\\
E_r^{T(3)-\{4\}} &= \{A_6, A_7, A_9\}\\
M_d^{T(3)-\{4\}} &= \{t, h\} \neq M_d^U \Rightarrow T(4) = T(3)\\
E_r^{T(4)-\{5\}} &= \{A_1, A_4, A_7\}\\
M_d^{T(4)-\{5\}} &= \{hm\} \neq M_d^U \Rightarrow T(5) = T(4)\\
E_r^{T(5)-\{6\}} &= \{A_4, A_6, A_8\}\\
M_d^{T(5)-\{6\}} &= \{m, t\} \neq M_d^U \Rightarrow T(6) = T(5).
\end{aligned}$$

Set $U' = T(6) = \{2,4,5,6\}$ then $DS' = (U', C \cup \{d\}, V, f)$ is the object reduct of the consistent decision table of $DS = (U, C \cup \{d\}, V, f)$ as Table 2.

Table 2. The object reduct of Table 1

STT	Headache	Muscle_pain	Temperature	Flu
2	yes	yes	high	yes
4	no	yes	normal	no
5	no	no	high	no
6	no	yes	very_high	yes

Example 22. (continue Example 21) By using Definition 3 we have

$$\begin{aligned} U'/h &= \{\{2\},\{4,5,6\}\} \\ U'/m &= \{\{2,4,6\},\{5\}\} \\ U'/t &= \{\{2,5\},\{4\},\{6\}\} \\ U'/f &= \{\{2,6\},\{4,5\}\} \\ U'/hm &= \{\{2\},\{4,6\},\{5\}\} \\ U'/ht &= \{\{2\},\{4\},\{5\},\{6\}\} \\ U'/tm &= \{\{2\},\{4\},\{5\},\{6\}\}. \end{aligned}$$

Example 23. (continue Example 22) By using Definition 5 we have

$$\begin{aligned} POS'_h(\{f\}) &= \{\{2\}\} \\ POS'_m(\{f\}) &= \{\{5\}\} \\ POS'_t(\{f\}) &= \{\{4\},\{6\}\} \\ POS'_{hm}(\{d\}) &= \{\{2\},\{5\}\} \\ POS'_{ht}(\{d\}) &= \{\{2\},\{4\},\{5\},\{6\}\} \\ POS'_{tm}(\{d\}) &= \{\{2\},\{4\},\{5\},\{6\}\}. \end{aligned}$$

Example 24. (continue Examples 17 and 23) Based on Definition 16, we have $RED_{U'}(C) = \{ht, tm\}$. By Definition 19, $RED_{U'}(C)$ is a Sperner-system and $RED_U(C) = RED_{U'}(C) = \{ht, tm\}$. Thus, the Algorithm 1 preserses $RED(C)$ of the consistent decision table DS (Table 1). Clearly, the consistent decision DS' is efficient for finding $RED(C)$ of consistent decision table according to rough set theory [3].

Algorithm 2: Finding the minimal key from a set of antikeys

```
Input  : Let $K$, $H$ be Sperner-systems and $C = \{c_1, ..., c_n\} \subseteq U$ such that
         $H^{-1} = K$ and $\exists B \in K : B \subseteq C$
Output: $D \in H$
Step 1: We set $A(0) = C$;
Step $i+1$: Set
    $A(i+1) = \begin{cases} A(i) - \{c_{i+1}\}, & \text{if } \forall B \in K : A(i) - \{c_{i+1}\} \not\subseteq B \\ A(i), & \text{otherwise} \end{cases}$
  Then we set $D = A(n)$.
```

Lemma 25. *[6] If K is a set of antikeys, then $A(n) \in H$.*

Algorithm 3: Finding an attribute reduct from a consistent decision table

```
Input  : $DS = (U, C \cup \{d\}, V, f)$
Output: $D \in RED(C)$
Step 1: Compute $E_r = \{A_1, ..., A_t\}$;
Step 2: Compute $M_d = \{A \in E_r : d \notin A, \nexists B \in E_r : d \notin B, A \subset B\}$;
Step 3: Set $H(0) = C = \{c_1, ..., c_n\}$;
Step 4: Set
    $H(i+1) = \begin{cases} H(i) - c_{i+1}, & \text{if } \nexists B \in M_d : H(i) - c_{i+1} \subseteq B \\ H(i), & \text{otherwise} \end{cases}$
  Then we set $D = H(n)$.
```

Theorem 26. $H(n) \in RED(C)$.

Proof. The Algorithm 3 is based on the Algorithm 2. By Lemma 18, $(K_d^r)^{-1} = M_d$. By Lemma 25, $H(n) \in K_d^r$ (1). By the result of Definition 16, $RED(C) = K_d^r - \{d\}$ (2). At step 3 of Algorithm 3 we set $C = \{c_1, ..., c_n\}$ then $d \notin C$. Thus, in Algorithm 3 $d \notin H(n)$ (3). From (1) and (3) we have $H(n) \in K_d^r - \{d\}$ (4). From (2) and (4) we obtain $H(n) \in RED(C)$. The theorem is proved. ∎

Similarly to the Algorithm 1, the time complexity of Algorithm 3 is not greater than $O(|C| \times |U|^4)$. If we change the order of the set C in step 3 we can get another attribute reduct of the consistent decision table DS.

Example 27. (continue Example 21) In step 1 of Algorithm 3 we have

$$E_{1,2} = \{m\}, E_{1,3} = \{t\}, E_{1,2} = \{mf\}, E_{2,3} = \{hf\}, E_{2,4} = \{hm\}, E_{3,4} = \{h\}.$$

In step 2 of Algorithm 3

$$E_r = \{m, t, mf, hf, hm, h\} \Rightarrow M_d = \{hm, t\} = \{B_1, B_2\}.$$

In step 3 and 4 of Algorithm 3 we have

$$\begin{aligned} C &= H(0) = \{hmt\} \\ temp &= H(0) - \{h\} = \{mt\} \not\subseteq (\forall B \in M_d) \Rightarrow H(1) = temp \\ temp &= H(1) - \{m\} = \{t\} \subseteq B_2 \in M_d \Rightarrow H(2) = H(1) \\ temp &= H(2) - \{t\} = \{m\} \subseteq B_1 \in M_d \Rightarrow H(3) = H(2). \end{aligned}$$

Set $H = H(3) = \{mt\}$, the Algorithm 3 and $H \in RED(C)$, we obtain table 3. By analogy with attributes [4], we can define elementary sets associated with the decision as subsets of the set of all objects with the same value of the decision. Such subsets will be called concepts. We observe that in terms of rough set theory, decision "Flu" depends on attributes "Muscle_pain" and "Temperature", since all elementary sets of indiscernibility relation associated with $\{mt\}$ are subsets of some concepts. The induced decision rules from Table 3 are equal to those from Table 1 with respect to reduct $\{mt\}$, they are

Table 3. An attribute reduct $\{mt\}$ of Table 2

STT	Muscle_pain	Temperature	Flu
1	yes	high	yes
2	yes	normal	no
3	no	high	no
4	yes	very_high	yes

$$\begin{aligned} normal(Temperature) &\rightarrow \neg(Flu), \\ \neg(Muscle\_pain) \wedge high(Temperature) &\rightarrow \neg(Flu), \\ (Muscle\_pain) \wedge high(Temperature) &\rightarrow (Flu), \\ very\_high(Temperature) &\rightarrow (Flu). \end{aligned}$$

4. CONCLUSION

In this paper, we proposed some novel algorithms to find an object reduct and an attribute reduct of consistent decision table and proved correctness of the algorithms. Based on this result, we can apply some machine learning methods such as learning decision rules or learning decision trees in an effective way. In addition, combination of attribute reduct and object reduct can help mine large consistent decision table in comparison with traditional methods.

REFERENCES

[1] J. Demetrovics and V. D. Thi, "Algorithms for generating an armstrong relation and inferring functional dependencies in the relational datamodel," *Computers & Mathematics with Applications*, vol. 26, no. 4, pp. 43–55, 1993.

[2] ——, "Keys, antikeys and prime attributes," in *Annales Univ. Sci. Budapest, Sect. Comp*, vol. 8, 1987, pp. 35–52.

[3] Z. Pawlak, "Rough sets," *International Journal of Computer & Information Sciences*, vol. 11, no. 5, pp. 341–356, 1982.

[4] Z. Pawlak, J. Grzymala-Busse, R. Slowinski, and W. Ziarko, "Rough sets," *Communications of the ACM*, vol. 38, no. 11, pp. 88–95, 1995.

[5] A. Skowron and C. Rauszer, "The discernibility matrices and functions in information systems," in *Intelligent Decision Support.* Springer, 1992, pp. 331–362.

[6] V. D. Thi, "The minimal keys and antikeys," *Acta Cybernetica*, vol. 7, no. 4, pp. 361–371, 1986.

[7] V. D. Thi and N. L. Giang, "A method to construct decision table from relation scheme," *Cybernetics and Information Technologies*, vol. 11, no. 3, pp. 32–41, 2011.

Received on March 06 - 2017
Revised on March 10 - 2017