Reducing Attributes in Rough Set Theory with the Viewpoint of Mining Frequent Patterns

Thanh-Trung Nguyen
Department of Computer Science
University of Information Technology,
Vietnam National University HCM City
Ho Chi Minh City, Vietnam

Phi-Khu Nguyen
Department of Computer Science
University of Information Technology,
Vietnam National University HCM City
Ho Chi Minh City, Vietnam

***Abstract*—The main objective of the Attribute Reduction problem in Rough Set Theory is to find and retain the set of attributes whose values vary most between objects in an Information System or Decision System. Besides, Mining Frequent Patterns aims finding items that the number of times they appear together in transactions exceeds a given threshold as much as possible. Therefore, the two problems have similarities. From that, an idea formed is to solve the problem of Attribute Reduction from the viewpoint and method of Mining Frequent Patterns. The main difficulty of the Attribute Reduction problem is the time consuming for execution, NP-hard. This article proposes two new algorithms for Attribute Reduction: one has linear complexity, and one has global optimum with concepts of Maximal Random Prior Set and Maximal Set.**

Keywords—accumulating frequent patterns; attribute reduction; maximal set; maximal random prior set; mining frequent patterns; rough set

I. INTRODUCTION

Attribute reduction has played an important role in rough set applied in many fields, such as data mining, pattern recognition, machine learning. In recent years, there are many proposed reduction algorithms based on positive-region, information entropy and discernibility matrix (Qian et. al. 2011).

Attribute reduction methods have been applied by reducing inadequate features to discover hidden patterns from high dimensional data sets. Meanwhile, the nature of the original features still remained and the time consuming for pattern recognition has been decreased (Dash et. al. 2010) (Liang et. al. 2013) (Qian et. al. 2010). The characteristics of the data set are remained by keeping the important attributes. Therefore, the quality of this data set has been enhanced through the removal of redundant attributes (Sadasivam et. al. 2012). Also, rule induction can be applied in rough set theory due to attribute reduction algorithms (Yao and Zhao 2008) (Ju et. al. 2011).

One of applications of attribute reduction is gene selection. A paper presented a Quick Reduct based Genetic Algorithm (Anitha 2012) while a minimal spanning tree based on rough set theory for gene selection was introduced (Pati and Das 2013). Based on cross entropy, the relatively dispensable attributes have been omitted in the decision system and the optimal attributes set has described the same discriminative features for the original data set (Zheng and Yan 2012). In the sense of entropies, many discernibility matrixes were introduced (Wei et. al. 2013).

Based on indiscernibility and discernibility, similarities and differences of objects have been figured out and hence, attribute reduction has been carried out according to these basic theories. Attribute set is reduced by generating redacts using the indiscernibility relation of Rough Set Theory (Sengupta and Das 2012). By transforming discernibility matrix into a simplest equivalent matrix, valuable attributes have been retained while unimportant attributes will be removed from the discernibility matrix (Yao and Zhao 2009). An attribute reduction algorithm based on genetic algorithm with improved selection operator and discernibility matrix was researched and introduced (Zhenjiang et. al. 2012). Some others discussed an algorithm on discernibility matrix and Information Gain to reduce attributes (Azhagusundari and Thanamani 2013).

In addition, a proposed hybrid algorithm for large data sets was studied to overcome the shortcoming about computationally time-consuming and inefficient significance measure for more attributes with the same greatest value (Qian et. al. 2011).

Heterogeneous attribute reduction technique can be based on neighborhood rough sets by using neighborhood dependency to evaluate the discriminating capability of a subset of heterogeneous attributes. This neighborhood model reduced the attributes according to the thresholds of samples in decision positive region (Hu et. al. 2008).

In incomplete decision systems, attribute reduction methods, such as distributive reduction and positive region reduction have been given by discernibility function (Jilin et. al. 2009). To deal with these systems, a paper proposed a new attribute reduction method based on information quantity. This approach improved traditional tolerance relationship calculation methods using an extension of tolerance relationship in rough set theory (Xu et. al. 2012). Another research presented a new attribute reduction algorithm based on incomplete decision table, which improves the two aspects of time and space complexity (Yue et. al. 2012).

Handling attribute reduction problem in special systems is also a challenged issue. There are some researches in attribute reduction about dynamic data sets (Wang et. al. 2013), fuzzy sets (Chen et. al. 2012), Inconsistent Disjunctive Set-valued Ordered Decision Information System (Zhang et. al. 2012) etc.

Even, the design and implement of rough set processor in VHDL have studied on Binary Discernibility matrix and reduct calculator block (Tiwari et. al. 2012). Thereby, the speed of the operation for a dedicated hardware has been increased.

The calculation time is always a big issue in attribute reduction. A new accelerator for attribute reduction has been proposed based on perspective of objects and attributes (Liang et. al. 2013). Particle swarm optimization was a new heuristic algorithm which has been applied to many optimization problems successfully (Ding et. al. 2012). Nowadays, it is often used to solve non-deterministic polynomial (NP)-hard problem such as attribute reduction problem. Co-PSAR was introduced based on this idea to find the minimal reduction set. An algorithm based on rough set and Wasp Swarm Optimization was also introduced. It utilizes mutual information based information entropy to find core attributes, and then utilizes the significance of feature as probability information to search through the feature space for minimum attributes reduction result (Fan and Zhong 2012). A popular method in swarm intelligence is Ant Colony Optimization (ACO). A research proposed hybrid approach can help in improving classification accuracy and also in finding more robust features to improve classifier performance based on ACO (Arafat et. al. 2013).

Genetic algorithm was also researched and applied to attribute reduction. The convergence speed of algorithm is faster in global optimal solution (Zhenjiang et. al. 2012) (Liu et. al. 2013).

Besides, granular computing has been a new research approach studied to reduce the attribute in decision system (Li et. al. 2013). A paper presented a novel granularity partition model and developed a fast effective feature selection algorithm in decision systems (Sun et. al. 2012).

Some other approaches have been researched recently about Nonlinear Great Deluge Algorithm (Jaddi and Abdullah 2013), Quantization (Li et. al. 2012), attribute significance (Zhai et. al. 2012), degree of condition attributes (Qiu et. al. 2012) … They are all proved their efficiency in solving attribute reduction problem.

This article introduces an algorithm based on bit-chains and maximal random prior set. It finds out a reduction with linear time but the result is not global optimization. Therefore, another algorithm based on maximal set (a new development of maximal random prior set) and the algorithm for Accumulating Frequent Pattern (Nguyen TT and Nguyen PK 2013) to find a global optimal reduction is also proposed.

II. Formulation Model

Definition 1 (***bit-chain***): $< a_1a_2 \dots a_m >$ (for $a_i \in \{0, 1\}$) is a m-bit-chain. Zero chain is a bit-chain with each bit equals 0.

Definition 2 (***intersection operation*** $\overline{\cap}$): The intersection operation $\overline{\cap}$ is a dyadic operation in bit-chains space.

$< a_1a_2 \dots a_m > \overline{\cap} < b_1b_2 \dots b_m > = < c_1c_2 \dots c_m >$, $a_i, b_i \in \{0, 1\}$, $c_i = \min(a_i, b_i)$

Definition 3 (***cover operation***$\hookrightarrow$): A bit-chain A is said to cover a bit-chain B if and only if with every position having bit-1 turned on in B, A has a corresponding bit-1 turned on.

Let $A = < a_1a_2 \dots a_m >$, $B = < b_1b_2 \dots b_m >$, $(\forall b_{i=1..m} \mid (b_i = 1) \rightarrow (a_i = 1)) \Rightarrow A \hookrightarrow B$

Consequence 1: A bit-chain the result of an intersection operation and differing from zero chain is always covered by two bit-chains generating it.

$(A \ \overline{\cap}\ B = C) \wedge (C \neq 0) \Rightarrow (A \hookrightarrow C) \wedge (B \hookrightarrow C)$

Definition 4 (***maximal random prior form* δ – S**): The maximal random prior form of a set S of bit-chains, denoted by $\delta - S$, is a bit-chain satisfying four criteria:

- Being covered most by elements in S.
- Being covered by the first element in S.
- Having number of bit-1 turned on as much as possible.
- If there are more than one bit-chain meeting three criteria above, the bit-chain chosen to be the maximal random prior form of S is one covered by the first elements in S.

For example, consider a set of 4-bit-chains <*abcd*>:

$$S = \{ \quad \begin{array}{cccccc} & a & b & c & d & \\ (& \boxed{1} & 0 & \boxed{1} & 1 &); \\ (& 0 & 0 & \boxed{1} & 1 &); \\ (& \boxed{1} & 1 & 0 & 0 &); \\ (& \boxed{1} & 0 & \boxed{1} & 0 &) \end{array} \quad \}$$

Review three bit-chains:

<0011>: has two bit-1 turned on but is only covered by the first two bit-chains of S.

<1000>: has one bit-1 turned on and is covered by three bit-chains of S.

<0010>: has one bit-1 turned on and is covered by three bit-chains of S.

Between <1000> and <0010>, <0010> is covered by the first two elements in S, so $\delta - S$ has to be <0010>.

Definition 5 (***maximal random prior elements***): Maximal random prior elements of set S of bit-chains have the following characteristics:

The first element (p_1) is form $\delta - S$

The second element (p_2) is form $\delta - S\backslash\{x \in S \mid x \hookrightarrow p_1\}$

The third element (p_3) is form $\delta - S\backslash(\{x \in S \mid x \hookrightarrow p_1\} \cup \{x \in S \mid x \hookrightarrow p_2\})$

…

The k^{th} element (p_k) is form $\delta - S\backslash(\{x \in S \mid x \hookrightarrow p_1\} \cup \{x \in S \mid x \hookrightarrow p_2\} \cup \dots \cup \{x \in S \mid x \hookrightarrow p_{k-1}\})$

and $S = \{x \in S \mid x \hookrightarrow p_1\} \cup \{x \in S \mid x \hookrightarrow p_2\} \cup \dots \cup \{x \in S \mid x \hookrightarrow p_k\}$

Definition 6 (***maximal random prior set***): A set P containing all maximal random prior elements of a set S of bit-chains is called maximal random prior set of S.

Consequence 2: All elements in maximal random prior set *P* do not have any the same position where bit-1 turned on.

Consequence 3: When the bit-chains set is arranged in different orders, it will produce different maximal random prior sets.

Theorem 1: When the intersection operations are made between an element in *S* and elements in *P*, the results differing from zero chain will not cover each other.

Proof: According to *Consequence 2*, the results made from intersection operations of an element in *S* and elements in *P* will not have bit-1 turned on at the same position. So that, these will not cover each other.

III. Algorithm for Finding Maximal Random Prior Set

A. Idea

Consider a Boolean function *f* the intersection ($\wedge$) of *n* propositions. Each proposition in *f* is a union ($\vee$) of *m* variables a_1, a_2, ..., a_m. According to commutative law of Boolean algebra, *n* propositions of *f* can be changed into the form: $f = A_1 \wedge A_2 \wedge ... \wedge A_m$, with:

$A_1 = \wedge_{k_1} (a_1 \vee \dots)$

$A_2 = \wedge_{k_2} (a_2 \vee \dots)$ A_2 does not contain a_1

$A_3 = \wedge_{k_3} (a_3 \vee \dots)$ A_3 does not contain a_1, a_2

…

$A_m = \wedge_{k_m} (a_m \vee \dots)$ A_m does not contain $a_1, a_2, \dots, a_{m-1}$

$\forall i = 1..m; 0 \le k_i \le n \mid k_1 + k_2 + ... + k_m = n;$

$\forall k_p \ne 0; 1 \le p \le m \mid A_p = a_p \vee X_p$; X_p is a certain proposition.

So, $f = \wedge A_p = \wedge (a_p \vee X_p) = (\wedge a_p) \vee (\wedge X_p)$

Clearly, $(\wedge a_p)$ is a reduction of *f*.

If *n* propositions in *f* are transformed into a set *S* of *m*-bit-chains, the maximal random prior set *P* will be a reduction of *f*.

According to the above analysis, an algorithm is taken shape to construct maximal random prior set *P* of the bit-chains set *S* with the following main ideas:

Each element in set *S* will be inspected with the existing order in *S*. At the same time, the set *P* will be also created or modified correspondingly with the number of elements inspected in *S*.

The initial set *P* is empty. Obviously, the set *S* with one first element has the corresponding set *P* also containing only this first element.

Scanning the next element of *S*, the intersection operations ($\overline{\cap}$) made between this element and the existing elements of *P* to find out the new maximal random prior forms. If the new form is generated, it will replace the old form in *P* because this new form is covered by elements of *S* more than the old form, evidently. If the new form is not generated, obviously, the next element of *S* is one new maximal random prior form.

However, a question maybe be brought out. Whenever the next element in *S* is inspected, the elements have to carry out intersection operations with the existing elements in *P*; at that time, we have two element groups listed such as: (1) the old elements of *P*, (2) the new elements created by the intersection operations. Maybe, the new elements will cover together or cover the old elements or be covered by the old elements. Therefore, whether the set *P* is ensured the consistency as *Consequence 2* stated? The answer is "Yes" since *Consequence 1* and *Theorem 1* are generated to ensure this.

B. Proposed Algorithm

```
FIND_MaximalRandomPriorSet
Input: m-bit-chains set S
Output: maximal random prior set P
1.  P = ∅;
2.  for each s in S do
3.    flag = False;
4.    for each p in P do
5.      temp = s $\overline{\cap}$ p;
6.      if temp <> 0 then//temp differs
                                from zero chain
7.        replace p in P by temp;
8.        flag = True;
9.        break;
10.     end if;
11.   end for;
12.   if flag = False then
13.     P = P ∪ {s};//s becomes ending
                                  element of P
14.   end if;
15. end for;
16. return P;
```

C. Accuracy of The Algorithm

Theorem 2: *FIND_MaximalRandomPriorSet* algorithm can find out the maximal random prior set *P* of a bit-chains set *S* with a given order.

Proof by Induction:

With number of elements in *S* is 1, the only element in *S* is also form $\delta - S$. According to the algorithm, the only element in *S* is inserted into *P*. Then, the only element in *P* satisfies the definition of maximal random prior set. Since, *Theorem 2* is correct when *S* has 1 element.

Assume that *Theorem 2* is correct when *S* has *k* elements. We need to prove *Theorem 2* is correct when *S* has *k* + 1 element, too.

Because *Theorem 2* is correct when *S* has *k* elements, we have the set *P* contains all maximal random prior elements of this set *S*.

When *S* has *k* + 1 elements, it means the original set *S* having *k* elements are added a new element.

According to *FIND_MaximalRandomPriorSet* algorithm, we make intersection operations between elements in current *P* and the new (*k* + 1)th element denoted s_{k+1} in *S* (line 4 and line 5):

- If the result of the intersection operation between s_{k+1} and an element p_i in P differs from zero chain (line 6), this result is form $\delta - S \backslash (\{x \in S \mid x \hookrightarrow p_1\} \cup \{x \in S \mid x \hookrightarrow p_2\} \cup ... \cup \{x \in S \mid x \hookrightarrow p_{i-1}\})$, with S has $k + 1$ elements. Replace p_i in P by this new result element (line 7). When s_{k+1}, together with p_i, create a new maximal random prior form, we terminate intersection operations between s_{k+1} and remaining elements in P (line 9).
- If all intersection operations between s_{k+1} and each element in P return zero chain, it means s_{k+1} does not cover any element in P. Thus, the element s_{k+1} is form $\delta - \{s_{k+1}\}$, then s_{k+1} is inserted into P (line 13).

In both cases, we receive the set P satisfying the properties of the maximal random prior set of S. So, *Theorem 2* is correct when S has $k + 1$ element.

In conclusion: *FIND_MaximalRandomPriorSet* algorithm can find out the maximal random prior set P of a bit-chains set S with a given order.

IV. Attribute Reduction in Rough Set Theory

The maximal random prior set P is useful in solving and reducing Boolean algebra functions. One of the most important applications of the set P is finding out a solution of attribute reduction problem in rough set theory.

A. Rough Set

In rough set theory, information system is a pair (U, A), where U is a non-empty finite set of objects and A is a non-empty finite set of attributes. A decision system is any information system of the form $(U; A \cup \{d\})$, where $d \notin A$ is decision attribute.

TABLE I. A Decision System "Play Sport"

	Wind	Temperature	Humidity	Outlook	Play Sport
x_1	Strong	Hot	Normal	Sunny	Yes
x_2	Strong	Mild	Normal	Rain	No
x_3	Weak	Hot	Normal	Rain	No
x_4	Weak	Cool	High	Rain	Yes

With $|U|$ denotes cardinal of U, discernibility matrix of a decision system is a symmetric $|U|\text{x}|U|$ matrix with each entry $c_{ij} = \{a \in A \mid a(x_i) \neq a(x_j)\}$ if $d(x_i) \neq d(x_j)$, otherwise $c_{ij} = \varnothing$.

TABLE II. Discernibility Matrix of Decision System "Play Sport"

	x_1	x_2	x_3	x_4
x_1	∅	∅	∅	∅
x_2	b,d	∅	∅	∅
x_3	a,d	∅	∅	∅
x_4	∅	a,b,c	b,c	∅

Table II presents a discernibility matrix of decision system "Play Sport" where a, b, c, d denote Wind, Temperature, Humidity and Outlook, respectively.

Discernibility function is a Boolean function retrieved from discernibility matrix and can be defined by the formula $f = \wedge \{ \vee c_{ij} \mid c_{ij} \neq \varnothing\}$. According to *Table II*, we have discernibility function $f = (b \vee d) \wedge (a \vee d) \wedge (a \vee b \vee c) \wedge (b \vee c)$.

Discernibility function can be simplified by using laws of Boolean algebra. All constituents in the minimal disjunctive normal form of this function are all reductions of decision system (Pawlak 2003). However, simplifying discernibility function is a NP-hard problem and attribute reduction is always the key problem in rough set theory.

B. The Maximal Random Prior Set and Attribute Reduction Problem

Consider a discernibility function f retrieved from discernibility matrix of a decision system with m attributes has n constituents. Each constituent in this function will be transformed into an m-bit-chain, with each bit denotes an attribute. The function will be converted into a set S has n bit-chains. The maximal random prior set P of the set S is the simplification of discernibility function f.

Set P shows (some) reduction(s) of function f. With each bit-chain in P, the positions where bit-1 is turned on need to be noticed. Value 1 of a bit means that the corresponding attribute will appear in reduction of f. The collection of all attributes retrieved from set P is a simplification of discernibility function f.

Example: According to discernibility function f of decision system in *Table I*, the set S includes:

$$\begin{aligned} S = \{ \quad & (0\ 1\ 0\ \boxed{1}); \\ & (1\ 0\ 0\ \boxed{1}); \\ & (1\ \boxed{1\ 1}\ 0); \\ & (0\ \boxed{1\ 1}\ 0) \quad \} \end{aligned}$$

Initialize $P = \varnothing$. Scan all elements in S

$S[1] = (0\ 1\ 0\ 1) \rightarrow$ insert $(0\ 1\ 0\ 1)$ into $P \rightarrow P = \{ (0\ 1\ 0\ 1) \}$

$S[2] = (1\ 0\ 0\ 1) \rightarrow (1\ 0\ 0\ 1) \overline{\cap} (0\ 1\ 0\ 1) = (0\ 0\ 0\ 1) \rightarrow$ replace $(0\ 1\ 0\ 1)$ in P by $(0\ 0\ 0\ 1) \rightarrow \text{P} = \{ (0\ 0\ 0\ 1) \}$

$S[3] = (1\ 1\ 1\ 0) \rightarrow (1\ 1\ 1\ 0) \overline{\cap} (0\ 0\ 0\ 1) = (0\ 0\ 0\ 0) \rightarrow$ insert $(1\ 1\ 1\ 0)$ into $P \rightarrow P = \{ (0\ 0\ 0\ 1); (1\ 1\ 1\ 0) \}$

$S[4] = (0\ 1\ 1\ 0) \rightarrow (0\ 1\ 1\ 0) \overline{\cap} (0\ 0\ 0\ 1) = (0\ 0\ 0\ 0); (0\ 1\ 1\ 0) \overline{\cap} (1\ 1\ 1\ 0) = (0\ 1\ 1\ 0) \rightarrow$ replace $(1\ 1\ 1\ 0)$ in P by $(0\ 1\ 1\ 0) \rightarrow P = \{ (0\ 0\ 0\ 1); (0\ 1\ 1\ 0) \}$

$(0\ 0\ 0\ 1) \rightarrow d$ and $(0\ 1\ 1\ 0) \rightarrow b \vee c$

So, minimal function $f = d \wedge (b \vee c)$.

In conclusion, $(d \wedge b)$ and $(d \wedge c)$ are two reductions of discernibility function f.

V. Experimentation 1

FIND_MaximalRandomPriorSet algorithm is developed and tested on a personal computer with specification: Windows 7 Ultimate 32-bit, Service Pack 1 Operating System; 4096MB RAM; Intel(R) Core(TM)2 Duo, E7400, 2.80GHz; 300GB

HDD. Programming language is C#.NET on Visual Studio 2008. The results of some testing patterns:

TABLE III. SOME TESTING PATTERNS OF *FIND_MAXIMALRANDOMPRIORSET* ALGORITHM

Length of bit-chain	Number of bit-chains	Time (unit: second)
10	1,000,000	0.2184004
10	2,000,000	0.3900007
10	5,000,000	1.0140017
10	10,000,000	2.0436036
25	1,000,000	0.2808005
25	2,000,000	0.546001
25	5,000,000	1.123202
25	10,000,000	2.8236049
50	1,000,000	0.2496004
50	2,000,000	0.7176013
50	5,000,000	1.9032033
50	10,000,000	3.978007
60	1,000,000	0.3744007
60	2,000,000	0.7644014
60	5,000,000	1.9344034
60	10,000,000	4.1964073

Attribute Reduction based on bit-chains and maximal random prior set has just been introduced. It found a reduction with linear time but the result is not global optimization. The following example will show this problem clearly:

$$S = \{ \quad \begin{array}{cccccc} a & b & c & d & e & g \\ (1 & 1 & 0 & 1 & 0 & 0); \\ (1 & 1 & 0 & 1 & 0 & 0); \\ (1 & 0 & 0 & 0 & 0 & 0); \\ (0 & 0 & 1 & 1 & 0 & 0); \\ (0 & 0 & 1 & 1 & 0 & 0); \\ (0 & 0 & 1 & 0 & 0 & 0); \\ (0 & 1 & 0 & 1 & 1 & 1); \\ (0 & 1 & 0 & 1 & 1 & 1) \end{array} \quad \}$$

When applying *FIND_MaximalRandomPriorSet* algorithm to *S*, we have:

$P = \{ (1\,0\,0\,0\,0\,0); (0\,0\,1\,0\,0\,0); (0\,1\,0\,1\,1\,1) \}$

$(1\,0\,0\,0\,0\,0) \rightarrow a$; $(0\,0\,1\,0\,0\,0) \rightarrow c$; $(0\,1\,0\,1\,1\,1) \rightarrow b \vee d \vee e \vee g$

So, minimal function $f = a \wedge c \wedge (b \vee d \vee e \vee g)$. Hence, $(a \wedge c \wedge b)$, $(a \wedge c \wedge d)$, $(a \wedge c \wedge e)$, and $(a \wedge c \wedge g)$ are four reductions of discernibility function *f*.

But, if the order of the elements in *S* is changed as follows:

$$S = \{ \quad \begin{array}{cccccc} a & b & c & d & e & g \\ (1 & 1 & 0 & 1 & 0 & 0); \\ (1 & 1 & 0 & 1 & 0 & 0); \\ (0 & 0 & 1 & 1 & 0 & 0); \\ (0 & 0 & 1 & 1 & 0 & 0); \\ (0 & 1 & 0 & 1 & 1 & 1); \\ (0 & 1 & 0 & 1 & 1 & 1) \\ (1 & 0 & 0 & 0 & 0 & 0); \\ (0 & 0 & 1 & 0 & 0 & 0); \end{array} \quad \}$$

then we have:

$P = \{ (0\,0\,0\,1\,0\,0); (1\,0\,0\,0\,0\,0); (0\,0\,1\,0\,0\,0) \}$

$(0\,0\,0\,1\,0\,0) \rightarrow d$; $(1\,0\,0\,0\,0\,0) \rightarrow a$; $(0\,0\,1\,0\,0\,0) \rightarrow c$

So, minimal function $f = d \wedge a \wedge c$. This is also the reduction of discernibility function *f*. Now, we can see that this result is better than the above one because it emphasize the importance of attribute *d* (the values of *d* show the difference up to 6 times between the objects), and it also is succinct.

Obviously, with an arbitrary order of elements in *S*, *FIND_MaximalRandomPriorSet* algorithm can not find out the best result.

The next section is going to propose a new model which is based on maximal set (a new development of maximal random prior set) and *NewRepresentative*, the algorithm for Accumulating Frequent Patterns (Nguyen TT and Nguyen PK 2013) to find a global optimal reduction.

VI. MAXIMAL SET

***Definition 7* (*maximal form* ε – S)**: The maximal form of a set *S* of bit-chains, denoted by $\varepsilon - S$, is a bit-chain which is covered most by elements in *S*.

***Definition 8* (*maximal elements*)**: Maximal elements of set *S* of bit-chains have the following characteristics:

The first element (q_1) is form $\varepsilon - S$

The second element (q_2) is form $\varepsilon - S \backslash \{x \in S \mid x \hookrightarrow q_1\}$

The third element (q_3) is form $\varepsilon - S \backslash \{x \in S \mid x \hookrightarrow q_1\} \cup \{x \in S \mid x \hookrightarrow q_2\}$

…

The k^{th} element (q_k) is form $\varepsilon - S \backslash \{x \in S \mid x \hookrightarrow q_1\} \cup \{x \in S \mid x \hookrightarrow q_2\} \cup \ldots \cup \{x \in S \mid x \hookrightarrow q_{k-1}\}$

and $S = \{x \in S \mid x \hookrightarrow q_1\} \cup \{x \in S \mid x \hookrightarrow q_2\} \cup \ldots \cup \{x \in S \mid x \hookrightarrow q_k\}$

***Definition 9* (*maximal set*)**: A set *Q*, which contains all maximal elements of a bit-chain set *S*, is called maximal set of *S*.

VII. THE ALGORITHM FOR FINDING MAXIMAL SET

In one of our previous papers, we introduced an algorithm to find out all frequent patterns of *S* set of transactions. Each transaction is a bit-chain with bit locations stand for a type of items. If bit-1 is turned on, it means customer bought it in transaction and otherwise. The algorithm for accumulating frequent patterns finds out the representative set P^* every time when a new bit-chain added to *S* (Nguyen TT and Nguyen PK 2013). The below is the full algorithm:

```
NewRepresentative
Input: P* is a representative set of S,
       z is a bit-chain added to S.
Output: The new representative set P* of
S ∪ {z}.
1.  M = ∅  // M: set of new elements of P*
2.  flag1 = 0
3.  flag2 = 0
4.  for each x ∈ P* do
```

```
    q = x o [z; 1]
    if q ≠ 0 // q is not a chain with all bits 0
      if x ⊆ q then $P^*$ = $P^*$ \ {x}
      if [z; 1] ⊆ q then flag1 = 1
      for each y ∈ M do
        if y ⊆ q then
          M = M \ {y}
          break for
        endif
        if q ⊆ y then
          flag2 = 1
          break for
        endif
      endfor
    else
      flag2 = 1
    endif
    if flag2 = 0 then M = M ∪ {q}
    flag2 = 0
  endfor
  if flag1 = 0 then $P^*$ = $P^*$ ∪ {[z; 1]}
  $P^*$ = $P^*$ ∪ M
  return $P^*$
```

Note (Nguyen TT and Nguyen PK 2013):

- $[z; n]$ is called a pattern. z is a bit-chain and n is the frequency ($n \in \aleph$, $n \geq 0$).
- o is called intersection operation between 2 patterns. $[a_1a_2\ldots a_m; n_1]$ o $[b_1b_2\ldots b_m; n_2] = [c_1c_2\ldots c_m; n_1 + n_2]$; a_i, $b_i \in \{0, 1\}$, $c_i = \min(a_i, b_i)$
- $\subseteq$ is called contained operation between 2 patterns. $[u_1; n_1] \subseteq [u_2; n_2] \Leftrightarrow (u_1 = u_2) \wedge (n_1 \leq n_2)$
- A pattern $[u; k]$ of S is called *maximal pattern* – denoted $[u; k]_{\max\to S}$ – if and only if it doesn't exist k' such that $[u; k']_{\max\to S}$ and $k' > k$.
- P^* is *representative set* of S when $P^* = \{[u; n]_{\max\to S} \mid \nexists[v; m]_{\max\to S} : (v \hookrightarrow u \text{ and } m > n)\}$. Each element in P^* is called a *representative pattern* of S.

Consequence 4: The bit-chain of the pattern which has the highest frequency in Representative Set of a set S is the maximal form of S.

From *Consequence 4*, the *Definition 8* can be modified to become the following definition.

Definition 10 (***maximal elements***): Maximal elements of set S of bit-chains have the following characteristics:

The first element (q_1) is the element $\{y_0 \in P^* \mid \exists x \in S, x \hookrightarrow y_0$ and $\forall y \in P^* \mid \exists x \in S, x \hookrightarrow y \Rightarrow y_0.frequency > y.frequency\}$

The second element (q_2) is the element $\{y_0 \in P^* \mid \exists x \in S_1, x \hookrightarrow y_0$ and $\forall y \in P^* \mid \exists x \in S_1, x \hookrightarrow y \Rightarrow y_0.frequency > y.frequency\}$, here $S_1 = S \backslash \{x \in S \mid x \hookrightarrow q_1\}$

The third element (q_3) is the element $\{y_0 \in P^* \mid \exists x \in S_2, x \hookrightarrow y_0$ and $\forall y \in P^* \mid \exists x \in S_2, x \hookrightarrow y \Rightarrow y_0.frequency > y.frequency\}$, here $S_2 = S_1 \backslash \{x \in S_1 \mid x \hookrightarrow q_2\}$

…

The (k + 1)th element (q_{k+1}) is the element $\{y_0 \in P^* \mid \exists x \in S_k, x \hookrightarrow y_0$ and $\forall y \in P^* \mid \exists x \in S_k, x \hookrightarrow y \Rightarrow y_0.frequency > y.frequency\}$, here $S_k = S_{k-1} \backslash \{x \in S_{k-1} \mid x \hookrightarrow q_k\}$

After *Definition 10* is appeared, the algorithm for finding Maximal Set is created as follows:

```
FIND_MaximalSet
Input: m-bit-chains set S
       A representative set $P^*$ of S
Output: the maximal set Q
  while S is not empty do
    z = GetMaximalForm($P^*$)
    Q = Q ∪ {z}
    for each x ∈ S do
      if x o z ≠ 0 then
        S ← S \ {x}
      endif
    endfor
  endwhile
  return Q
```

The pseudo-code of *GetMaximalForm* algorithm is shown here:

```
GetMaximalForm
Input: A representative set $P^*$
Output: the maximal form m
  m = $P^*$[1];
  for each x ∈ $P^*$ do
    if m.Frequency < x.Frequency then
      m = x;
    endif
  endfor
  for each x ∈ $P^*$ do
    if x o m ≠ 0 then
      $P^*$ = $P^*$\{x}
    endif
  endfor
  return m;
```

Theorem 3: *FIND_MaximalSet* algorithm can find out the maximal set Q of a bit-chains set S.

Proof: The algorithm *FIND_MaximalSet* works as follows: First we find the element q_j, then delete elements in P^* and in S covering q_j. Repeat this until the set S is empty.

In the above, if we do not delete elements in P^* covering q_j, then we can see that the q_j we find is the same as in *Definition 10*. Hence to prove the correctness of the *FIND_MaximalSet* algorithm, we need to show that when we delete elements in P^* covering q_j then we obtain the same maximal elements as defined in *Definition 10*.

We show this by induction on j.

If $j = 1$, then q_1 is determined unambiguously. We define $S_1 = S \backslash \{x \in S \mid x \hookrightarrow q_1\}$ and $P_1 = P^* \backslash \{y \in S \mid y \hookrightarrow q_1\}$.

From *Definition 10*, q_2 is determined as follows: It is the element in P^* which is covered by at least one element in S_1 and is the one with the most frequency among such. We show now that q_2 can be determined from P_1 by the same criteria.

Indeed, if q_2 is not an element in P_1, then by definition q_2 must cover q_1. Now by the choice of q_2, q_2 must be covered by one element in S_1, called that element x_2. Since $x_2 \hookrightarrow q_2$ and $q_2 \hookrightarrow q_1$, x_2 must cover q_1. But x_2 is an element in S_1, and any element in S_1 cannot cover q_1, hence we obtain a contradiction. This shows that *Theorem 3* is true for $j = 2$.

Now assume that *Theorem 3* is true for $j = j_0$. We now prove it is true for $j = j_0 + 1$. We prove this exactly like the case from $j = 1$ to $j = 2$ above. (Q.E.D.)

Similar to the Maximal Random Prior Set, one of the applications which can integrate the Maximal Set is reducing the discernibility function of rough set. Consider an example of discernibility function $f = (a \vee b \vee d) \wedge (a \vee b \vee d) \wedge a \wedge (c \vee d) \wedge (c \vee d) \wedge c \wedge (b \vee d \vee e \vee g) \wedge (b \vee d \vee e \vee g)$ with a, b, c, d, e, g are attributes in a decision system. Change f to a set of bit-chains S:

$$S = \{ \begin{array}{cccccc} a & b & c & d & e & g \\ (1 & 1 & 0 & 1 & 0 & 0); \\ (1 & 1 & 0 & 1 & 0 & 0); \\ (1 & 0 & 0 & 0 & 0 & 0); \\ (0 & 0 & 1 & 1 & 0 & 0); \\ (0 & 0 & 1 & 1 & 0 & 0); \\ (0 & 0 & 1 & 0 & 0 & 0); \\ (0 & 1 & 0 & 1 & 1 & 1); \\ (0 & 1 & 0 & 1 & 1 & 1) \end{array} \}$$

Initialize $P^* = \varnothing$. Scan all elements in S.

* S[1] = [110100; 1]: P^* is empty. Put S[1] into P^*. P^* = {[110100; 1]}

* S[2] = [110100; 1]:

S[2] o P^*[1] = [110100; 1] o [110100; 1] = [110100; 2]

// [110100; 1] ⊆ [110100; 2]

P^* = {[110100; 2]}

* S[3] = [100000; 1]:

S[3] o P^*[1] = [100000; 1] o [110100; 2] = [100000; 3]

// S[3] ⊆ [100000; 3]

P^* = {[110100; 2]; [100000; 3]}

* S[4] = [001100; 1]:

S[4] o P^*[1] = [001100; 1] o [110100; 2] = [000100; 3]

S[4] o P^*[2] = [001100; 1] o [100000; 3] = **0** (zero chain)

P^* = {[110100; 2]; [100000; 3]; [000100; 3]; [001100; 1]}

* S[5] = [001100; 1]:

S[5] o P^*[1] = [001100; 1] o [110100; 2] = [000100; 3]

S[5] o P^*[2] = [001100; 1] o [100000; 3] = **0** (zero chain)

S[5] o P^*[3] = [001100; 1] o [000100; 3] = [000100; 4]

S[5] o P^*[4] = [001100; 1] o [001100; 1] = [001100; 2]

// [000100; 3] ⊆ [000100; 4]

// [001100; 1] ⊆ [001100; 2]

P^* = {[110100; 2]; [100000; 3]; [000100; 4]; [001100; 2]}

* S[6] = [001000; 1]:

S[6] o P^*[1] = [001000; 1] o [110100; 2] = **0** (zero chain)

S[6] o P^*[2] = [001000; 1] o [100000; 3] = **0** (zero chain)

S[6] o P^*[3] = [001000; 1] o [000100; 4] = **0** (zero chain)

S[6] o P^*[4] = [001000; 1] o [001100; 2] = [001000; 3]

// S[6] ⊆ [001000; 3]

P^* = {[110100; 2]; [100000; 3]; [000100; 4]; [001100; 2]; [001000; 3]}

* S[7] = [010111; 1]:

S[7] o P^*[1] = [010111; 1] o [110100; 2] = [010100; 3]

S[7] o P^*[2] = [010111; 1] o [100000; 3] = **0** (zero chain)

S[7] o P^*[3] = [010111; 1] o [000100; 4] = [000100; 5]

S[7] o P^*[4] = [010111; 1] o [001100; 2] = [000100; 3]

S[7] o P^*[5] = [010111; 1] o [001000; 3] = **0** (zero chain)

// [000100; 3], [000100; 4] ⊆ [000100; 5]

P^* = {[110100; 2]; [100000; 3]; [001100; 2]; [001000; 3]; [010100; 3]; [000100; 5]; [010111; 1]}

* S[8] = [010111; 1]:

S[8] o P^*[1] = [010111; 1] o [110100; 2] = [010100; 3]

S[8] o P^*[2] = [010111; 1] o [100000; 3] = **0** (zero chain)

S[8] o P^*[3] = [010111; 1] o [001100; 2] = [000100; 3]

S[8] o P^*[4] = [010111; 1] o [001000; 3] = **0** (zero chain)

S[8] o P^*[5] = [010111; 1] o [010100; 3] = [010100; 4]

S[8] o P^*[6] = [010111; 1] o [000100; 5] = [000100; 6]

S[8] o P^*[7] = [010111; 1] o [010111; 1] = [010111; 2]

// [010100; 3] ⊆ [010100; 4]

// [000100; 3], [000100; 5] ⊆ [000100; 6]

// [010111; 1] ⊆ [010111; 2]

P^* = {[110100; 2]; [100000; 3]; [001100; 2]; [001000; 3]; [010100; 4]; [000100; 6]; [010111; 2]} (*Accumulated Frequent Patterns*)

* Finding maximal set:

Initialize $Q = \varnothing$.

P^*[6] = [000100; 6] has highest frequency. Remove all elements in P^* and S which cover bit-chain 000100 of P^*[6]. Put it into Q:

Q = {000100};

P^* = {[100000; 3]; [001000; 3]};

$S = \{100000; 001000\}$

Now, $P^*[1] = [100000; 3]$ and $P^*[2] = [001000; 3]$ have the same frequencies. It allows us to select one of them as the next maximal form in maximal set. Select $P^*[1]$. Remove all elements in P^* and S which cover bit-chain 100000 of $P^*[1]$. Put it into Q:

$Q = \{000100; 100000\}$;

$P^* = \{[001000; 3]\}$;

$S = \{001000\}$

Finally, P^* has just only one element $P^*[1] = [001000; 3]$. Remove all elements in P^* and S which cover bit-chain 001000 of $P^*[1]$. Put it into Q:

$Q = \{000100; 100000; 001000\}$;

$P^* = \varnothing$;

$S = \varnothing$

S is empty. The algorithm is terminated. Q is the maximal set of the set S.

$Q = \{000100; 100000; 001000\}$;

$(000100) \rightarrow d$; $(100000) \rightarrow a$; and $(001000) \rightarrow c$

So, minimal function $f = d \wedge a \wedge c$

In conclusion, $(d \wedge a \wedge c)$ is a reduction of discernibility function f.

VIII. EXPERIMENTATION 2

The experiments of proposed algorithms are conducted on a machine with Pentium(R) Dual-Core CPU, E6500 @ 2.93GHz (2 CPUs), ~2.1GHz and 2048MB main memory installed. The operating system is Windows Server 2008 R2 Enterprise 64-bit (6.1, Build 7601) Service Pack 1. Programming language is C#.NET.

Data for experiments are DataFoodMart 2000 and T40I10D100K taken from http://fimi.ua.ac.be/data/ and http://www.dagira.com/2009/12/23/foodmart-2000-universe-review-part-i-introduction website, respectively.

IX. CONCLUSION AND FUTURE WORK

The result of experimentation 1 reflects the efficiency and accuracy of *FIND_MaximalRandomPriorSet* algorithm. The complexity of this algorithm is $n.2^m$ where n is the number of bit-chains in the set S and m is the length of a bit-chain. In fact, m is often unchanged, so that, 2^m can be treated as a large constant and the complexity of *FIND_MaximalRandomPriorSet* algorithm is linear.

Mining frequent patterns is applied successfully into attribute reduction problem. *FIND_MaximalSet* algorithm maybe takes much time to execute but the result reflects the global optimization.

TABLE IV. THE RESULT FOR RUNNING THE ALGORITHMS

Data	No. of Records	No. of Attributes	AFP running time (second)	No. of FP	AR running time (second)	No. of Remaining Attributes
Food Mart 2000	10,281	18	80.979	7,184	0.140	12
T40I10 D100K	1,452,990	60	74,882.424	558,193	345.774	60

[a.] AFP: Accumulating Frequent Patterns

[b.] FP: Frequent Patterns

[c.] AR: Attribute Reduction

In future, paralleling the algorithms will be a good approach to reduce the calculation time and enhance the attribute reduction result.

Besides, integrating maximal random prior set and maximal set into practical applications will help verify their accuracy more clearly.

REFERENCES

[1] K. Anitha, "Gene selection based on rough set: applications of rough set on computational biology," International Journal of Computing Algorithm Volume 01, Issue 02, December 2012.

[2] H. Arafat, R. M. Elawady, S. Barakat, and N. M. Elrashidy, "Using rough set and ant colony optimization in feature selection," International Journal of Emerging Trends & Technology in Computer Science (IJETTCS), Volume 2, Issue 1, January - February 2013.

[3] B. Azhagusundari and A. S. Thanamani, "Feature selection based on information gain," International Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN: 2278-3075, Volume-2, Issue-2, January 2013.

[4] D. Chen, L. Zhang, S. Zhao, Q. Hu, and P. Zhu, "A novel algorithm for finding reducts with fuzzy rough sets," IEEE Transactions On Fuzzy Systems, Vol.20, No.2, April 2012.

[5] Rajashree Dash, Rasmita Dash, and D. Mishra, "A hybridized rough-PCA approach of attribute reduction for high dimensional data set," European Journal of Scientific Research ISSN 1450-216X Vol.44 No.1 (2010), pp.29-38.

[6] W. Ding, J. Wang, and Z. Guan, "Cooperative extended rough attribute reduction algorithm based on improved PSO," Journal of Systems Engineering and Electronics Vol.23, No.1, February 2012, pp.160–166.

[7] H. Fan and Y. Zhong, "A rough set approach to feature selection based on wasp swarm optimization," Journal of Computational Information Systems 8: 3 (2012) 1037–1045.

[8] Q. Hu, D. Yu, J. Liu, and C. Wu, "Neighborhood rough set based heterogeneous feature subset selection," Elsevier, Information Sciences 178 (2008) 3577-3594.

[9] N. S. Jaddi and S. Abdullah, "Nonlinear great deluge algorithm for rough set attribute reduction," Journal Of Information Science And Engineering 29, 49-62 (2013).

[10] Y. Jilin, Q. Keyun, and D. Weifeng, "Attribute reduction based on generalized similarity relation in incomplete decision system," Proceedings of the 2009 International Symposium on Information Processing (ISIP'09).

[11] L. Ju, X. Wenbin, and Z. Bei, "Construction of customer classification model based on inconsistent decision table," International Journal of e-Education, e-Business, e-Management and e-Learning, Vol.1, No.3, August 2011.

[12] B. Li, P. Tang, and T. W. S. Chow, "Quantization of rough set based attribute reduction," A Journal of Software Engineering and Applications, 2012, 5, 117-123.

[13] D. Li, Z. Chen, and J. Yin, "A new attribute reduction recursive algorithm based on granular computing," Journal Of Computers, Vol.8, No.3, March 2013.

[14] J. Liang, J. Mi, W. Wei, and F. Wang, "An accelerator for attribute reduction based on perspective of objects and attributes," Knowledge-Based Systems 9 February 2013.

[15] J. Liu, F. Min, S. Liao, and W. Zhu, "Test cost constraint attribute reduction through a genetic approach," Journal of Information & Computational Science 10: 3 (2013) 839–849.

[16] T. T. Nguyen and P. K. Nguyen, "A new viewpoint for mining frequent patterns," International Journal of Advanced Computer Science and Application (IJACSA), Vol.4, No.3, March 2013.

[17] S. K. Pati and A. K. Das, "Constructing minimal spanning tree based on rough set theory for gene selection," International Journal of Artificial Intelligence & Applications (IJAIA), Vol.4, No.1, January 2013.

[18] Z. Pawlak, "Rough sets," The Tarragona University seminar on Formal Languages and Rough Sets in August 2003.

[19] J. Qian, D. Q. Miao, Z. H. Zhang, and W. Li, "Hybrid approaches to attribute reduction based on indiscernibility and discernibility relation," Elsevier, International Journal of Approximate Reasoning 52 (2011) 212- 230.

[20] Y. Qian, J. Liang, W. Pedrycz, and C. Dang, "Positive approximation: an accelerator for attribute reduction in rough set theory," Elsevier, Artificial Intelligence 174 (2010) 597-618.

[21] T. Qiu, Y. Lin, X. Bai, "The difference degree of condition attributes and its application in the reduction of attributes," Journal Of Computers, Vol.7, No.5, May 2012.

[22] G. S. Sadasivam, S. Sangeetha, and K. S. Priya, "Privacy preservation with attribute reduction in quantitative association rules using PSO and DSR," Special Issue of International Journal of Computer Application (0975-8887) on Information Processing and Remote Computing – IPRC, August 2012.

[23] S. Sengupta and A. K. Das, "Single reduct generation based on relative indiscernibility of rough set theory," International Journal on Soft Computing (IJSC) Vol.3, No.1, February 2012.

[24] L. Sun, J. Xu, J. Ren, T. Xu, and Q. Zhang, "Granularity partition-based feature selection and its application in decision systems," Journal of Information & Computational Science 9: 12 (2012) 3487–3500.

[25] K. S. Tiwari, A. G. Kothari, and A. G. Keskar, "Reduct generation from binary discernibility matrix: an hardware approach," International Journal of Future Computer and Communication, Vol.1, No.3, October 2012.

[26] E. Xu, Y. Yang, and Y. Ren, "A new method of attribute reduction based on information quantity in an incomplete system," Journal Of Software, Vol.7, No.8, August 2012.

[27] Y. Yao, and Y. Zhao, "Attribute reduction in decision - theoretic rough set models," Information Sciences, 178(17), 3356-3373, Elsevier B.V., 2008.

[28] Y. Yao and Y. Zhao, "Discernibility matrix simplification for constructing attribute reducts," Information Sciences, Vol.179, No.5, 867-882, 2009.

[29] D. Yue, W. Jian, and Z. Xu, "Attribute reduction algorithm based on incomplete decision table," National Conference on Information Technology and Computer Science (CITCS 2012).

[30] Y. Zhai, C. Zhou, and Y. Sun, "Approach of rule extracting based on attribute significance and decision classification," Journal Of Computers, Vol.7, No.2, February 2012.

[31] Q. Zhang, W. Shen, and Q. Yang, "Attribute reduction in inconsistent disjunctive set-valued ordered decision information system," Journal of Computational Information Systems 8: 12 (2012) 5161-5167.

[32] J. Zheng and R. Yan, "Attribute reduction based on cross entropy in rough set theory," Journal of Information & Computational Science 9: 3 (2012).

[33] W. Zhengjiang, Z. Jingmin, and G. Yan, "An attribute reduction algorithm based on genetic algorithm and discernibility matrix," Journal Of Software, Vol.7, No.11, November 2012.

[34] F. Wang, J. Liang, C. Dang, and Y. Qian, "Attribute reduction for dynamic data sets," Journal Applied Soft Computing Volume 13, Issue 1, January 2013, Pages 676-689.

[35] W. Wei, J. Liang, J. Wang, and Y. Qian, "Decision-relative discernibility matrixes in the sense of entropies," International Journal of General Systems 2013.