Structure of Association Rule Set Based on Min-Min Basic Rules

Conference Paper · November 2010

DOI: 10.1109/RIVF.2010.5633246

CITATIONS
6

READS
25

3 authors:

Tin C Truong
University of Dalat
21 PUBLICATIONS **117** CITATIONS
SEE PROFILE

Anh Tran
University of Dalat
11 PUBLICATIONS **65** CITATIONS
SEE PROFILE

Thong Tran
University of Dalat
4 PUBLICATIONS **9** CITATIONS
SEE PROFILE

Structure of Association Rule Set based on Min-Min Basic Rules

Tin C. Truong
Department of Mathematics and Computer Science,
University of Dalat, Dalat, Vietnam
tintc@dlu.edu.vn

Anh N. Tran
Department of Mathematics and Computer Science,
University of Dalat, Dalat, Vietnam
anhtn@dlu.edu.vn

Thong Tran
Department of Information Technology,
University of Dalat, Dalat, Vietnam
thongt@dlu.edu.vn

***Abstract*— In this paper, we partition the association rule set into disjoint equivalence rule classes. Each of them contains rules having the same confidence and then it is split into basic and consequence rule sets based on the order relation on it. Basic rule set, which includes minimal elements according to this relation, is directly found by our algorithm *MG_BARS*. In addition, by adding appropriate eliminable itemsets to both sides of basic rules in our algorithm *MG_CARS*, the consequence rules are completely and non-repeatedly generated and are confidence-preserved. Results of the experiments proved the efficiency of the above algorithms.**

Association rule; basic rule; consequence rule; generator; eliminable itemset.

I. INTRODUCTION

The number of association rules found in knowledge discovery problems in data mining is usually enormous [1]. Hence, understanding the structure of the association rule set enables us to generate various efficient algorithms to find out the rules. Recently, researchers often split the association rule set into sets of basis and consequent. Taouil et al. [4] proposed a basis for association rules. In [3], based on the concept of minimal rules, Pasquier et al. considered basic rules as min-max form (i.e. left-hand side of the basic rule is minimal and its right-hand side is maximal). Based on the concept of the most general rules, Zaki [6], considered basic rules as min-min form (i.e. both their left-hand and right-hand sides of the basic rules are minimal). However, in order to find out basic rules, Zaki's algorithm used many candidates. Moreover, he did not figure out an algorithm that can generate consequence rules from basic rules, whereas the consequence rules together with their support and confidence are essential for users.

In [5], we proposed algorithms to completely and quickly find basic rules as min-max form, and thus were able to find their consequence rules. In this paper, we explicitly show the structure of the association rule set by using the concept of basic rules as min-min form. Based on an equivalence relation, the association rule set will be partitioned into disjoint equivalence rule classes. As the result, we need to investigate only the structure of each equivalence rule class independently. Due to this partition, efficient parallel algorithms for mining association rules can be easily obtained. After proposing an order relation between rules belonging to the same equivalence rule class, we define the basic rules as minimal elements according to this relation. The algorithm *MG_BARS* is indicated to quickly and directly find basic rules. Then with our algorithm *MG_CARS*, by adding eliminable itemsets to both sides of the basic rules, consequence rules are generated. Both the basic rules and consequence rules are in the same equivalence class. Hence, the consequence rules are confidence-preserved, non-repeated. Moreover, they are totally different from all rules in other equivalence classes. This efficient algorithm helps to reduce a considerable amount of time for mining consequence rules.

The rest of the paper is organized as follows. Section II recalls some primitive concepts and results of closed and eliminable itemsets in the association rule mining problem. Section III presents the structure of the association rule set based on the concept of basic rules as min-min form. It also indicates efficient algorithms for finding rule sets of basis and consequent. Sections IV and V show the experimental results and conclusions.

II. PRIMITIVE CONCEPTS AND RESULTS

A. Primitive concepts

Given set $\mathcal{O}$ contained transactions and $\mathcal{A}$ contained items related to each of transaction $o \in \mathcal{O}$ and $\mathcal{R}$ is a binary relation in $\mathcal{O} \times \mathcal{A}$. Now, consider two set functions: $\lambda: 2^{\mathcal{O}} \to 2^{\mathcal{A}}$, $\rho: 2^{\mathcal{A}} \to 2^{\mathcal{O}}$ determined in the following: $\forall A \subseteq \mathcal{A}$, $O \subseteq \mathcal{O}$: $\lambda(O) = \{a \in \mathcal{A} \mid (o, a) \in \mathcal{R}, \forall o \in O\}$, $\rho(A) = \{o \in \mathcal{O} \mid (o, a) \in \mathcal{R}, \forall a \in A\}$. Defining the set function h in $2^{\mathcal{A}}$ by: $h = \lambda \circ \rho$, we say that h(A) are the closure of A. An itemset $A \subseteq \mathcal{A}$ is a closed itemset if h(A)=A [2].

Let s_0 be the minimum support, $s_0 \in [0; 1]$. The support of an itemset $A \subseteq \mathcal{A}$ is defined as $s(A) = |\rho(A)|/|\mathcal{O}|$. If $s(A) \geq s_0$ then A is frequent itemset [1]. Let $\mathcal{CS}$ be the class of all closed frequent itemsets.

Let c_0 be the minimum confidence, $c_0 \in (0; 1]$. For any frequent itemset S (with threshold s_0), we take a non-empty, strict subset L from S ($\emptyset \neq L \subset S$), R = S\L. Let r: L→R denote the rule created by L, R (or by L, S). Then, s(r) =

978-1-4244-8075-3/10/$26.00 ©2010 IEEE

s(S), c(r) = |ρ(S)|/|ρ(L)| = s(S)/s(L) are the support and confidence of r respectively. The rule r is an association rule if c(r) ≥ c_0 [1]. Let $\mathcal{AR}\_\mathcal{S} \equiv \mathcal{AR}\_\mathcal{S}(s_0, c_0)$ be the set of all association rules with thresholds s_0, c_0. Briefly, we call the association rules simply rules. For two non-empty itemsets G, A: G⊆A⊆$\mathcal{A}$, G is called a generator of A if h(G)=h(A) and (∀G' ⊂ G ⇒ h(G') ⊂ h(G)) [3]. Let $\mathcal{G}$(A) be the class of all generators of A.

In class $2^{\mathcal{A}}$, an itemset R is called eliminable in S if R⊂S and ρ(S) = ρ(S\R). Let $\mathcal{N}$(S) denote the class of all *eliminable itemsets* in S, $\mathcal{N}^*$(S) := $\mathcal{N}$(S) \ {∅}, we have [5]:

$$\mathcal{N}(S) = \{A: A \subseteq S \backslash G_0,\ G_0 \in \mathcal{G}(S)\}.$$

B. *Primitive results*

Closed mapping h: $2^{\mathcal{A}} \to 2^{\mathcal{A}}$ generates a binary relation $\sim_h$ in class $2^{\mathcal{A}}$: ∀ A, B ⊆ $\mathcal{A}$:

$$A \sim_h B \Leftrightarrow h(A) = h(B).$$

We see that $\sim_h$ is an equivalence relation. It partitions $2^{\mathcal{A}}$ into the disjoint equivalence classes [5]. The supports of all itemsets in each equivalence class are the same. The equivalence class containing A is denoted as [A].

Theorem 1 (Presentation of itemsets [5]). For every itemset A such that ∅≠A∈ $\mathcal{CS}$, we have:

$$X \in [A] \Leftrightarrow \exists G_0 \in \mathcal{G}(A),\ \exists X' \in \mathcal{N}(A): X = G_0 + X'.$$ [1]

The above presentation of itemsets is not unique because each itemset can have various generators. Hence, when we mention that R' is eliminable in S, we have to assign R' to a particular generator of S.

Proposition 1 (Support-preserved role of eliminable itemset). Adding an eliminable set R' of S to R (R∈[S]) will not change its closure, thus will not change its support.

Proof: h(R) = h(R∪R') = h(S) ⇔ ρ(R) = ρ(R∪R') = ρ(S) ⇔ s(R∪R') = s(R).

III. Structure of Association Rule Set Generated from Min-Min Basic Rules

In this section, we will partition the association rule set into disjoint equivalence rule classes using an equivalence relation based on the closures of left-hand side and also of both sides of association rules. Next, we construct an order relation to indicate how to find min-min basic rules by our algorithm *MG_BARS*. Then we suggest the way to derive confidence-preserved consequence rules which belong to the same equivalence rule class as basic rules. Finally, the efficient algorithm *MG_CARS* to generate non-repeated consequence rules is proposed.

A. *Partitioning association rule set based on equivalence relation*

[1] The symbol + is denoted as the union of two disjoint sets.

For a brief notation, let (L, S) denote two closed frequent itemsets: L, S ∈ $\mathcal{CS}$, ∅≠L⊆S. In [6] an equivalence relation was used for partitioning the association rule set into equivalence rule classes. Each of them, based on this relation, contains all rules which have the same support and confidence. Using the equivalence relation in definition 1, the following theorem 2 will show a smoother partition of the rule set $\mathcal{AR}\_\mathcal{S}$.

Definition 1. (Equivalence relation on association rule set). Let $\sim_r$ be a binary relation in $\mathcal{AR}\_\mathcal{S}$ determined as follows: ∀ L', S', Ls, Ss ⊆ $\mathcal{A}$, ∅≠L'⊂S', ∅≠Ls⊂Ss, r:L'→S'\L', s:Ls→Ss\Ls:

$$s \sim_r r \Leftrightarrow (Ls \in [L'] \text{ and } Ss \in [S']).$$

From the above definition, we have theorem 2. Although this theorem is easy to prove, it plays an important role in partitioning the association rule set.

Theorem 2 (*Partition of the association rule set*). Relation $\sim_r$ is an equivalence relation. It partitions $\mathcal{AR}\_\mathcal{S}$ into disjoint equivalence rule classes $\mathcal{AR}\_\mathcal{S}$(L, S). For each class, we usually consider the representative rule r_0:$G_L \to S \backslash G_L$, $G_L \in \mathcal{G}(L)$. The rules in $\mathcal{AR}\_\mathcal{S}$(L, S) have the same support $s(r_0)$ and confidence $c(r_0)$ (the reverse is not always true).

$$\mathcal{AR}\_\mathcal{S} = \sum_{(L,S)} AR\_S(L,S).$$

Proof: Consider the rule r:L'→R', L' ∈ [L], L'+R' ∈ [S]. Since L' and G_L belong to [L] so s(G_L) = s(L'); s(r_0) = |ρ(G_L)∩ρ(S\G_L))| = |ρ(S)| = |ρ(L'∪R')| = |ρ(L')∩ρ(R'))| = s(r), hence, c(r_0) = c(r).

To investigate the structure of $\mathcal{AR}\_\mathcal{S}$ thoroughly, we need to investigate only the structure of each equivalence rule class $\mathcal{AR}\_\mathcal{S}$(L, S) independently. Moreover, from this partition, the parallel algorithms are easily generated to quickly find association rules. This is the significance of the equivalence relations and is a typical instance of the divide-and-conquer method widely used in computer science.

Since the size of the paper is limited, we prove only some results related to equivalence rule class $\mathcal{AR}\_\mathcal{S}$(L, S) with L ⊂ S. With L = S, the similar results are easily proved.

B. *Basic rules as min-min form*

In this section, an order relation in each equivalence rule class is obtained to propose basic rule set as min-min basis. Then, the algorithm for finding it is also indicated.

Definition 2. Consider a partial order relation ≤ on $\mathcal{AR}\_\mathcal{S}$(L, S), defined by: ∀r_j: $L_j \to R_j$ ∈ $\mathcal{AR}\_\mathcal{S}$(L, S), $S_j = L_j + R_j$, j=1,2:

$$r_1 \le r_2 \Leftrightarrow (L_1 \subseteq L_2 \text{ and } R_1 \subseteq R_2).$$

Basic rule set $\mathcal{B}$(L, S) of $\mathcal{AR}\_\mathcal{S}$(L, S) contains all minimal rules based on the relation ≤:

$$\mathcal{B}(L, S) = \{r_0: L_i \to R_i^* \mid L_i \in \mathcal{G}(L),\ R_i^* \in R_{min}(L_i, S)\},$$

where $R_{min}(L_i, S)$ is the set containing all minimal elements of $\{S_k \backslash L_i \mid S_k \backslash L_i \neq \emptyset,\ S_k \in \mathcal{G}(S)\}$ according to the normal

order relation "$\subseteq$" in the set theory and $c(r_0) < 1$, $\forall\ r_0 \in \mathcal{B}(L, S)$.

Similarly, $\forall\ L \in \mathcal{CS}: L \notin \mathcal{G}(L)$,

$$\mathcal{B}(L, L) = \{r_0: L_0 \rightarrow \{a\}, L_0 \in \mathcal{G}(L),\ \forall a \in L \backslash L_0\}$$

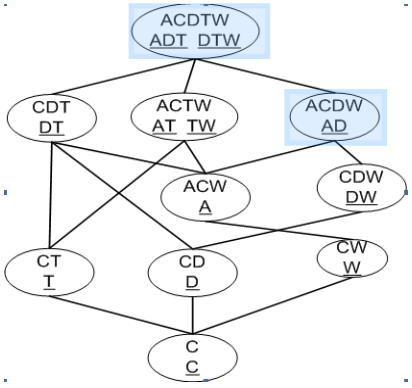
and $c(r_0) = 1$, $\forall\ r_0 \in \mathcal{B}(L, L)$.

Both sides of all basic rules which Zaki considered the most general rules are minimal. In [6], he introduced the algorithm *GenerateRules* to find these rules. However, his algorithm used many candidates (for the most general rules with the confidence equal to 1) and operations for finding the closure of itemsets to check unnecessary conditions. Based on definition 2, our below algorithm *MG_BARS* is better. The results of experiments will prove that.

$\mathcal{B}(L, S)$ *MG_BARS* (L, S)
(1) $\mathcal{B}(L, S) := \emptyset$; $c := |\rho(S)| / |\rho(L)|$;
(2) for each ($L_i \in \mathcal{G}(L)$) do
(3) MS := $R_{min}(L_i, S)$;
(4) for each ($R_k \in$ MS) do
(5) $\mathcal{B}(L, S) := \mathcal{B}(L, S) \cup \{<L_i \rightarrow R_k, s(S), c>\}$;
(6) return $\mathcal{B}(L, S)$;
where $\mathcal{G}(L)$ is found by the algorithm *MinimalGenerators* [6].

Trans	Items
1	ACTW
2	CDW
3	ACTW
4	ACDW
5	ACDTW
6	CDT

(a) Database 1



(b) Frequent closed itemset lattice

Figure 1. Database 1 and Frequent closed itemset lattice

Example 1. Consider database 1 and the corresponding frequent closed itemset lattice in figure 1 (minimum support $s_0 = 0$), where underlined itemsets are generators of corresponding closed itemsets. Consider the pair of closed itemsets (L, S), where S=ACDTW[2] having $\mathcal{G}(S)$={ADT, DTW} and L=ACDW having $\mathcal{G}(L)$={AD}. We have a basic rule AD$\rightarrow$T since Minimal{ADT\AD=T, DTW\AD=TW} = {T}.

From basic rules in $\mathcal{B}(L, S)$, based on eliminable itemsets related to them, we will propose efficient methods for non-repeatedly deriving all confidence-preserved consequence rules of $\mathcal{C}(L, S) := \mathcal{AR\_S}(L, S) \setminus \mathcal{B}(L, S)$.

[2] $\{a_1, a_2, \ldots, a_n\}$ is abbreviated that $a_1a_2\ldots a_n$

C. *Complete derivation of confidence-preserved consequence rules from basic ones by adding eliminable itemsets.*

From proposition 1, we see that adding appropriate eliminable itemset to both sides of a rule will not change the support of that rule. Hence, its confidence is preserved. This is the key point of the methods to generate confidence-preserved consequence rules.

Definition 3. Consider two set functions that generate consequence rules F_{+L}, F_{+R}: $\mathcal{AR\_S}(L, S) \rightarrow 2^{\mathcal{AR\_S}(L, S)}$, $\forall r_0: L_0 \rightarrow R_0 \in \mathcal{AR\_S}(L, S)$,

$$F_{+L}(r_0) = \{r_{+L}: L_0 + L' \rightarrow R_0 \mid L' \in \mathcal{N}^*(L),\ L' \subseteq L \backslash (L_0 + R_0)\}.$$
$$F_{+R}(r_0) = \{r_{+R}: L_0 \rightarrow R_0 + R \mid R \in \mathcal{N}^*(S),\ R \subseteq S \backslash (L_0 + R_0)\}.$$

Proposition 2. Obviously, $F_{+L}(r_0) = \{r_{+L}: L_0 + L' \rightarrow R_0 \mid \emptyset \neq L' \subseteq L \backslash (L_0 + R_0)\}$ and $F_{+R}(r_0) = \{r_{+R}: L_0 \rightarrow R_0 + R \mid \emptyset \neq R \subseteq S \backslash (L_0 + R_0)\}$. The above definition is correct.

Proof: Since $r_0: L_0 \rightarrow R_0 \in \mathcal{AR\_S}(L, S)$, so $h(L_0) = L \subseteq h(L_0 + R_0) = S$. *(a)* If $\emptyset \neq L' \subseteq L \backslash (L_0 + R_0)$ then $L_0 \subseteq L_0 + L' \subseteq L$, $L_0 + R_0 \subseteq L_0 + L' + R_0 \subseteq L + R_0 \subseteq S$. Hence, $L = h(L_0 + L')$ and $h(L_0 + L' + R) = S$, i.e., $r_{+L} \in \mathcal{AR\_S}(L, S)$. *(b)* If $\emptyset \neq R \subseteq S \backslash (L_0 + R_0)$ then $L_0 + R_0 \subseteq L_0 + R_0 + R \subseteq S$, $h(L_0 + R_0 + R) = S$, i.e., $r_{+R} \in \mathcal{AR\_S}(L, S)$.

Consider equivalence rule class $\mathcal{AR\_S}(L, S)$. With $r \in \mathcal{AR\_S}(L, S)$, the rule sets $\{r\}$, $F_{+L}(r)$ and $F_{+R}(r)$ are disjoint because either the left-hand side or right-hand side of any rules in them is different from the other rules. From each basic rule $r_0 \in \mathcal{AR\_S}(L, S)$, the rules in each set $F_{+L}(r_0)$, $F_{+R}(r_0)$ are non-repeatedly generated and have the same confidence as r_0 since they all belong to $\mathcal{AR\_S}(L, S)$. Moreover, we have:

$$F_{+L}(\mathcal{B}(L, S)) = \{r_{+L}: L_i + L' \rightarrow R^*_{ik} \mid L_i \in \mathcal{G}(L),\ R^*_{ik} \in R_{min}(L_i, S),\ \emptyset \neq L' \subseteq L \backslash (L_i \cup R^*_{ik})\},$$

$$F_{+R}[\mathcal{B}(L,S) + F_{+L}(\mathcal{B}(L,S))] = \{r_{+R}: L_i + L'' \rightarrow R^*_{ik} + R' \mid L_i \in \mathcal{G}(L),\ L'' \subseteq L \backslash (L_i \cup R^*_{ik}),\ R^*_{ik} \in R_{min}(L_i, S),\ \emptyset \neq R' \subseteq S \backslash (L_i + L'' + R^*_{ik})\}.$$

Proposition 3. (Disjoint splitting of each equivalence rule class). $\mathcal{AR\_S}(L, S) = \mathcal{B}(L,S) + F_{+L}(\mathcal{B}(L,S)) + F_{+R}[\mathcal{B}(L,S) + F_{+L}(\mathcal{B}(L,S))]$.

Proof: Since $F_{+L}(\mathcal{B}(L, S)) + F_{+R}[\mathcal{B}(L,S) + F_{+L}(\mathcal{B}(L,S))] \subseteq \mathcal{C}(L, S)$, so we need to prove only $\mathcal{AR\_S}(L, S) \subseteq \mathcal{B}(L,S) + F_{+L}(\mathcal{B}(L,S)) + F_{+R}[\mathcal{B}(L,S) + F_{+L}(\mathcal{B}(L,S))]$. For every $r: L' \rightarrow R \in \mathcal{AR\_S}(L, S)$, since $h(L') = L$, $L' \cap R = \emptyset$ and $h(S') = S$, where $S' = L' + R$, so there exists $L_i \in \mathcal{G}(L')$, $S_k \in \mathcal{G}(S')$: $L_i \subset S_k$ and $S_k \backslash L_i \subseteq R$. Indeed, if there exists $a \in (S_k \backslash L_i) \backslash R$ then $a \in L'$ and $\{a\} + L_i$ is subset of S_k, so $\{a\} + L_i$ is also generator and $\{a\} + L_i \in \mathcal{G}(L')$: it contradicts with $L_i \in \mathcal{G}(L')$. With $S'' := S' \backslash S_k$, we have $S' = S_k + S'' = L_i + (S_k \backslash L_i) + (S' \backslash (L_i \cup S_k)) = L_i + (S_k \backslash L_i) + (L' \backslash (L_i \cup S_k)) + (R \backslash (L_i \cup S_k)) = L_i + (L' \backslash (L_i \cup S_k)) + (S_k \backslash L_i) + + (R \backslash (L_i \cup S_k)) = L_i + (L' \backslash L_i) +$

$(S_k\backslash L_i)+(R\backslash S_k)$, since $L'\cap R=\emptyset$ and $S_k\backslash L_i\subseteq R$ so $(L'\backslash(L_i\cup S_k)) = L'\backslash(L_i+(S_k\backslash L_i)) = L'\backslash L_i$ and $R\backslash(L_i\cup S_k)=(R\backslash S_k)\backslash L_i=R\backslash S_k)$. Thus, $L'=L_i+(L'\backslash L_i)$ and $R=(S_k\backslash L_i)+(R\backslash S_k)$. Let ik be the minimum index of S_k such that $R^*_{ik}=S_{ik}\backslash L_i$ is minimal. Then $L'=L_i+L''$, $R=R^*_{ik}+R'$, where $L''=(L'\backslash L_i)$ and $R'=R\backslash S_{ik}$. Otherwise, with $K_{U,L'} := \bigcup_{L_i\in G(L')} L_i$, we have $L'' = (L'\backslash L_i)\cap K_{U,L'} + L'\backslash(L_i\cup K_{U,L'}) = L'_i+L^{\sim}\subseteq L'\backslash(L_i\cup R^*_{ik})\subseteq L\backslash(L_i\cup R^*_{ik})$, where $L'_i=K_{U,L'}\backslash L_i$, $L^{\sim}=L'\backslash K_{U,L'}= L'\backslash(K_{U,L'}\cup R_{ik})$. Similarly, with $S_{U,L'+R}= \bigcup_{S_k\in G(L'+R)} S_k$, we have $R' = R\backslash S_{ik} = (R\backslash S_{ik})\cap S_{U,L'+R} + R\backslash(S_{ik}\cup S_{U,L'+R}) = S_{U,L'+R,ik}+ S^{\sim}\subseteq R\backslash(L'\cup S_{ik})\subseteq S\backslash(L'+R^*_{ik})$, where $S_{U,L'+R,ik}=(R\cap S_{U,L'+R})\backslash S_{ik}$, $S^{\sim}=R\backslash S_{U,L'+R}$. Then $r:L_i+L'_i+L^{\sim}\to R^*_{ik}+S_{U,L'+R,ik}+S^{\sim}$. Let us call $r_0:L_i\to R^*_{ik}\in \mathcal{B}(L,S)$, $r_{+L}:L_i+L'_i+L^{\sim}\to R^*_{ik}$, and $r_{+R}:L_i+L'_i+L^{\sim}\to R^*_{ik}+ S_{U,L'+R,ik}+S^{\sim}$. We have three following cases: *(a)* if $S_{U,L'+R,ik}+S^{\sim} = \emptyset$, $L'_i+L^{\sim} = \emptyset$: then $r\equiv r_0:L_i\to R^*_{ik}\in \mathcal{B}(L,S)$, *(b)* if $S_{U,L'+R,ik}+S^{\sim} = \emptyset$, $L'_i+L^{\sim} \neq \emptyset$: then $r\equiv r_{+L}\in F_{+L}(\mathcal{B}(L,S))$, *(c)* if $S_{U,L'+R,ik}+S^{\sim} \neq \emptyset$: then $r\equiv r_{+R}\in F_{+R}[\mathcal{B}(L,S)+F_{+L}(\mathcal{B}(L,S))]$.

Based on proposition 3, each equivalence rule class $\mathcal{AR}\_\mathcal{S}(L,S)$ is split into two disjoint rule sets of basis and consequent. From basic rules, the consequence rules are completely generated and are totally different from the rules of all other equivalence rule classes. However, in each equivalence class the consequence rules generated by the functions F_{+L}, F_{+R} can be repeated. For example, with r_{01} and r_{02} different but having the same right-hand sides, two rule sets $F_{+L}(r_{01})$ and $F_{+L}(r_{02})$ can have some identical rules, i.e., $F_{+L}(r_{01})\cap F_{+L}(r_{02}) \neq \emptyset$.

Example 2. Using the figure 1, we consider S=ADTWC having $\mathcal{G}(S)$={ADT, TDW} with L'=ACTW having $\mathcal{G}(L')$={AT, TW}. Since $W\subseteq L'\backslash(AT+D)=CW$ so the basic rule r_1:AT→D derives the consequence rule r_{1+L}:ATW→D. The basic rule r_2:TW→D also derives r_{1+L}. Then, S is considered with L''=CD where $\mathcal{G}(L'')$={D}. The consequence rule r_{3+R}:D→ATW derived from the basic rule r_3:D→AT (since $W\subseteq S\backslash(D+AT)=CW$) is the same of one consequence rule of the basic rule r_4:D→TW.

D. Complete derivation of the non-repeated, confidence preserved consequence rules from basic rules

This section proposes different forms of the functions F_{+L}, F_{+R}. These forms will non-repeatedly generate all consequence rules in each equivalence rule class. The corresponding algorithms will be shown.

1) The non-repeated form F'_{+L} of F_{+L}

For every $L\in \mathcal{CS}$, $L_i\in \mathcal{G}(L)$, $R\subseteq\mathcal{A}$, $R\cap L_i=\emptyset$, denote $K_U = \bigcup_{L_i\in G(L)} L_i$, $K^{\sim}_R = L\backslash(K_U\cup R)$, $K_{U,Li} = K_U\backslash L_i$, $\mathcal{FS}\_(L_i, R) = \{L_i+L^{\sim} \mid L^{\sim}\subseteq K^{\sim}_R\}$, we define:

$\mathcal{FS}1(L_i, S) \equiv \{L_i+L'_i+L^{\sim} \mid R^*\in R_{min}(L_i, S), L_i+L^{\sim}\in \mathcal{FS}\_(L_i, R^*), L'_i\subseteq K_{U,Li}, i=1 \text{ or } (i>1: \text{not}(L_k\subseteq(L_i+L'_i)\backslash R^*), \forall 1\le k<i)\}$

and for every $L^*\subseteq L\backslash L_i$,

$\mathcal{FS}2(L_i, L^*) \equiv \{(L_i+L^{\sim}+L'_i)\backslash L^* \mid L_i+L^{\sim}\in \mathcal{FS}\_(L_i, L^*), |L'_i+L^{\sim}|>1, |L^*|\ge 1, L'_i\subseteq K_{U,Li}, i=1 \text{ or } (i>1: \text{not}(L_k\subseteq(L_i+L'_i)\backslash L^*), \forall 1\le k<i)\}$.

Definition 4. Consider $r_1:L_i\to R\in \mathcal{B}(L, S)$ and $r_2:L_i\to\{a\}\in \mathcal{B}(L, L)$. We define

$F'_{+L}(r_1) \equiv \{r_{1+L}:L'\to R \mid L'\in \mathcal{FS}1(L_i, S)\backslash\{L_i\}\}$,

$F'_{+L}(L, S) \equiv \bigcup_{r_1\in B(L,S)} F'_{+L}(r_1)$

and $F'_{+L}(r_2) \equiv \{r_{2+L}:L'\to\{a\} \mid L'\in \mathcal{FS}2(L_i, \{a\})\}$,

$F'_{+L}(L, L) \equiv \bigcup_{r_2\in B(L,L)} F'_{+L}(r_2)$.

Proposition 4. Replace $F'_{+L}(\mathcal{B}(L, S))$ with $F'_{+L}(L, S)$, we have: (a) The consequence rules in $F'_{+L}(L, S)$ are non-repeatedly generated, (b) $F'_{+L}(L, S) = F_{+L}(\mathcal{B}(L, S))$.

Proof: *(a)* Assume that $\exists i1>i2$, $i1>1$: $L_{i1}+L'_{i1}+L^{\sim}_{i1} \equiv L_{i2}+L'_{i2}+L^{\sim}_{i2}$, $L_{ik}\in\mathcal{G}(L)$, $L^{\sim}_{ik}\subseteq K^{\sim}$, $L'_{ik}\subseteq K_{U,Lik}$, k=1,2. Since $L_{i2}\cap L^{\sim}_{i1}=\emptyset$, so $L_{i2}\subseteq L_{i1}+L'_{i1}$. It contradicts the way selected L'_{i1}.

(b) "⊆": For every $r:L_i\to R\in\mathcal{B}(L, S)$, consider r_{+L}: $L_i+L'_i+L^{\sim}\to R \in F'_{+L}(r)$. Since $\emptyset\neq L':=(L'_i+L^{\sim}) \subseteq K_{U,Li}+K^{\sim}_R \subseteq L\backslash(L_i\cup R)$ so $r_{+L}\in F_{+L}(r)\subseteq F_{+L}(\mathcal{B}(L, S))$. *"⊇":* $\forall r'\in F_{+L}(\mathcal{B}(L, S))$, from theorem 1, let i be the minimum index such that $r':L''\to R^*$, where $L''=L_i+O_i$, $L_i\in\mathcal{G}(L'')$, and $O_i\subseteq L\backslash(L_i\cup R^*)$. Let us call $L'_i=O_i\cap K_U\subseteq K_{U,Li}$, $L^{\sim}_{R^*}=O_i\backslash K_U\subseteq K^{\sim}_{R^*}$. We have $L''=L_i+L'_i+L^{\sim}_{R^*}$, and hence, $r':L_i+L'_i+L^{\sim}_{R^*}\to R^*$. Assume that there exists i>1 and k<i: $L_k\subseteq(L_i+L'_i)\backslash R^*$, $L_k\in\mathcal{G}(L)$, hence $L_k\in\mathcal{G}(L'')$. Then $L''=L_k+O_k$, where $O_k=L'_k+L^{\sim}_{R^*}$, $L'_k=(L_i+L'_i)\backslash L_k\subseteq L\backslash(L_k\cup R^*)$ and $L^{\sim}_{R^*}\subseteq L\backslash(K_U\cup R^*)\subseteq L\backslash(L_k\cup R^*)$ so $O_k\subseteq L\backslash(L_k\cup R^*)$. It contradicts the way selected the i index! Therefore, $\text{not}(L_k\subseteq(L_i+L'_i)\backslash R^*), \forall k<i$, i.e., $r'\in F'_{+L}(L, S)$.

2) The non-repeated form F'_{+R} of F_{+R}

For every $L'\in \text{Left}(L, S) := \{L' \mid \exists r:L'\to R'\in[\mathcal{B}(L,S)+F'_{+L}(L,S)]\}$, we denote $R_{min}(L', S) = \{R^*_{ik}:=S_k\backslash L' \mid S_k\in\mathcal{G}(S)$, $S_k\backslash L_i$ is minimal for each $L_i\in\mathcal{G}(L')\}$, $S_{U,L'} := \bigcup_{R^*_i\in R_{min}(L',S)} R^*_i$, $S_{U,L',i} := S_{U,L'}\backslash R^*_i$, $S^{\sim}_{L'}:=S\backslash(S_{U,L'}+L')$ and $\text{Right}(L', S) := \{R^*_i+R'_i+R^{\sim} \mid R^*_i\in R_{min}(L', S), R'_i\subseteq S_{U,L',i}, R^{\sim}\subseteq S^{\sim}_{L'}, i=1 \text{ or } (i>1: \text{not } (R^*_k\subseteq R^*_i+R'_i), \forall 1\le k<i), |R'_i+R^{\sim}|>1\}$.

Definition 5.

$F'_{+R}(L, S) \equiv \sum_{L'\in Left(L,S)} \{r_{+R}:L'\to R'' \mid R''\in Right(L', S)\}$

and $F'_{+R}(L, L) \equiv \{r_{+R}:L'\to L^* \mid L_i\in\mathcal{G}(L), L^*\subseteq L\backslash L_i, |L^*|>1, L'\in\mathcal{FS}2(L_i, L^*)\}$.

Similarly to proposition 4, we have proposition 5.

Proposition 5. (a) The consequence rules in $F'_{+R}(L, S)$ are non-repeatedly generated,

(b) $F'_{+R}(L, S) = F_{+R}[\mathcal{B}(L,S)+F_{+L}(\mathcal{B}(L,S))]$.

Proof: *(a)* All rules in $F'_{+R}(L, S)$ have either the left-hand sides in Left(L, S) or the right-hand sides in Right(L', S) which are different. Hence, all consequences rules generated in $F'_{+R}(L, S)$ are non-repeatedly generated.

(b) Obviously by the definitions of $F'_{+R}(L, S)$ and $F_{+R}[\mathcal{B}(L,S)+F_{+L}(\mathcal{B}(L,S))]$.

3) Complete derivation of non-repeated and confidence-preversed consequence rules

Theorem 3 (Disjoint splitting of non-repeated rules in each equivalence class).

$$\mathcal{AR}\_\mathcal{S}(L, S) = \mathcal{B}(L, S) + F'_{+L}(L, S) + F'_{+R}(L, S).$$

Similarly, $\mathcal{AR}\_\mathcal{S}(L, L) = \mathcal{B}(L, L) + F'_{+L}(L, L) + F'_{+R}(L, L)$.

Proof: Consequence of propositions 3, 4 and 5.

From the above propositions, the algorithm *MG_CARS* is suggested for deriving all non-repeated consequence rules $\mathcal{C}(L, S)$ in every equivalence rule class $\mathcal{AR}\_\mathcal{S}(L, S)$.

$\mathcal{C}(L, S)$ *MG_CARS* (L, S)
(1) $F'_{+L}(L, S) :=$ *LeftAdding* (L, S); $F'_{+R}(L, S) := \emptyset$;
(2) for each (L'∈ Left(L, S)) do
(3) $F'_{+R}(L, S) := F'_{+R}(L, S) +$ *RightAdding*(L', S);
(4) $\mathcal{C}(L, S) := F'_{+L}(L, S) + F'_{+R}(L, S)$;
(5) return $\mathcal{C}(L, S)$;

The algorithm *RightAdding* for finding consequence rule subset $F'_{+R}(L', S)$ is indicated as follows (the algorithm *LeftAdding* can be derived in the same way).

$F'_{+R}(L', S)$ *RightAdding* (L', S)
(1) $F'_{+R}(L', S) := \emptyset$; MS := $R_{min}(L', S)$;
(2) $S_{U,L'} := \bigcup_{R_i \in MS} R_i$; $S^{\sim}_{L'} := S \backslash (K_{U,L'}+L')$;
(3) for each ($R^{\sim} \subseteq S^{\sim}_{L'}$) do
(4) for each ($R_i \in$ MS) do
(5) $S_{U,L',i} := S_{U,L'} \backslash R_i$;
(6) for each ($R'_i \subseteq S_{U,L',i}$) do
(7) if ($R'_i \neq \emptyset$ or $R^{\sim} \neq \emptyset$) then
(8) Repeated := false;
(9) if (i>1) then for each ($R_k \in$ MS | k<i) do
(10) if ($R_k \subseteq R_i+R'_i$) then
(11) Repeated := true;
(12) break; // for each R_k
(13) if (not(Repeated)) then
(14) $F'_{+R}(L', S) := F'_{+R}(L', S) + \{r_{+R}: L' \rightarrow R_i+R'_i+R^{\sim}\}$;
(15) return $F'_{+R}(L', S)$;

IV. EXPERIMENTAL RESULTS

Four benchmark databases in [8] are used during these experiments. Table I shows their characteristics. The source code of M. J. Zaki [9] is also used to find the frequent closed itemset lattice (Charm-L [7]) and generators.

TABLE I. DATABASE CHARACTERISTICS

Database (DB)	# Transaction	# Items	Average size
P	49046	7117	74
M	8124	119	23
Co	67557	129	43
Ch	3196	75	37

Table II shows the experimental results of our approach for finding association rule set based on the basic rules as min-min form. It shows the minimum support and the minimum confidence (MS=MC), the cardinality of the association rule set (#Tra) and the cardinality of basic rule set (#BAR). Column RT_c shows the percent ratio of the time for finding the consequence rules to the one for finding all association rules. The ratio of the basic rules to all association rules and the number of redundant candidates generated in the *GenerateRules* algorithm [6] are in turn showed in columns R_{BT} and $\#R_z$. Table II also shows the run time for mining the basic rules by our algorithm *MG_BARS* (T_o) and the ratio (RT) of the one by the *GenerateRules* (T_z) to T_o.

TABLE II. THE EXPERIMENTAL RESULTS WITH BENCHMARK DATABASES

DB	MS = MC	#Tra	#BAR	R_{BT}	RT_c	$\#R_z$	T_o (s)	RT
Ch	80	552564	316493	0.6	88.9	480	1.07	1.5
Ch	70	8171198	3396360	0.4	91.3	6498	11.34	1.6
Co	97	8092	4621	0.6	88.0	21	0.02	1.5
Co	90	3640704	324974	0.1	96.9	10000	1.00	2.2
M	40	7020	1419	0.2	93.1	26	0.01	1.5
M	20	19191656	59297	0.0	99.9	16166	0.30	8.2
P	95	1170	786	0.7	81.4	0	0.01	2.0
P	85	1408950	727532	0.5	87.4	1368	2.90	1.7

The experimental results show that: first, the algorithm *MG_BARS* quickly and directly finds basic rules; second, the algorithm *MG_CARS* completely derives all confidence-preserved and non-repeated consequence rules from the basic rules by adding appropriate eliminable itemsets to two sides of them. The total number of rules generated in our approach is the same as the one in the traditional algorithms [6]. Figures 2 and 3 show the relation between the cardinality of basic rule set and the one of association rule set and also the effect of minimum confidence on the number of basic rules.

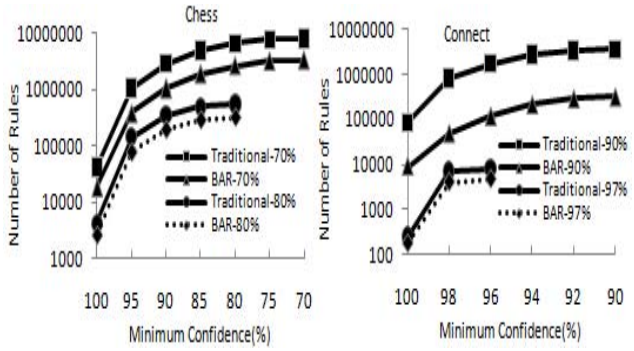


Figure 2. All rules vs basic rules: Chess and Connect.

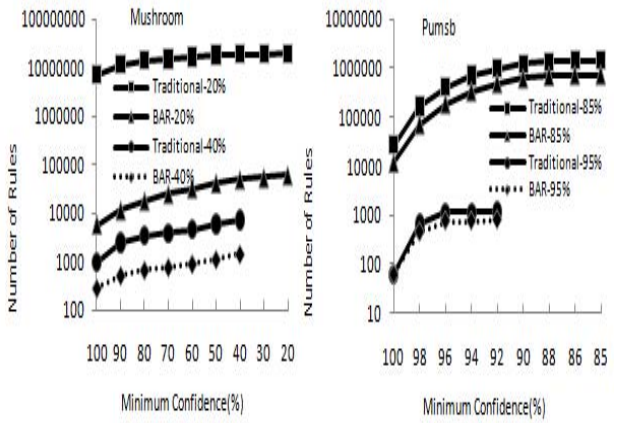


Figure 3. All rules vs basic rules: Mushroom and Pumsb.

Figure 4, 5 and 6 compare the run times for finding the basic rules by the *MG_BARS* and *GenerateRules* algorithms on Co (similar to Ch), M and P with the different minimum confidences. It shows that the run time of our algorithm *MG_BARS* is shorter in almost cases.

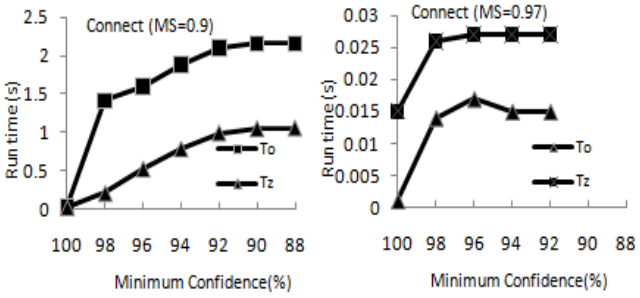


Figure 4. The run times of *MG_BARS* vs. *GenerateRules*: Connect.

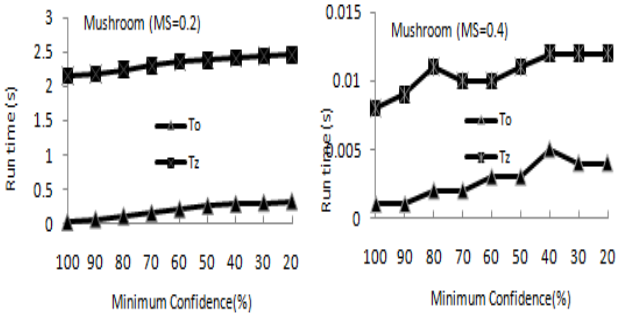


Figure 5. The run times of *MG_BARS* vs. *GenerateRules*: Mushroom.

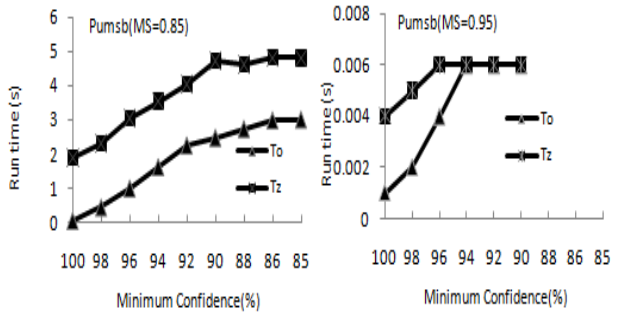


Figure 6. The run times of *MG_BARS* vs. *GenerateRules*: Pumsb.

V. CONCLUSION

In this paper, based on the eliminable itemset concept [5] which plays an important role in preserving support of itemsets and confidence of rules in each equivalence class, an efficient approach for extracting all association rules based on min-min basis is proposed. This approach with four phases is built based on the theoretical results and tested on benchmark databases. The first phase is to partition the association rule set into the disjoint equivalence rule classes. The second one is to disjointly split each of them into two rules sets of min-min basis and consequent. Using the above structures, in the third phase the algorithm *MG_BARS* which significantly reduces the time for mining basic rules is obtained. And in the last phase, the algorithm *MG_CARS* that non-repeatedly and completely derive all consequence rules (together with their support and confidence) from the basic rules is proposed.

ACKNOWLEDGMENT

We would like to express our sincere thanks to M. J. Zaki for his permission of using his source code [9] in our research. We also would like to thank the Department of Mathematics and Informatics, University of Dalat for their valuable support in the completion of this article.

REFERENCES

[1] C. C. Aggarwal, P. S. Yu, "Online generation of association rules" in Proceedings of the international conference on data engineering, pp. 402-411, 1998.

[2] H. T. Bao, "An approach to concept formation based on formal concept analysis," IEICE Trans. Infor. and systems, vol. E78-D, no. 5, 1995.

[3] N. Pasquier, R. Taouil, Y. Bastide, G. Stumme, L. Lakhal, "Generating a condensed representation for association rules" in J. of Intelligent Information Systems, vol. 24, no. 1, pp. 29-60, 2005.

[4] R. Taouil, Y. Bastide, N. Pasquier, L. Lakhal, "Mining bases for association rules using closed sets" in 16th IEEE Intl. Conf. on Data Engineering, 2000.

[5] T. C. Truong, A. N. Tran, "Structure of set of association rules based on concept lattice" in Advances in intelligent information and database systems, pp. 217–227, N. T. Nguyen et al. (Eds.), Springer, 2010.

[6] M. J. Zaki, "Mining non-redundant association rules" in Data mining and knowledge discovery, 9, pp. 223-248, 2004.

[7] M. J. Zaki, C-J. Hsiao, "Efficient algorithms for mining closed itemsets and their lattice structure," IEEE Trans. Knowledge and data engineering, vol. 17, no. 4, 2005.

[8] Frequent Itemset Mining Dataset Repository, http://fimi.cs.helsinki.fi/data/ (2009).

[9] http://www.cs.rpi.edu/~zaki/www-new/pmwiki.php/Software/Software#patutils.