ARCA: An Algorithm for Mining Association Rules based Concept Lattice

Qing Xia
Library
Huaihai Institute of Technology
Lianyungang, China, 222005
Email:xiaqing1@163.com

Sujing Wang
College of Computer Science
and Technology,
Jilin University, Changchun, China, 130012
Email: sujingwang@hotmail.com

Zhen Chen
College of Computer Science
and Technology,
Jilin University, Changchun, China, 130012
Email: chenzhen@jlu.edu.cn

Tao Lv
Weapon System Engineering Center
Artillery Command Academy,XuanHua, China 075100
Email: lvtom@sohu.com

Dongjing Wang
Shizhuang Middle School, Lianyungang, China, 222200
Email: rpcwangdongjing@163.com

***Abstract*—Association rule discovery is one of kernel tasks of data mining. Concept lattice, induced from a binary relation between objects and features, is a very useful formal analysis tool. It represents the unification of concept intension and extension. It reflects the association between objects and features, and the relationship of generalization and specialization among concepts. There is a one-to-one correspondence between concept intensions and closed frequent itemsets. This paper presents an efficient algorithm for mining association rules based concept lattice called Arca (Association Rule based Concept lAttice). Arca algorithm uses concept-matrix to build a part of concept lattice, in which the intension of every concept be put into one-to-one correspondence with a closed frequent itemset. Then all association rules are discovered by 4 operators which are defined in this paper performed on these concepts.**

***Index Terms*—Concept lattice, rank of matrix, formal concept analysis.**

I. INTRODUCTION

Association rule mining from a transaction database has been a very active research area since the publication of the Apriori algorithm [1]. Several improvements to the basic algorithm and many new approaches [2]–[10] have been proposed during the last decade. With the development of research, Association rule discovery is one of kernel tasks of data mining.

Formal Concept Analysis (FCA) was developed by Pro. Wille in 1982 [11]. Concept Lattice, the core data structure in Formal Concept Analysis, has been widely in machine learning, data mining and knowledge discovery, etc. Every node of concept lattice is a formal concept consisting of extent and intent. Concept lattice embodies the relations between extension and intension. Here is a one-to-one correspondence between concept intensions and closed frequent itemsets.

There are various algorithms [12]–[16] of association rule mining using concept lattice. However, These algorithms need to build a complete concept lattice. Based on CMCG algorithm [17], this paper presents an algorithm Arca of association rule mining using a part of concept lattice.

The paper is organized as follows. Section 2 recalls basic definitions of association rule and concept lattice. Section 3 discusses Arca algorithm and four operator. Section 4 gives an experimental evaluation on the time spent of Arca algorithm and Apriori algorithm. Section 5 concludes the paper.

II. THE DEFINES OF ASSOCIATION RULE AND CONCEPT LATTICE

Let $\mathscr{I} = \{i_1, i_2, \ldots, i_m\}$ be a set of m items.Let $(T) = \{t_1, t_2, \ldots, t_n\}$, the task-relevant data, be a set of database transactions where each transaction t is a set of items such that $t \subseteq \mathscr{I}$. Each transaction is associated with an identifier, called TID. Each transaction t consists of a set of items I from $\mathscr{I}$. If $|I| = k$, then I is called a $k-itemset$. A transaction t is said to contain I if and only if $I \subseteq t$. An association rule is an implication of ten form $I_1 \Rightarrow I_2$, where $I_1, I_2 \subset \mathscr{I}$ and $I_1 \cap I_2 = \emptyset$. The rule $I_1 \Rightarrow I_2$ holds in the transaction set $\mathscr{T}$ with support s, where s is the percentage of transactions in $\mathscr{T}$ that contain $I_1 \cup I_2$ (i.e., both I_1 and I_2). This is taken to be the probability, $P(I_1 \cup I_2)$. The rule $I_1 \Rightarrow I_2$ has confidence c in the transaction set $\mathscr{T}$ if c is the percentage of transactions in $\mathscr{T}$ containing I_1 that also contain I_2. This is taken to be the conditional probability, $P(I_2|I_1)$. That is,

$$support(I_1 \Rightarrow I_2) = P(I_1 \cup I_2) \tag{1}$$

$$confidence(I_1 \Rightarrow I_2) = P(I_2|I_1) \tag{2}$$

Given the user defined minimum support $minsupp$ and minimum confidence $minconf$ thresholds. If the support of $I \subseteq t$ itemset I be greater or equal to $minsupp$, I is called a $frequentitemset$.

Example 2.1: For $\mathscr{T} = \{A, B, C, D, E\}, \mathscr{I} = \{A, B, C, D, E\}$, Table I represents a transaction database.

Definition 2.1: A data mining context is a triple:$\mathscr{D} = (\mathscr{T}, \mathscr{I}, \mathscr{R})$, where $\mathscr{I}$ and $\mathscr{T}$ are two sets, and $\mathscr{R}$ is a relation between $\mathscr{I}$ and $\mathscr{T}$. $\mathscr{T} = \{t_1, t_2, \ldots, t_n\}$, each $t_i (i \leq n)$ is

TABLE I
A TRANSACTION DATABASE

TID	Iterms
1	A CD
2	BC E
3	ABC E
4	B E
5	ABC E

TABLE II
A TRANSACTION DATABASE

	A	B	C	D	E
1	1	0	1	1	0
2	0	1	1	0	1
3	1	1	1	0	1
4	0	1	0	0	1
5	1	1	1	0	1

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} A & B & C & D & E \end{array} \\ \begin{pmatrix} 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 \end{pmatrix} \end{array}$$

Fig. 1. A context-matrix of the data mining context showed in TableII

called an object. $\mathscr{I} = \{i_1, i_2, \ldots, i_m\}$, each $i_j (j \leq m)$ is called an attribute.
In a data mining context $\mathscr{D} = (\mathscr{T}, \mathscr{I}, \mathscr{R})$, if $(t, i) \in \mathscr{R}$, we say that the attribute i is an attribute of the object t, or that t verifies i. In this paper, $(t, i) \in \mathscr{R}$ is denoted by 1, and $(t, i) \notin (R)$ is denoted by 0. Thus, a data mining context can be represented by a matrix only with 0 and 1. We say that the matrix is the context-matrix of $\mathscr{D}$.

Example 2.2: Table II represents a data mining context corresponding with the transaction database showed in Table I.

Example 2.3: Fig 1 a context-matrix of the data mining context showed in Table II.

Definition 2.2: Let $\mathscr{D} = (\mathscr{T}, \mathscr{I}, \mathscr{R})$ be a data mining context. We define a function $f(T)$ that produces the set of their common attributes for every set $T \subseteq \mathscr{T}$ of objects to know which attributes from $\mathscr{I}$ are common to these entire objects: $f(T) = \{i \in \mathscr{I} | \forall t \in \mathscr{T}, (t, i) \in \mathscr{R}\}$

Dually, we define Y for subset of attributes $I \subset \mathscr{I}$, $g(I)$ denotes the set consisting of those objects in $\mathscr{T}$ that have all the attributes from $\mathscr{I}$: $g(I) = \{t \in \mathscr{T} | \forall i \in \mathscr{I}, (t, i) \in \mathscr{R}\}$. Let $h(I) = f(g(I))$.

These two functions are used to determine a formal concept.

Definition 2.3: Let $\mathscr{D} = (\mathscr{T}, \mathscr{I}, \mathscr{R})$ be a data mining context. A pair (T, I) is called a formal concept of $\mathscr{D}$, for short, a concept, if and only if $T \subseteq \mathscr{T}$, $I \subseteq \mathscr{I}$, $f(T) = I$ and $g(I) = T$. T is called extent, I is called intent.

Definition 2.4: Let $\mathscr{D} = (\mathscr{T}, \mathscr{I}, \mathscr{R})$ be a data mining context. The set of all concepts of $\mathscr{D}$ is denoted by $B(\mathscr{D})$, $C_1 = (T_1, I_1)$ and $C_2 = (T_2, I_2)$ are two concepts in $B(\mathscr{D})$. An partial ordering relation (¡) is defined on B(D) by: $C_1 < C_2 \Leftrightarrow T_1 \subset T_2$ or $C_1 < C_2 \Leftrightarrow I_1 \supset I_2$

We say that C_2 is called a *superconcept* of C_1 and C_1 is called a *subconcept* of C_2. $B(\mathscr{D})$ and the partial ordering relation (¡) form a compete lattice called the *concept lattice* of $\mathscr{D}$ and denoted by $L(\mathscr{D})$.

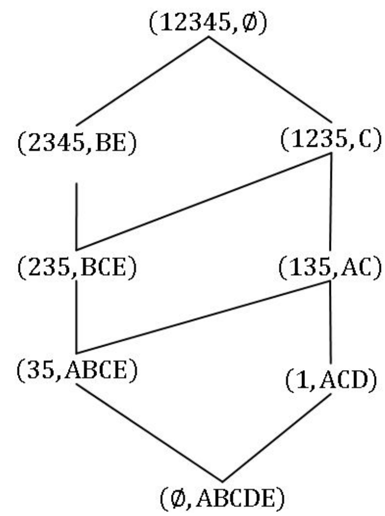

Fig. 2. A Concept lattice for the context of TableII

Example 2.4: Fig 2 a concept lattice for the context of Table II.

Definition 2.5: Let $\mathscr{D} = (\mathscr{T}, \mathscr{I}, \mathscr{R})$ be a data mining context. C_1 and C_2 are two concepts in $B(\mathscr{D})$. If $C_1 < C_2$ and there is no concept C_3 in $B(\mathscr{D})$ fulfilling $C_1 < C_3 < C_2$, C_1 is called a lower neighbor of C_2 , denoted by $C_1 \prec C_2$, and C_2 is called a *upper neighbors* of C_1.

The set of all lower neighbors of a given concept is a subset of the set consisting of all subconcepts of it.

Definition 2.6: Let $I \subseteq \mathscr{I}$ be a set of items from $\mathscr{D} = (\mathscr{T}, \mathscr{I}, \mathscr{R})$. The *support count* of the itemset I in $\mathscr{D}$ is:

$$support(I) = \frac{|g(I)|}{|\mathscr{T}|} \tag{3}$$

Definition 2.7: Let $I \subseteq \mathscr{I}$ be a set of items from $\mathscr{D} = (\mathscr{T}, \mathscr{I}, \mathscr{R})$. If $support(I) minsupp$, I is called a *frequent itemset*.

Definition 2.8: Let $\mathscr{D} = (\mathscr{T}, \mathscr{I}, \mathscr{R})$ be a data mining context. For given a item $i \in \mathscr{I}$, if the count of 1s in the corresponding column of the item i in the concept-matrix of $\mathscr{D}$ is n we say that the rank of item i in concept-matrix of $\mathscr{D}$ is n, denoted by $r(i) = n$. If $m = \max\{r(i) | i \in \mathscr{I}\}$, we say that the rank of the data mining contgext $\mathscr{D}$ is m.

Definition 2.9: Let $\mathscr{D} = (\mathscr{T}, \mathscr{I}, \mathscr{R})$ be a data mining context. The concept-matrix of $C = (T, I)$ is the matrix consisting of these rows what are the corresponding rows of the each element t in set T in context-matrix of $\mathscr{D}$.

Example 2.5: Fig 3 represents the concept-matrix of concept $(135, AC)$.

$$\begin{array}{c} \\ 1 \\ 3 \\ 5 \end{array}\begin{array}{c} \begin{array}{ccccc} A & B & C & D & E \end{array} \\ \begin{pmatrix} 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 \end{pmatrix} \end{array}$$

Fig. 3. The concept-matrix of concept $(135, AC)$

Definition 2.10: Let $\mathscr{D} = (\mathscr{T}, \mathscr{I}, \mathscr{R})$ be a data mining context. $C = (T, I)$ is a concept in $B(\mathscr{D})$. If the count of 1s in the corresponding column of the item i in the concept-matrix of C is n, we say that the rank of item i in concept-matrix of concept C is n, denoted by $R_C(i) = n$. If $m = \max\{R_C(i) | i \in \mathscr{I}, i \notin I\}$, we say that the rank of concept C is m.

Property 2.1: The count of objects of subconcept of concept C is equal or lesser than m.

Definition 2.11: Let $\mathscr{D} = (\mathscr{T}, \mathscr{I}, \mathscr{R})$ be a data mining context. $C = (T, I)$ is a concept in $B(\mathscr{D})$.Given subset $I_1 \subseteq \mathscr{I}$, $g_C(I_1)$ denotes the set consisting of those transactions in T that have all the itemsets from $\mathscr{I}$: $g_C(I_1) = \{t \in T | \forall i \in I_1, (t, i) \in \mathscr{R}\}$

Definition 2.12: Let $\mathscr{D} = (\mathscr{T}, \mathscr{I}, \mathscr{R})$ be a data mining context. $C = (T, I)$ is a concept in $B(\mathscr{D})$.If $|T| \geq |\mathscr{T}| \times minsupp$. C is called a *frequent concept*.

Property 2.2: If $C = (T, I)$ is a frequent concept, I is a frequent itemset.

Proof: $|T| \geq |\mathscr{T}| \times minsupp$, so $\frac{|T|}{|\mathscr{T}|} \geq minsupp$. And $T = g(I)$, so $\frac{|g(I)|}{|T|} \geq minsupp$. Then I is a frequent itemset. ■

Theorem 2.1: Let $\mathscr{D} = (\mathscr{T}, \mathscr{I}, \mathscr{R})$ be a data mining context. $C = (T, I)$ is a concept in $B(\mathscr{D})$. The rank of concept C is m. For $\forall i \in \{i | R_C(i) = m, i \in \mathscr{I}\}$, $C_1 = (g_C(i), f(g_C(i)))$. Then C_1 is a lower neighbor of C.

Proof: By Definition 2.10 And Definition 2.11, we have $|g_C(i)| = m$. Suppose there exist $C_2 = (T_2, I_2)$, where $C_1 \prec C_2$. We can obtain $m = |g_C(i)| < |T_2| < |T|$. By Property 2.1, we have $|T_2| \leq m$. This result contradicts with $m < |T_2|$. Then C_1 is a lower neighbor of C. ■

Theorem 2.2: Let $\mathscr{D} = (\mathscr{T}, \mathscr{I}, \mathscr{R})$ be a data mining context. $C = (T, I)$ is a concept in $B(\mathscr{D})$. The rank of concept C is m. $C_1 = (g_C(i_1), f(g_C(i_1)))$ is a *subconcept* of C, where $i_1 \in \mathscr{I}$ and $R_C(i_1) = m_1 > 0$. For $\forall C_2 \in \{C_2 = (T_2, I_2) | C_2 < C, m_1 < |T_2|\}$, there NOT exist $g_C(i_1) \subset T_2$. Then C_1 is a lower neighbor of C.

Proof: Suppose there exist $C_3 = (T_3, I_3)$, where $C_1 < C_3 < C$, We have $m_1 = |g(i_1) \cap T| < |T_3| < |T|$, it implies that $C_3 \in \{C_2 = (T_2, I_2) | C_2 < C, m_1 < |T_2|\}$, and NOT $g_C(i_1) \subset I_3$. On the other hand $C_1 < C_3 < C$ implies that $g_C(i_1) \subset I_3$. This result contradicts with NOT $g_C(i_1) \subset I_3$. Then C_1 is a lower neighbor of C. ■

Definition 2.13: An association rule is an implication between itemsets of the form $r : I_1 \to I_2$, where $I_1, I_2 \subset \mathscr{I}$ and $I_1 \cap I_2 = \emptyset$. I_1 is called the antecedent of r and I_2 is called the consequent of r. Below, we define the support and confidence of an association r:

$$support(r) = \frac{|g(I_1 \cup I_2)|}{|\mathscr{T}|} \tag{4}$$

$$confidence(r) = \frac{support(I_1 \cup I_2)}{support(I_1)} = \frac{|g(I_1 \cup I_2)|}{|g(I_1)|} \tag{5}$$

Mining association rules is to find all rules r, where $support(r) \geq minsupp$ and $confidendce(r) \geq minconf$.

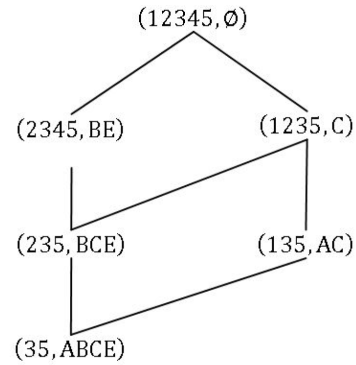

Fig. 4. A concept lattice while $minsupp = 0.4$

TABLE III
BASIC ASSOCIATION RULES FROM FIG 4 WITH $minconf = 0.5$

basic association rule	minimum support
$\emptyset \to BE$	4/5
$\emptyset \to C$	4/5
$BE \to C$	3/4
$C \to BE$	3/4
$C \to A$	3/4
$BCE \to A$	2/3
$AC \to BE$	2/3

III. ARCA ALGORITHM

When a concept lattice is built, each concept $C = (T, I)$ holds $|T| \geq minsupp$.

Example 3.1: Fig 4 represents a concept lattice while $minsupp = 0.4$.

Definition 3.1: Let $\mathscr{D} = (\mathscr{T}, \mathscr{I}, \mathscr{R})$ be a data mining context. $C_1 = (T_1, I_1)$ and $C_2 = (T_2, I_2)$ are two concepts in $B(\mathscr{D})$ and $C_1 \prec C_2$. If $\frac{|T_1|}{|T_2|} \geq minconf$, the rule $r : I_2 \Rightarrow I_1 - I_2$ is called a *basic association rules*. The minimum support of r is defined as $\frac{|T_1|}{|T_2|}$.

Basic association rules can be mined from concept lattice.

Example 3.2: Table III represents basic association rules from Fig 4 with $minconf = 0.5$.

Definition 3.2: Let $r_1 \equiv I_1 \to I_2$, $r_2 \equiv I_3 \to I_4$, the minimum supports of r_1 and r_2 are $conf_1$ and $conf_2$, respectively. If $I_1 \cup I_2 = I_3$, The operator '+' can be implemented on r_1 and r_2. $r_1 + r_2 \equiv I_1 \to I_4$. The minimum support of $I_1 \to I_4$ is defined as $conf_{12} = conf_1 \times conf_2$. Let $r_3 \equiv I_4 \to I_5$ and its minimum supports is $conf_3$. If $I_1 \cup I_2 = I_3$ and $I_3 \cup I_4 = I_5$, the operator '+' can be implemented on r_1, r_2 and r_3. $r_1 + r_2 + r_3 \equiv I_1 \to I_6$ and its minimum support

TABLE IV
THE RESULT OF PERFORMING '+' AND '$\oplus$' ON BASIC ASSOCIATION RULES IN TABLE III.

Basic rules	The result of '+'	The result of '$\oplus$'
	$\emptyset \to C$	$\emptyset \to BCE$
	$BE \to A$	$BE \to AC$
	$\emptyset \to A$	$\emptyset \to ABCE$
	$\emptyset \to BE$	$\emptyset \to BE$
	$C \to A$	$C \to ABE$
	$C \to A$	$C \to ABCE$
	$\emptyset \to A$	$\emptyset \to AC$
	$C \to BE$	$C \to ABE$
	$\emptyset \to BE$	$\emptyset \to ABCE$

is $conf_{123} = conf_1 \times conf_2 \times conf_3$. The operator '+' can be defined among n rules by same way.

Definition 3.3: Let $r_1 \equiv I_1 \to I_2$, $r_2 \equiv I_3 \to I_4$, the minimum supports of r_1 and r_2 are $conf_1$ and $conf_2$, respectively. If $I_1 \cup I_2 = I_3$, The operator '$\oplus$' can be implemented on r_1 and r_2. $r_1 \oplus r_2 \equiv I_1 \to I_2 \cup I_4$. The minimum support of $I_1 \to I_2 \cup I_4$ is defined as $conf_{12} = conf_1 \times conf_2$. Let $r_3 \equiv I_4 \to I_5$ and its minimum supports is $conf_3$. If $I_1 \cup I_2 = I_3$ and $I_3 \cup I_4 = I_5$, the operator '$\oplus$' can be implemented on r_1, r_2 and r_3. $r_1 \oplus r_2 \oplus r_3 \equiv I_1 \to I_2 \cup I_4 \cup I_6$ and its minimum support is $conf_{123} = conf_1 \times conf_2 \times conf_3$. The operator '$\oplus$' can be defined among n rules by same way.

Let's examine how to perform '+' and '$\oplus$' on basic association rules in Table III Using the following example.

Example 3.3: Since BE, the union of the antecedent and consequent of rule , be not equal to the antecedent of rule , the operators '+' and '$\oplus$' cannot be implemented on rule and rule . The antecedent of rule be equal to BE. Therefore, the operators '+' and '$\oplus$' cannot be implemented on rule and rule .

$$+ \quad \equiv \emptyset \to C \tag{6}$$

$$\oplus \quad \equiv \emptyset \to BCE \tag{7}$$

Every basic association rule after rule is check whether the operators '+' and '$\oplus$' can be implemented on rule and itself. and not. yes. Therefore, we can obtain:

$$+ \quad \equiv BE \to A \tag{8}$$

$$+ \quad + \quad \equiv \emptyset \to A \tag{9}$$

$$\oplus \quad \equiv BE \to AC \tag{10}$$

$$\oplus \quad \oplus \quad \equiv \emptyset \to ABCE \tag{11}$$

Table IV represents the result of performing '+' and '$\oplus$' on basic association rules in Table III.

Definition 3.4: Let $r_1 \equiv I_1 \to I_2$. The result of performing operator *right move* is a set of association rules $\{I_1' \to I_2 \cup I_1'' | I_1' \cap I_1'' = \emptyset, I_1' \cup I_1'' = I_1, I_1' \neq, I_1'' \neq \emptyset, g(I_1) \leq g(I_1')\}$

Definition 3.5: Let $r_1 \equiv I_1 \to I_2$. The result of performing operator decompose is a set of association rule $\{I_1 \to I_2' | I_2' \neq \emptyset, I_2' \subset I_2\}$

All association rules which confidence be less than 1 are mining out by performing four operators on basic association rules. The pseudo codes are given by Algorithm 1.

Algorithm 1 GenBasicRules($\mathscr{D}$,$minsupp$,$minconf$)

Require: Input: minimum support $minsupp$, minimum confidence $minconf$ and data mining context $\mathscr{D} = (\mathscr{T}, \mathscr{I}, \mathscr{R})$
Ensure: Output: a set of basic association rules $BasicRules$
  Initialize a queue $Queue$
  $BasicRules \leftarrow \emptyset$
  $Queue.EnQueue((\mathscr{T}, \emptyset))$
  while $Queue \neq \emptyset$ **do**
    $C = Queue.DeQueue$
    for all C_1 such that $C_1 \in SUBNODES*(C)$ **do**
      if $(|Extent(C_1)|/|Extent(C)|) \geq minconf$ **then**
        if $C_1 \notin Queue$ **then**
          $Queue.EnQueue(C_1)$
        end if
        generate a rule $r : Intent(C) \to Intent(C_1)$
        $BasicRules \leftarrow BasicRules \cup r$
      end if
    end for
  end while
  return $BasicRules$

Algorithm 2 SUBNODES*(C,$\mathscr{D}$,$minsupp$))

Require: Input: given a concept $C = (T, I)$, minimum support $minsupp$, and data mining context $\mathscr{D} = (\mathscr{T}, \mathscr{I}, \mathscr{R})$
Ensure: Output:a set $subnodes$ of concepts which extent's count be greater or equal to $minsupp$
  $subnodes \leftarrow \emptyset$
  $M \leftarrow$ the concept matrix of concept C
  compute the rank of every attribute in M
  $m \leftarrow$ the rank of concept C
  while $m \geq minsupp$ **do**
    $S \leftarrow$the set of attributes which ranks equal to m
    while $S \neq \emptyset$ **do**
      $I_1 \leftarrow$ the set consisting of a attribute a from S and these attributes from S which corresponding columns are same as column of a in M
      $S \leftarrow S - I_1$
      $T_1 \leftarrow g_C(I_1)$
      $I_1 \leftarrow I \cup I_1$
      if $\forall C_2 = (T_2, I_2) \in subnodes$, such that NOT $T_1 \subset T \cap T_2$ **then**
        $subnodes \leftarrow subnodes \cup (T_1, I_1)$
      end if
    end while
    $m \leftarrow m - 1$
  end while
  return $subnodes$

Algorithm 1 is to generate basic association rules by build a concept lattice called function SUBNODES* The pseudo codes of this function are given by Algorithm 2.

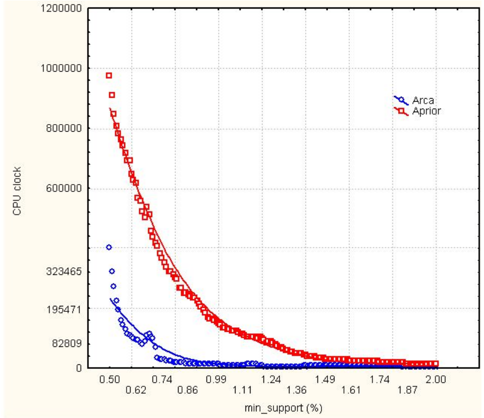

Fig. 5. Running time of algorithms with $minconf = 0.01$

IV. Evaluation

In order to evaluate, we implement Algorithm Arca and Algorithm Aprior by Visual C++ and STL. The data set, generated randomly by IBM dataset generator, have 1000 items and 10000 transactions. The result shows that the performance of Arca is as four times higher as Aprior on average. Fig 5 represents Running time of algorithms versus lattice size with $minconf = 0.01$.

V. Conclusion

Now, there are many algorithms of mining association rules. There is a one-to-one correspondence between concept intensions and closed frequent itemsets. Concept lattice is a good tool for mining association rules.

References

[1] Agrawal R, Imielinski T, Swami A. Mining association rules between sets of items in large databases[J]. ACM SIGMOD Record: 1993, 22: 207-216

[2] Savasere A, Omiecinski E, Navathe S. An efficient algorithm for mining association rules in large databases[C]. Proceedings of the 21st VLDB Conference. the 21st VLDB Conference, Zurich, 1995. 432-444

[3] Park J S, Chen M S, Yu P S. An effective hash-based algorithm for mining association rules[C]. ACM SIGMOD Record . ACM Press, 1995, 24: 175-186.

[4] Zaki M J, et al. New algorithms for fast discovery of association rules[R]. University of Rochester : 1997.

[5] Shenoy P, et al. Turbo-charging vertical mining of large databases[C]. ACM SIGMOD Record. ACM Press, 2000, . 22-33.

[6] Pasquier N, et al. Discovering frequent closed itemsets for association rules[C]. Proceeding of the 7th International Conference on Database Theory . The 7th International Conference on Database Theory, Jerusalem, 1999. Berlin: Springer-Verlag, 1999, 1540: 398-416.

[7] Pei J, Han J, Mao R. An efficient algorithm for mining frequent closed itemsets[C]. 2000 ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery, Dallas, 2000. 21-30.

[8] Han J, et al. FreeSpan: frequent pattern-projected sequential pattern mining[C]. Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining. The sixth ACM SIGKDD international conference on Knowledge discovery and data mining, 2000. Boston : ACM Press, 2000, 355-359.

[9] Han J, et al. PrefixSpan: mining sequential patterns efficiently by prefix-projected pattern growth[C]. The Proceedings of 17th International Conference. The17th International Conference, .2001. 215-224

[10] Han J, Pei J, Yin Y. Mining frequent patterns without candidate generation[C]. 2000 ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery, Dallas, 2000. 1-12.

[11] Godin R, Missaoui R. An incremental concept formation approach for learning from databases[J]. Theoretical Computer Science: 1994, 133: 387-419

[12] Valtchev P, Missaoui R, Lebrun P. A partition-based approach towards constructing Galois (concept) lattices[J]. Discrete Mathematics: 2002, 256(3):801-829.

[13] Pasquier N, Bastide Y, Taouil R, Lakhal L. Discovering frequent closed itemsets for association rules[C]. Proceeding of the 7th International Conference on Database Theory . The 7th International Conference on Database Theory, Jerusalem, 1999. Berlin: Springer-Verlag, 1999, 1540: 398-417

[14] Stumme G. Efficient data mining based on formal concept analysis[C]. Proceeding of the 13th International Conference. Database and Expert Systems Applications : 13th International Conference, DEXA 2002 Aix-en-Provence, 2002. Berlin: Springer-Verlag, 2453:534-547

[15] Pasquier N, Bastide Y, Taouil R, Lakhal L. Efficient mining of association rules using closed lattices[J]. Information System: Elsevier, 1999, .24(1): 25-46 16. Stumme G, Taouil R, Bastide Y, Pasquier N, Lakhal L. Fast Computation of Concept Lattices Using Data Mining Techniques[C].

[16] Proceeding of 7th Intl. Workshop on Knowledge Representation Meets Databases (KRDB'00). The 7th Intl. Workshop on Knowledge Representation Meets Databases (KRDB'00), Berlin, 2000.

[17] Sujing Wang, Zhen Chen and Dongjing Wang. An Algorithm based on Concept Matrix for Building Concept Lattice with Hasse[C]. Proceeding of 2007 International Conference on Wireless Communications, Networking and Mobile Computing : 2007 International Symposium on Information Systems And Management, Shanghai, 2007. Volume 8, 5593-5596