Acta Cybernetica **23** (2018) 1039–1054.

An Efficient Method to Reduce the Size of Consistent Decision Tables

János Demetrovics[a] Hoang Minh Quang[b]
Vu Duc Thi[c] and Nguyen Viet Anh[b]

Abstract

Finding reductions from decision tables is one of the main objectives in information processing. Many studies focus on attribute reduct that reduces the number of columns in the decision table. The problem of finding all attribute reducts of consistent decision table is exponential in the number of attributes. In this paper, we aim at finding solutions for the problem of decision table reduction in polynomial time. More specifically, we deal with both the object reduct problem and the attribute reduct problem in consistent decision tables. We proved theoretically that our proposed methods for the two problems run in polynomial time. The proposed methods can be combined to significantly reduce the size of a consistent decision table both horizontally and vertically.

Keywords: attribute reduct, object reduct, consistent decision table, rough set theory, relational database theory

1 Introduction

Rough set theory was first represented by Pawlak in 1982. Since then, rough set theory [9] has found many interesting applications in areas such as knowledge acquisition, decision analysis, knowledge discovery from databases, expert systems, inductive reasoning and so on. The theory seems to be particularly important when applied for *information systems* (sometimes called data tables, decision tables, attribute-value systems, or condition-action tables) for knowledge representing, knowledge reduction, dependency reasoning and many other research problems.

Knowledge reduction [6, 17] is considered one of the most fundamental and important research tasks when working with information systems. The knowledge

[a]Computer and Automation Institute Hungarian Academy of Sciences, E-mail: `demetrovics@sztaki.mta.hu`
[b]Institute of Information Technology - Vietnam Academy of Science and Technology, E-mail: `{hoangquang,anhnv}@ioit.ac.vn`
[c]The Information Technology Institute (ITI) - Vietnam National University, Hanoi, E-mail: `vdthi@vnu.edu.vn`

DOI: 10.14232/actacyb.23.4.2018.4

reduction problem relates to the general concept of independence and knowledge core. It is about removing redundant attributes from the information system in such a way that the set of remaining attributes preserves only part of knowledge that is really useful. However, as proved in [11], the problems of generating the minimal reduct and minimal dependency are both NP-hard. Thus, the problem of finding all reducts as well as finding the minimal reduct using rough set theory may only be effective on small data sets. Knowledge reduction can be categorised into to finding reducts or relative reducts [7]. The concept of a reduct is built based on the idea of information systems and is easy to apply in applications. In contrast, the concept of a relative reduct is built based on decision systems. In fact, there are various definitions of reducts. The positive region reduct [8] is the most popular one, which is defined based on lower and upper approximation sets for defining an attribute reduct, for examples, Shannon's information entropy based reduct [8], classical rough set model based reduct[10], reduct based on discernibility matrix and discernibility function [16], reduct based on heuristic search [17].

Most of reducts implied that they are attribute reducts and not object reducts. [7] defines a reduct based on a distance measure with three evaluation metrics: *finding optimal*, *maximal exceeding*, *average exceeding*. Some authors introduce the concept of *variable precision rough set* [6] in which the concept of β lower distribution reduct and β upper distribution reduct are used. In these works, the equivalent definitions are given and the relationships among β lower, β upper distribution reducts and alternative types of knowledge reduction in inconsistent systems [16] are investigated. Moreover, with some special threshold, β lower and β upper distribution reducts are equivalent to the maximum distribution and the possible reduct, respectively. The authors use discernibility matrices associated with the β lower and upper distribution reducts from which the approaches to knowledge reduction in variable precision rough set can be obtained. [4] propose to use fuzzy rough set and information granulation for finding attribute reduct, and obtain different semantics in numerical attribute reduct and categorical attribute reduct. [4] derive several attribute significance measures based on the proposed fuzzy-rough model. From this, authors construct a greedy forward algorithm to find attribute reduct as feature subset selection. [4] also propose two strategies in attribute subset selection such as wrapper and filter for granular computing by using fuzzy information granules from numerical features and transforming numerical attribute into fuzzy linguistic variables. Some studies find attribute reducts according to the definition of concept lattices [1]. By combining rough set theory and formal concept analysis, these studies obtained reduction of a context by deleting rows (object oriented concept lattice) or columns (attribute oriented concept lattice) or both. Then, based on granular computing theory, they use the information granules or discernibility matrix and discernibility function to explore the attribute reduct. Therefore, the relationships between the attribute reducts of the concept lattices and the attribute reducts of the information system in rough set theory are found. Because of non-polynomial time complexity, most of algorithms mentioned above have to use a heuristic approach to search for reducts.

In this paper we propose two methods for dealing with the problem of finding all reduct attributes (or non-redundant attributes), columns of the consistent decision table are involved at least in one of attribute reducts, and the problem of finding an object reduct that removes redundant objects, rows of the consistent decision table are no effect to finding set of all attribute reducts over decision attributes. The proposed methods are proved theoretically having polynomial complexity in running time. Moreover, by combining the two methods, we can obtain a consistent decision table that its size is reduced in both horizontal and vertical dimensions. Our ideas are based on some basis concepts of relational database theory [2, 3, 12] and rough set theory [8, 11]. In relational dabase theory, the basis important concept is the concept of minimal keys and antikeys. They form the so-called Sperner-systems. We consider decision tables that can be regarded as relation tables in relational database theory. Decision tables and relation tables are tables containing rows and columns. A decision table has an attribute set that can be divided into the condition set and the decision set. It is obvious that there is a correspondence between function dependencies in a relation and dependencies in a decision table. By applying methods of finding keys and antikeys, we construct keys and antikeys for a consistent decision table. Some results in relation about keys and antikeys have polynomial time complexity. By using these results of minimal keys and antikeys and based on the maximal equality set definition, we build an algorithm for finding a reduct of consistent decision table in polynomial time. To the best of our knowledge, this is the first time some interesting results in the relational database theory are directly applied in efficiently finding reducts from decision tables.

The rest of the paper is organized as following. In Section 2, we give necessary notions and definitions regarding the relational database theory and rough set theory which will be later used in the paper. In Section 3, we describe our proposed methods for object reduct and attribute reduct from a consistent decision table. In Section 4, we give a case study to illustrate our proposed methods. We summarize the paper in Section 5.

2 Preliminaries

In this section we show some basis concepts of relational database theory [2, 3, 12] and rough set theory [8, 9, 11, 13].

2.1 Relational database theory

Definition 1. *Let $R = \{a_1, ..., a_n\}$ be a finite set of attributes and let $D(a_i)$ be the set of all possible values of attribute a_i, the* relation *r over R is the set of tuples $\{h_1, ..., h_m\}$ where $h_j : R \rightarrow \bigcup_{a_i \in R} D(a_i), 1 \leq j \leq m$, is a function that $h_j(a_i) \in D(a_i)$.*

Definition 2. *Let $r = \{h_1, ..., h_m\}$ be a relation over $R = \{a_1, ..., a_n\}$. Any pair of attribute sets $A, B \subseteq R$ is called* functional dependency *(FD) over R, and it is* *denoted by $A \to B$ if and only if*

$$(\forall h_i, h_j \in r)((\forall a \in A)(h_i(a) = h_j(a)) \Rightarrow (\forall b \in B)(h_i(b) = h_j(b))).$$

Definition 3. *The set $F_r = \{(A, B) : A, B \subset R, A \to B\}$ is called a* full family *of functional dependencies in r. Let $P(R)$ be the power set of attribute set R. A family $F \subseteq P(R) \times P(R)$ is called a* f-family *over R if and only if for all subsets of attributes $A, B, C, D \subseteq R$ the following properties hold:*
1) $(A, A) \in F$.
2) $(A, B) \in F, (B, C) \in F \Rightarrow (A, C) \in F$.
3) $(A, B) \in F, A \subseteq C, D \subseteq B \Rightarrow (C, D) \in F$.
4) $(A, B) \in F, (C, D) \in F \Rightarrow (A \cup C, B \cup D) \in F$.

Clearly, F_r is an f-family over R. It is also known that if F is an f-family over R, then there is a relation r such that $F_r = F$. Let us denote by F^+ the set of all FDs, which can be derived from F by using rules 1)-4).

Definition 4. *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 5. *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 $A \not\subseteq B$. It is clear that K_r (K_s) are Sperner-systems.*

Definition 6. *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:*

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

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 7. *Let r be a relation over R. Denote $E_r = \{E_{ij} : 1 \leq i \leq j \leq |r|\}$, 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 $A_r^+ = R$.

Definition 8. *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.*

Definition 9. *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 10. *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:*

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

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.

2.2 Rough set theory

Definition 11. *An* information system *S is an order quadruple $S = (U, A, V, f)$ where U is a finite set of* objects, *called the* universe*; A is a finite set of* attributes*; $V = \bigcup_{a \in A} V_a$ and V_a is the* domain *of attribute a; $f : U \times A \to V$ is a total function, such that $f(x, a) \in V_a$ for every $a \in A$ and $x \in U$ called the* information function. *The function $f_x : A \to V$ such that $f_x(a) = f(x, a)$ for every $a \in A$ and $x \in U$ will be called* information about *x in S. We denote $a(x) = f_x(a)$. If $B = \{b_1, b_2, ..., b_k\} \subseteq A$ is subset of attributes, then the set of $b_i(x)$ is denoted as $B(x)$. Therefore, if x, y are two objects in U, then $B(x) = B(y)$ if and only if $b_i(x) = b_i(y), \forall i = 1, ..., k$.*

Definition 12. Decision table *is an information system $S = (U, A, V, f)$, where $A = C \cup D$ and $C \cap D = \emptyset$. Without loss of generality, suppose that D consists of only one decision attribute d. Therefore, from this time we consider the decision table $DS = (U, C \cup \{d\}, V, f)$, where $\{d\} \notin C$.*

Definition 13. *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.*

Definition 14. *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.

- 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* $POS_B(D) = \bigcup\limits_{X \in U/D} (\underline{B}X)$

Definition 15. *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 attribute 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 15 and formula* K_a^r *in definition 9 we have* $RED(C) = K_d^r - \{d\}$ *where* K_d^r *is the family of all minimal set of the attribute* $\{d\}$ *over* $r = \langle U, C \cup \{d\}\rangle$

3 Object reduct and attribute reduct

In this section, we construct some methods to finding all non-redundant attributes, an object reduct and an attribute reduct and all of them have complexity in polynomial. A lot of existing approaches try to find all attribute reducts first, and then select the most *suitable* one. Unfortunately, the problem of finding all attribute reducts of consistent decision table is exponential in the number of attributes [5]. Because of exponential computational time, many research using heuristic methods to find an attribute reduct [1, 4, 6, 7, 10, 16, 17]. The method of finding an attribute we propose is not a heuristic algorithm. First, we eliminate all redundant attributes, that are not involved in any attribute reduct of consistent decision table, by using the algorithm 1. After that, we build two algorithms that one find an object reduct, the algorithm 2, and another find an attribute reduct, the algorithm 4. The combination of these methods generate a consistent decision table that is reduced in size in both vertical and horizontal dimensions. These results will reduce cost of storage data, specially for massive dataset, and the object reduct completely preserve information for finding all attribute reducts.

Lemma 1. *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 a family of minimal sets of the attribute* $\{d\}$ *over relation r.*

The lemma 1 is proved in [13].

Theorem 1. *[2] Let K be a Sperner-system over Ω. Then*

$$\bigcup_{A\in K} A = \Omega - \bigcap_{B\in K^{-1}} B$$

Definition 16. *Given a consistent decision table $DS = (U, C \cup \{d\}, V, f)$, let DS be relation $U = \{u_1, ..., u_m\}$ over attribute set $R = C \cup \{d\}$, from definition 15 we have $RED(C) = K_d^r - \{d\}$, if denote $REAT(C)$ a set of all non-redundant attributes or reduct attributes of C then:*

$$REAT(C) = \bigcup_{A\in RED(C)} A = \left(\bigcup_{A\in K_d^r} A\right) - \{d\}$$

Algorithm 1 Finding the set of all reduct attributes of C

<u>**Function**</u> REAT($DS = (U, C \cup \{d\}, V, f)$, $POS_C(\{d\}) = U$, $C = \{c_1, ..., c_n\}$, $U = \{u_1, ..., u_m\}$)

```
1: Consider the relation $r = \{u_1, ..., u_m\}$ over the attribute set $R = C \cup \{d\}$.
2: Step 1: Compute $E_r = \{A_1, ..., A_t\}$
3: Step 2: Compute $M_d = \{A \in E_r : d \notin A, \nexists B \in E_r : d \notin B, A \subset B\}$.
4: Step 3: Construct $N = R - \bigcap_{B\in M_d} B$
5: Step 4: Set $REAT(C) = N - \{d\}$
```

Theorem 2. *$REAT(C)$ is set of all reduct attributes of C.*

Proof. The theorem 2 is proved in [14]. It is restated as follows:
By lemma 1 $M_d = (K_d^r)^{-1}$. At step 3, combine with definition 6, $(K_d^r)^{-1}$ and (K_d^r) are Sperner-systems, with theorem 1 we have:

$$N = R - \bigcap_{B\in M_d} B = R - \bigcap_{B\in \left(K_d^r\right)^{-1}} B = \bigcup_{A\in K_d^r} A$$

At step 4 we have:

$$REAT(C) = N - \{d\} = \left(\bigcup_{A\in K_d^r} A\right) - \{d\} = \bigcup_{A\in RED(C)} A$$

Thus, by definition 16, $REAT(C)$ is the set of all reduct attributes of C, $REAT(C)$ is the set of all non-redundant attributes of C. □

It can be seen that the number of computational steps of E_r is not greater than $|U|^2$ and the number of computational steps of M_d is not greater than $|E_r|^2$. Thus, the worst case time computational complexity of the algorithm is $O(|U|^4 + |C \cup \{d\}|)$ which is polynomial by number of rows and columns of decision table DS.

Definition 17. *An object reduct of a 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 2 Finding an object reduct over consistent decision table

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

Theorem 3. *$T(m)$ satisfies the two conditions 1), 2) and 3) in definition 17.*

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 17. 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 17.

- In the second case: If $T(i+1) = T(i) - \{u_{i+1}\}$ then $M_d^U = M_d^{U(i+1)}$. By lemma 1, $M_d^U = \left(K_d^U\right)^{-1}$ 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 6 and 10 (K and K^1 are uniquely determined by one another), it can see that $\left(K_d^U\right) = \left(K_d^{U(i+1)}\right)$. From definition 15 and the result of definition 15, 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 and $\Rightarrow K_d^U - \{d\}$ is a Sperner-system), $RED_{U(i)}(C)$ and $RED_{U(i+1)}(C)$ are Sperner-systems. Finally, the two conditions in definition 17 are satisfied at step $i+1$ as follow:

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 2 stops. Now we need to show that $U(m)$ satisfies the condition 3) in definition 17 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 15, $RED_{U(m)-u_{i+1}}(C) = K_d^{U(m)-u_{i+1}} - \{d\}$ 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 6, 10 and lemma 1 (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 above proving induction, if $M_d^{U(m)-u_{i+1}} = M_d^U$ then u_{i+1} will be removed, thus $u_{i+1} \notin U(m)$ contradicts with hypothesis $u = u_{i+1} \in U(m)$. Hence, the condition 3) in definition 17 is satisfied. The theorem is proved. $\square$

It is clear that the number of steps computing E_r by definition 7 is less than $|U|^2$. The number of steps computing M_d is less than $|E_r|^2$ and $|E_r| \leq \dfrac{|U|(|U|-1)}{2}$. Thus, the worst-case time complexity of algorithm 2 is not greater than $O(|U|^5)$. If we change the order of the universe set U, we can find another object reduct.

Algorithm 3 Finding the minimal key from a set of antikeys

Function MinimalKey(Let K, H be Sperner-systems and $C = \{c_1, ..., c_n\} \subseteq U$ such that $H^{-1} = K$ and $\exists B \in K : B \subseteq C$)

```
1: Step 1: We set $A(0) = C$
2: Step $i+1$: Set
       $$A(i+1) = \begin{cases} A(i) - \{c_{i+1}\}, & \text{if } \forall B \in K : A(i) - \{c_{i+1}\} \not\subseteq B \\ A(i), & \text{otherwise} \end{cases}$$
3: Then we set $D = A(n)$.
```

Lemma 2. *[12] If K is a set of antikeys, then $A(n) \in H$.*

Algorithm 4 Finding an attribute reduct from a consistent decision table

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

Theorem 4. *$H(n) \in RED(C)$, where $H(n)$ in algorithm 4.*

Proof. The algorithm 4 is based on the algorithm 3. By lemma 1, $(K_d^r)^{-1} = M_d$. By lemma 2, $H(n) \in K_d^r$ (1). By the result of definition 15, $RED(C) = K_d^r - \{d\}$ (2). At step 3 of algorithm 4 we set $C = \{c_1, ..., c_n\}$ then $d \notin C$. Thus, in algorithm 4 we have $d \notin H(n)$ (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. □

Similar to the algorithm 2, the time complexity of algorithm 4 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. Thus, the problem of finding all attribute reducts is exponential time complexity in the number of attributes [5].

In order to reduce the size of consistent decision table in both vertical and horizontal dimensions, the first step in our method is to use the algorithm 1 to determine $REAT(C)$ and then use the algorithm 2 to get an object reduct. So $REAT(C)$ is the set of all reduct attributes, we obtain reduction in the horizontal dimension, reducing number of columns, of the consistent decision table. After that the object reduct is reduction in the vertical dimension, reducing number of rows, of the consistent decision table. It is easy to prove that our method run in polynomial time because the algorithm 1 and 2 are polynomial time complexity. It is obvious that the consistent decision table that is reduced both vertically and horizontally occupies much less capacity of storage than the original, but it preserves all necessary information for finding all attribute reducts. In addition, our method applies the algorithm 4 to find an attribute reduct that run in polynomial time and the attribute reduct help more and more efficiently and effectively in learing process.

4 A case study

Example 1. Given a consistent decision table $DS = (U, C \cup \{d\}, V, f)$ where $U = \{u_1, ..., u_{14}\}$ ($\{1, ..., 14\}$),

d is decision attribute "Play Golf",
$C = \{Outlook, Grass, Temperature, Humidity, Windy, NumberHoles\}$
$(\{o, g, t, h, w, n\}$ or $\{ogthwn\})$,
$R = C \cup \{d\} = \{ogthwnd\}$.
$V_{Outlook} = \{Sunny, OverCast, Rain\}$,
$V_{Temperature} = \{High, Middle, Low\}$,
$V_{Humidity} = \{High, Middle\}$,
$V_{Grass} = \{Wet, Dry\}$,
$V_{Windy} = \{Weak, Strong\}$,
$V_{NumberHoles} = \{20, 10\}$,
$V_d = \{No, Yes\}$, $V = V_{Outlook} \cup V_{Grass} \cup V_{Temperature} \cup V_{Humidity} \cup V_{Windy} \cup V_{NumberHoles} \cup V_d$,
and function $f : U \times C \cup \{d\} \rightarrow \bigcup_{a \in C} V_a$ as table 1.

Table 1: A consistent decision table

No.	O	G	T	H	W	N	d
1	Sunny	Wet	High	High	Weak	10	No
2	Sunny	Dry	High	High	Strong	20	No
3	Overcast	Wet	High	High	Weak	10	Yes
4	Rain	Dry	Middle	High	Weak	10	Yes
5	Rain	Wet	Low	Middle	Weak	20	Yes
6	Rain	Wet	Low	Middle	Strong	20	No
7	Overcast	Dry	Middle	Middle	Strong	20	Yes
8	Sunny	Wet	Low	High	Weak	10	No
9	Sunny	Wet	Middle	Middle	Weak	10	Yes
10	Rain	Dry	Middle	Middle	Weak	20	Yes
11	Sunny	Dry	Middle	Middle	Strong	20	Yes
12	Overcast	Dry	Middle	High	Strong	10	Yes
13	Overcast	Dry	High	Middle	Weak	20	Yes
14	Rain	Dry	Middle	High	Strong	10	No

Example 2. (continue the example 1) By applying algorithm 1 we have that E_r contains all $E_{i,j}$ as follows:

$E_{1,2} = othd$, $E_{1,3} = gthwn$, $E_{1,4} = hwn$, $E_{1,5} = gw$, $E_{1,6} = gd$, $E_{1,8} = oghwnd$, $E_{1,9} = ogwn$, $E_{1,10} = w$, $E_{1,11} = o$, $E_{1,12} = hn$, $E_{1,13} = tw$, $E_{1,14} = hnd$, $E_{2,3} = th$, $E_{2,4} = gh$, $E_{2,5} = n$, $E_{2,6} = wnd$, $E_{2,7} = gwn$, $E_{2,8} = ohd$, $E_{2,10} = gn$, $E_{2,12} = ghw$, $E_{2,13} = gtn$, $E_{2,14} = ghwd$, $E_{3,4} = hwnd$, $E_{3,5} = gwd$, $E_{3,6} = g$, $E_{3,7} = od$, $E_{3,8} = ghwn$, $E_{3,9} = gwnd$, $E_{3,10} = wd$, $E_{3,11} = d$, $E_{3,12} = ohnd$, $E_{3,13} = otwd$, $E_{4,5} = owd$, $E_{4,7} = gtd$, $E_{4,9} = twnd$, $E_{4,10} = ogtwd$, $E_{4,12} = gthnd$, $E_{4,14} = ogthn$, $E_{5,8} = gtw$, $E_{5,10} = ohwnd$, $E_{6,10} = ohn$, $E_{7,9} = thd$, $E_{7,11} = gthwnd$, $E_{7,13} = oghnd$, $E_{9,10} = thwd$, $E_{9,12} = tnd$, $E_{9,13} = hwd$, $E_{9,14} = tn$,

$E_{10,13} = ghwnd$, $E_{10,14} = ogt$, $E_{11,12} = gtwd$, $E_{11,13} = ghnd$, $E_{12,13} = ogd$

$M_d = \{gthwn, ogthn, ogwn\} = \{M_1, M_2, M_3\}$

$G = \bigcap_{M \in M_d} = M_1 \cap M_2 \cap M_3 = \{gthwn\} \cap \{ogthn\} \cap \{ogwn\} = \{gn\}$

$REAT(C) = N - \{d\} = R - G - \{d\} = \{ogthwnd\} - \{gn\} - \{d\} = \{othw\}$

Thus, the two attributes "Grass" and "NumberHoles" are redundant, by removing these two attributes, we obtain non-redundant attributes consistent decision table $NOREDS = (U, \{o, t, h, w\} \cup \{d\}, V, f)$ of the consistent decision table 1. From this example, we consider $C = \{o, t, h, w\}$ instead of $C = \{o, g, t, h, w, n\}$.

Example 3. (continue example 2) By defintion 14 we find $RED(C)$.
$POS_o(\{d\}) = \{3, 7, 12.13\}$,
$POS_t(\{d\}) = \emptyset, POS_h(\{d\}) = \emptyset, POS_w(\{d\}) = \emptyset$,
$POS_{ot}(\{d\}) = \{1, 2, 3, 7, 8, 9, 11, 12, 13\}$,
$POS_{oh}(\{d\}) = \{1, 2, 3, 7, 8, 9, 11, 12, 13\}$,
$POS_{ow}(\{d\}) = \{3, 4, 5, 6, 7, 12, 13, 14\}$,
$POS_{th}(\{d\}) = \{7, 8, 9, 10, 11, 13\}$,
$POS_{tw}(\{d\}) = \{2, 4, 6, 9, 10\}$,
$POS_{hw}(\{d\}) = \{5, 9, 10, 13\}$,
$POS_{oth}(\{d\}) = \{1, 2, 3, 7, 8, 9, 10, 11, 12, 13\}$,
$POS_{otw}(\{d\}) = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14\}$,
$POS_{ohw}(\{d\}) = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14\}$,
$POS_{thw}(\{d\}) = \{2, 4, 5, 6, 7, 8, 9, 10, 11, 13\}$.
We see that $POS_{\{otw\}}(\{d\}) = POS_{\{ohw\}}(\{d\}) = POS_C(\{d\})$. By definition 15, the set of all reducts of $C = \{othw\}$ over the consistent decision table $DS = \{U, C \cup \{d\}, V, f\}$ in example 2 is $RED(C) = \{otw, ohw\}$

Example 4. (continue example 2) In step 1 in algorithm 2, from example 2 and definition 7, for each pair of rows (i, j), we construct the sets E_{ij}. We have:

$E_{1,2} = \{othd\}$, $E_{1,3} = \{thw\}$. By doing the same thing with pairs (1, 4), ..., (1, 14), (2, 3), (2, 4), ..., (13, 14) we obtain the set E_r containing sets A_i as follows:

$A_1 = \{othd\}$, $A_2 = \{thw\}$, $A_3 = \{hw\}$, $A_4 = \{w\}$, $A_5 = \{d\}$, $A_6 = \{ohwd\}$, $A_7 = \{ow\}$, $A_8 = \{o\}$, $A_9 = \{h\}$, $A_{10} = \{tw\}$, $A_{11} = \{hd\}$, $A_{12} = \{th\}$, $A_{13} = \{wd\}$, $A_{14} = \{ohd\}$, $A_{15} = \{t\}$, $A_{16} = \{hwd\}$, $A_{17} = \{od\}$, $A_{18} = \{otwd\}$, $A_{19} = \{owd\}$, $A_{20} = \{td\}$, $A_{21} = \{twd\}$, $A_{22} = \{thd\}$, $A_{23} = \{oth\}$, $A_{24} = \{oh\}$, $A_{25} = \{thwd\}$, $A_{26} = \{ot\}$

$$E_r = \{A_1, ..., A_{26}\} = E_r^U$$

Example 5. (continue example 4) In step 2 of algorithm 2, we construct the set M_d^U being the maximal equality system of E_r that do not have decision attribute d. We obtain:

$$M_d = M_d^U = \{thw, oth, ow\} = \{B_1, B_2, B_3\}$$

Example 6. (continue example 5) In step 3 and step 4 of algorithm 2 we have: $T(0) = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14\}$, by using definition 7 and formula at

step 2 in algorithm 2 we compute:
$M_d^{T(0)-\{1\}} = \{thw, oth, ow\} = M_d^U \Rightarrow T(1) = T(0) - \{1\}$
$M_d^{T(1)-\{2\}} = \{thw, oth, ow\} = M_d^U \Rightarrow T(2) = T(1) - \{2\}$
$M_d^{T(2)-\{3\}} = \{thw, oth, ow\} = M_d^U \Rightarrow T(3) = T(2) - \{3\}$
$M_d^{T(3)-\{4\}} = \{thw, oth, ow\} = M_d^U \Rightarrow T(4) = T(3) - \{4\}$
$M_d^{T(4)-\{5\}} = \{thw, ow, oh, ot\} \neq M_d^U \Rightarrow T(5) = T(4)$
$M_d^{T(5)-\{6\}} = \{thw, ow, ot\} \neq M_d^U \Rightarrow T(6) = T(5)$
$M_d^{T(6)-\{7\}} = \{thw, oth, ow\} = M_d^U \Rightarrow T(7) = T(6) - \{7\}$
$M_d^{T(7)-\{8\}} = \{thw, oth\} \neq M_d^U \Rightarrow T(8) = T(7)$
$M_d^{T(8)-\{9\}} = \{thw, oth\} \neq M_d^U \Rightarrow T(9) = T(8)$
$M_d^{T(9)-\{10\}} = \{thw, oth, ow\} = M_d^U \Rightarrow T(10) = T(9) - \{10\}$
$M_d^{T(10)-\{11\}} = \{thw, oth, ow\} = M_d^U \Rightarrow T(11) = T(10) - \{11\}$
$M_d^{T(11)-\{12\}} = \{oth, ow, tw\} \neq M_d^U \Rightarrow T(12) = T(11)$
$M_d^{T(12)-\{13\}} = \{thw, oth, ow\} = M_d^U \Rightarrow T(13) = T(12) - \{13\}$
$M_d^{T(13)-\{14\}} = \{oth, ow, tw\} \neq M_d^U \Rightarrow T(14) = T(13)$
Set $U' = T(14) = \{5, 6, 8, 9, 12, 14\}$ then $OBREDS = (\{5, 6, 8, 9, 12, 14\}, C \cup \{d\}, V, f)$ is the object reduct of the consistent decision table $NOREDS$

Example 7. (continue example 2, 3 and 6) Based on the object reduct of the consistent decision table $OBREDS = DS' = (U', C \cup \{d\}, V, f)$ from example 6, we use definition 14 to find $RED_{U'}(C)$.
$POS'_o(\{d\}) = \{12\}$
$POS'_t(\{d\}) = \emptyset, POS'_h(\{d\}) = \emptyset, POS'_w(\{d\}) = \emptyset$
$POS'_{ot}(\{d\}) = \{8, 9, 12, 14\}$
$POS'_{oh}(\{d\}) = \{8, 9, 12, 14\}$
$POS'_{ow}(\{d\}) = \{5, 6, 12, 14\}$
$POS'_{th}(\{d\}) = \{8, 9\}$
$POS'_{tw}(\{d\}) = \{6, 9\}$
$POS'_{hw}(\{d\}) = \{5, 6, 8, 9, 12, 14\}$
$POS'_{oth}(\{d\}) = \{8, 9, 12, 14\}$
$POS'_{otw}(\{d\}) = \{5, 6, 8, 9, 12, 14\}$
$POS'_{ohw}(\{d\}) = \{5, 6, 8, 9, 12, 14\}$
$POS'_{thw}(\{d\}) = \{5, 6, 8, 9\}$
We see that $POS'_{\{hw\}}(\{d\}) = POS'_{\{otw\}}(\{d\}) = POS'_{\{ohw\}}(\{d\}) = U'$. Let the set $P = \{POS'_B(\{d\})\} = \{hw, otw, ohw\}$. Because U' is an object reduct of U according to the definition of the maximal equality system of attribute $\{d\}$, the set of all reducts of C is a Sperner-system.
Thus, $RED_{U'}(C) = \{B \in P, \nexists A \in P, A \subset B\}$. In P, $\{hw\} \subset \{ohw\}$, we remove $\{hw\}$ and P becomes a Sperner-system. It is obvious that $RED_{U'}(C) = P - \{hw\} = \{otw, ohw\} = RED_U(C)$. Clearly, $RED_{U'}(C)$ generated by object reduct in the consistent decision table DS' equals to $RED_U(C)$ generated by the original consistent decision table DS.

Example 8. (continue example 5) By applying algorithm 4 we find only one attribute reduct on consistent decision table from example 2, $C = \{othw\}$
$temp = H(0) - \{o\} = \{thw\} = B_1 \in M_d \Rightarrow H(1) = H(0)$
$temp = H(1) - \{t\} = \{ohw\} \not\subseteq \{B \in M_d\} \Rightarrow H(2) = temp$
$temp = H(2) - \{h\} = \{ow\} = B_3 \in M_d \Rightarrow H(3) = H(2)$
$temp = H(3) - \{w\} = \{oh\} \subseteq B_2 \in M_d \Rightarrow H(4) = H(3)$
Set $H = H(4) = \{ohw\}$ and algorithm 4 stops and $H \in RED(C)$. We have table $ATREDS = (U, \{o, h, w\} \cup \{d\}, V, f)$.

Example 9. Combining algorithm 1 and 2 on examples 2 and 6, we obtain the consistent decision table which are reduced in both vertical and horizontal dimensions. In addition to attribute reduct that is obtained by algorithm 4 on example 8, the relation $r_1 = \{5, 6, 8, 9, 12, 14\}$ over $\{ohw\}$ is a consistent decision table as table 2 for learning process.

Table 2: Table r_1 is combination of NOREDS, OBREDS and ATREDS

No.	Outlook	Humidity	Windy	d
5	Rain	Middle	Weak	Yes
6	Rain	Middle	Strong	No
8	Sunny	High	Weak	No
9	Sunny	Middle	Weak	Yes
12	Overcast	High	Strong	Yes
14	Rain	High	Strong	No

A decision tree that is generated from the consistent decision table r_1 (table 2) as Fig 1. The decision tree (Fig 1) is also one of the decision trees that are generated from the consistent decision table 1 by algorithm ID3 (or C4.5).

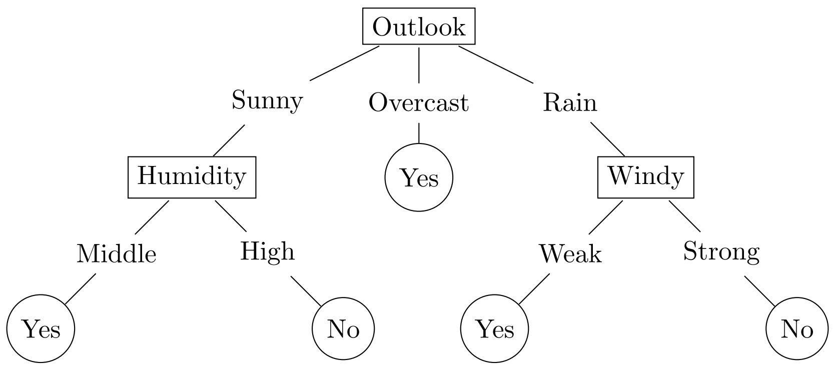

Figure 1: The decision tree generated from combination reducts table 2

5 Conclusion

In this paper, we have proposed some novel methods to reduce the consistent decision tables in both horizontal and vertical dimensions. Our ideas are based on some results from relational database theory and rough set theory. The algorithm of finding all reduct attributes and the algorithm of finding an object reduct run in polynomial time complexity. The algorithm of finding attribute reducts may be either polynomial time complexity in the case of finding only one attribute reduct or exponential time complexity [5] in the case of finding all attribute reducts of consistent decision table. The learning decision trees [15] that are generated from the reduced decision table are obtained from those generated from the original decision table. Thus, our methods can help to facilitate the learning process from larger decision tables compared with existing methods.

References

[1] Cornejo, Ma Eugenia, Medina, Jesús, and Ramírez-Poussa, Eloisa. Attribute reduction in multi-adjoint concept lattices. *Information Sciences*, 294:41–56, 2015.

[2] Demetrovics, János and Thi, Vu Duc. Keys, antikeys and prime attributes. In *Annales Univ. Sci. Budapest, Sect. Comp*, volume 8, pages 35–52, 1987.

[3] Demetrovics, János and Thi, Vu Duc. Algorithms for generating an armstrong relation and inferring functional dependencies in the relational datamodel. *Computers & Mathematics with Applications*, 26(4):43–55, 1993.

[4] Hu, Qinghua, Xie, Zongxia, and Yu, Daren. Hybrid attribute reduction based on a novel fuzzy-rough model and information granulation. *Pattern recognition*, 40(12):3509–3521, 2007.

[5] Janos, Demetrovics, Thi, Vu Duc, and Giang, Nguyen Long. On finding all reducts of consistent decision tables. *Cybernetics and Information Technologies*, 14(4):3–10, 2014.

[6] Mi, Ju-Sheng, Wu, Wei-Zhi, and Zhang, Wen-Xiu. Approaches to knowledge reduction based on variable precision rough set model. *Information sciences*, 159(3):255–272, 2004.

[7] Min, Fan, He, Huaping, Qian, Yuhua, and Zhu, William. Test-cost-sensitive attribute reduction. *Information Sciences*, 181(22):4928–4942, 2011.

[8] Pawlak, Zdzisław. Rough sets. *International Journal of Computer & Information Sciences*, 11(5):341–356, 1982.

[9] Pawlak, Zdzisław and Skowron, Andrzej. Rough sets and boolean reasoning. *Information sciences*, 177(1):41–73, 2007.

[10] Qian, Yuhua and Liang, Jiye. Combination entropy and combination granulation in rough set theory. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 16(02):179–193, 2008.

[11] Skowron, Andrzej and Rauszer, Cecylia. The discernibility matrices and functions in information systems. In *Intelligent Decision Support*, pages 331–362. Springer, 1992.

[12] Thi, Vu Duc. The minimal keys and antikeys. *Acta Cybernetica*, 7(4):361–371, 1986.

[13] Thi, Vu Duc and Giang, Nguyen Long. A method to construct decision table from relation scheme. *Cybernetics and Information Technologies*, 11(3):32–41, 2011.

[14] Thi, Vu Duc and Giang, Nguyen Long. Some problems concerning condition attributes and reducts in decision tables. In *Proceeding of the fifth National Symposium Fundamental and Applied Information Technology Research*, pages 142—152. FAIR, Dong Nai, Vietnam, 2012.

[15] Vens, Celine, Struyf, Jan, Schietgat, Leander, Džeroski, Sašo, and Blockeel, Hendrik. Decision trees for hierarchical multi-label classification. *Machine Learning*, 73(2):185–214, 2008.

[16] Yao, Yiyu and Zhao, Yan. Discernibility matrix simplification for constructing attribute reducts. *Information sciences*, 179(7):867–882, 2009.

[17] Zheng, Kai, Hu, Jie, Zhan, Zhenfei, Ma, Jin, and Qi, Jin. An enhancement for heuristic attribute reduction algorithm in rough set. *Expert Systems with Applications*, 41(15):6748–6754, 2014.