Implementing Association Rules Generator using CHARM

Zay Aung, Thinn Naing
University of Computer Studies, Yangon
mgzayaung.cs@gmail.com

Abstract

In traditional association rule mining that based on Apriori-like algorithm, all frequent itemsets are computed and strong association rules are then generated from the set of all frequent itemsets as a next step. By contrast, in **closed association rule mining**, *instead of mining all frequent itemsets in the first step, only the set of all closed frequent itemsets is mined where the size of the set of all closed frequent itemsets is much smaller than that of all frequent itemsets. Instead of generating rules from the set of all frequent itemsets, rules are generated from the set of all closed frequent itemsets , where the size of rules generated from the set of closed frequent itemsets is much smaller than that generated from the set of all closed frequent itemsets. This paper describes the concept of closed association rule mining, how CHARM algorithm works to find out all closed frequent itemsets, and the implementation of rules generating system or rules generator that mainly uses closed association rule mining techniques by using CHARM Algorithm.*

Keywords: itemsets, transaction sets, support, confidence, closed frequent itemsets, closure of itemsets, closure of transaction sets.

1. INTRODUCTION

Today, association rule mining has become one of the most challenging tasks among data mining researchers and practitioners. Association Rule Mining is involves two main steps: (1) Finding all frequent itemsets, (2) Generating strong association rules from the set of all frequent itemsets that has been just computed in step (1). In association rule mining, every rule has support and confidence to measure its interestingness and strongness. Thus, association rules are considered interesting and strong if they satisfy given minimum confidence. Itemsets are considered frequent if they satisfy given minimum support count.

In closed association rule mining, all frequent itemsets are distinguished into two main broad categories, namely, closed frequent itemsets and non-closed frequent itemsets. It is found that the set of association rules can grow rapidly as users lower the minimum support and minimum confidence. The larger the set of all frequent itemsets have, the larger the set of all association rules. In addition to this, the set of all closed frequent itemsets is much smaller than the set of all frequent itemsets since the set of all closed frequent itemsets is a subset of the set of all frequent itemsets.

In addition to this, all association rules generated from the set of all non-closed frequent itemsets are equivalent to those generated from the sets of all closed frequent itemsets. Consequently, it is not feasible to generate all association rules from the set of all frequent itemsets. It is just needed to generate those association rules from the set of all closed frequent itemsets where redundant rules are eliminated. The main steps involved in closed association rule mining are: (1) Finding all closed frequent itemsets, and (2) generating association rules from the set of all closed frequent itemsets that satisfy minimum confidence specified [1].

The system uses closed association rule mining instead of traditional association rule mining to generate interesting Boolean association rules. The main computation-intensive step of finding all closed frequent itemsets is implemented with the use of CHARM. It is not feasible to generate the set of all closed frequent itemsets by using Apriori-like methods that examine all subsets of a frequent itemset. Neither is it possible to use algorithms like MaxMiner [6] or Pincer-Search [7]. Thus, CHARM avoids enumerating all possible subsets of a closed itemset when enumerating the closed frequent itemsets [5].

The rest of the paper is organized as follows. Section 2 describes association rule mining concepts in general. Section 3 describes the concepts and techniques of closed association rule mining in detail. In section 4, overview of the system is described. Section 5 intends to explain how CHARM works to find out all closed frequent itemsets from the given input dataset. Section 6 describes how association rules are generated from each of the closed frequent itemsets. Section 7 describes experimental analysis of the system. Conclusion is presented in section 8.

2. ASSOCIATION RULE MINING

Let $\mathcal{I}$= {i1,i2,…..,im } be a set of items. Let $\mathcal{D}$, the task-relevant data, be a set of database transactions where each transaction T is a set of items such that T ⊆ $\mathcal{I}$. Each transaction is associated with an identifier, called TID. Let A be a set of items.

A transaction T is said to contain A if and only if A⊆T. An association rule is an implication of the form A⇒B, where $A \subset \mathcal{I}$, $B \subset \mathcal{I}$, and $A \cap B = \phi$. The rule A⇒B holds in the transaction set $\mathcal{D}$ with support s, where s is the percentage of transactions in $\mathcal{D}$ that contain $A \cup B$ (i.e., both A and B). This is taken to be the probability, P $(A \cup B)$. The rule A⇒B has confidence c in the transaction set $\mathcal{D}$ if c is the percentage of transactions in $\mathcal{D}$ containing A that also contain B. This is taken to be the conditional probability, P (B|A). That is, support (A⇒B) = $P(A \cup B)$ and confidence(A⇒B)=P(B|A). Rules that satisfy both a minimum support (min_sup) and a minimum confidence (min_conf) are called strong. By convention, support and confidence values are written as to occur between 0% and 100%, rather than 0 to 1.0 [2].

3. CLOSED ASSOCIATION RULE MINING

3.1 Closed Frequent Itemsets

3.1.1 Partial Order and Lattices

Let P be a set. A *partial order* on P is a binary relation $\leq$, such that for all x, y, z $\in$ P, the relation is: 1) Reflexive: $x \leq x$. 2) Anti-Symmetric: $x \leq y$ and $y \leq x$, implies x=y. 3) Transitive: $x \leq y$ and $y \leq z$, implies $x \leq z$. The set P with the relation _is called an *ordered set*, and it is denoted as a pair (P, $\leq$). $x \leq y$ is written if y and $x \neq y$.

Let $(P, \leq)$ be an ordered set, and let S be a subset of P. An element $l \in P$ is an *upper bound* of S if $s \leq u$ for all $s \in S$. An element $l \in P$ is a *lower bound* of S if $s \geq l$ for all $s \in S$. The least upper bound is called the **join** of S, and is denoted as $\bigvee S$, and the greatest lower bound is called the **meet** of S, and is denoted as $\bigwedge S$. If $S=\{x, y\}$, $x \vee y$ is also written for the join, and $x \wedge y$ for the meet. An ordered set $(L, \leq)$ is a *lattice*, if for any two elements x and y in L, the join $x \vee y$ and meet $x \wedge y$ always exist. L is a *complete lattice* if $\bigvee S$ and $\bigwedge S$ exist for all $S \subseteq L$. Any finite lattice is complete. L is called a *join semilattice* if only the join exists. L is called a *meet semilattice* if only the meet exists.

Let ρ denote the power set of S (i.e., the set of all subsets of S). The ordered set $(\rho(S), \subseteq)$ is a complete lattice, where the meet is given by set intersection, and the join is given by set union. For example, the partial orders $(\rho\ (\mathcal{I}), \subseteq)$, the set of all possible itemsets, and $(\rho\ (\mathcal{T}), \subseteq)$, the set of all possible tidsets are both complete lattices. The set of all frequent itemsets, on the other hand, is only a meet-semilattice. For any two itemsets, only their meet is guaranteed to be frequent, while their join may or may not be frequent. This follows from the well-known principle in association mining that, if an itemset is frequent, then all its subsets are also frequent [3].

3.1.2 Closed Itemsets

Let the binary relation δ be input database for association rule mining. Let $X \subseteq \mathcal{I}$ and $Y \subseteq \mathcal{T}$. Then, the mappings

$t : \mathcal{I} \rightarrow \mathcal{T}$, $t(X) = \{y \in \mathcal{T} \mid \forall\, x \in X, x\, \delta\, y\}$,

$i : \mathcal{T} \rightarrow \mathcal{I}$, $i(Y) = \{x \in \mathcal{I} \mid \forall\, y \in Y, x\, \delta\, y\}$

define a *Galois connection* between the partial orders $(\rho\ (\mathcal{I}), \subseteq)$ and $(\rho\ (\mathcal{T}), \subseteq)$, the power sets of $\mathcal{I}$ and $\mathcal{T}$ respectively. The Galois connection satisfies the following properties (where $X_1, X_2, X_3 \in \rho\ (\mathcal{I})$ and $Y_1, Y_2, Y_3 \in \rho\ (\mathcal{T})$):

1) $X_1 \subseteq X_2 \Rightarrow t(X_1) \supseteq t(X_2)$

2) $Y_1 \subseteq Y_2 \Rightarrow i(Y_1) \supseteq i(Y_2)$

3) $X \subseteq i(t(X))$ and $Y \subseteq t(i(Y))$.

Let S be a set. A function c: $\rho(S) \longrightarrow \rho(S)$ is a *closure operator* on S if, for $X, Y \subseteq S$, c satisfies the following properties:

1) Extension: $X \subseteq c(X)$,
2) Monotonicity: if $X \subseteq Y$, $c(X) \subseteq c(Y)$ and
3) Idempotency: $c(c(X)) = c(X)$.

A subset X of S is said to be *closed* if $c(X) = X$.

Lemma 1 *Let $X \subseteq \mathcal{I}$ and $Y \subseteq \mathcal{T}$. Let $c_{it}(X)$ denote the composition of the two mappings I o t(X) = i(t(X)). Dually, let $c_{ti}(Y)$ =t o i(Y) = t(i(Y)). Then c_{it} : $\mathcal{P}(\mathcal{I}) \mapsto \mathcal{P}(\mathcal{I})$ and c_{ti} : $\mathcal{P}(\mathcal{T}) \mapsto \mathcal{P}(\mathcal{T})$ are both closure operators on itemsets and tidsets respectively.*

A **closed itemset** is defined as an itemset X that is the same as its closure, i.e., $X = c_{it}(X)$. A closed tidset is a tidset $Y = c_{ti}(Y)$. The mappings c_{it} and c_{ti}, being closure operators, satisfy the three properties of extension, monotonicity, and idempotency [1].

3.2 Closed Frequent Itemsets Vs All Frequent Itemsets

Theorem 1 *For any itemset X, its support is equal to the support of its closure, i.e., $\sigma(X) = \sigma(c_{it}(X))$.*

Theorem 2: *The rule $X_1 \rightarrow X_2$ with confidence p is equivalent to the rule $i(t(X_1)) \rightarrow i(t(X_2))$ with confidence q where p=q.*

The above two theorems imply that the rules generated from the set of all non-closed frequent itemsets are equivalent to those generated from the set of all closed frequent itemsets. In other words, rules generated from the set of all frequent itemsets are redundant and therefore can be eliminated. Thus, non-closed frequent itemsets are not required to generate. Only closed frequent itemsets are required to generate [1].

The fact that rules generated from non-closed frequent itemsets are all equivalent to those generated from closed frequent itemsets plays a crucial role in implementing the rule generator or rule generating system. Because the performance overhead can be reduced dramatically in two ways. First, instead of generating all frequent itemsets, only closed frequent itemsets are generated whose size is much smaller than that of all frequent itemsets. Much of computing time can thus be reduced. Second, instead of generating rules from all frequent itemsets, association rules are generated only from closed frequent itemsets, whereby reducing computing time that has to be used in generating rules from non-closed frequent itemsets.

4. SYSTEM OVERVIEW

The system accepts a dataset of transaction itemset pairs as input and produces association rules as output. Since the system is designed to work for any application domain, input dataset is transformed into a uniform format and that formatted dataset is stored in the local server. The system is experimented with a sale dataset and a supply chain dataset. Association rules generated by the system as output are presented to the users of the system in form of an MS Excel file.

The input to the generator is a dataset of any kind such as sales transaction dataset, medical dataset , educational dataset, etc. However, the input dataset is limited to be such that it consists of a set of transactions and, for each transaction, it consists of a set of itemsets. The next inputs are minimum support and minimum confidence. Then the generator computes all closed frequent itemsets from the given dataset by using CHARM. The next step of the generator is to generate the association rules from the closed frequent itemsets.

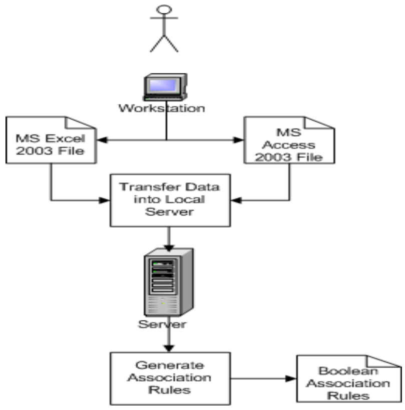
Workstation
MS Excel 2003 File
MS Access 2003 File
Transfer Data into Local Server
Server
Generate Association Rules
Boolean Association Rules

Figure 1. Overview of the System

4.1 Process Flow of the System

The system has two main processes. The first process is the process of transferring the data given by the user in the form of Excel or Access files with specified format into local server where the system is installed, converting the input data to the format the system processes. The second process is the process of generating strong association rules from the data on the local server already posted by the user. The first process includes reading the input file, transforming the data from it into the desired format and storing the formatted data on the server for subsequent processing. The second process starts with loading and initializing the data on a linked list. While loading the data, each itemset is checked to ensure that it does satisfy the minimum support count specified by the user. Then, CHARM algorithm is applied on the linked list to generate closed frequent itemsets. After closed frequent itemsets have been generated, strong association rules that do satisfy the minimum confidence specified by the user are then generated.

5. FINDING CLOSED FREQUENT ITEMSETS (CHARM ALGORITHM)

5.1 Basic Idea

The main computation intensive step in this process is to identify the closed frequent itemsets. Unlike all previous association mining methods, CHARM algorithm avoids enumerating all possible subsets of a closed itemset when enumerating the closed frequent itemsets. Further, CHARM uses a two-pronged pruning strategy. It prunes candidates based not only on subset infrequency as do all association mining methods, but it also prunes candidates based on non-closure property, i.e., any non-closed itemset is pruned.

The fundamental operation used in CHARM algorithm is a union of two itemsets and an intersection of two transactions lists where the itemsets are contained. The main computation in CHARM relies on the following properties.

1. If $t(X_1) = t(X_2)$, then $t(X_1 \cup X_2) = t(X_1) \cap t(X_2) = t(X_1) = t(X_2)$. Thus, every occurrence of X_1 can simply be replaced with $X_1 \cup X_2$ and X_2 must be removed from further consideration, since its closure is identical to the closure of $X_1 \cup X_2$. In other words, $X_1 \cup X_2$ is treated as a composite itemset.
2. If $t(X_1) \subset t(X_2)$, then $t(X_1 \cup X_2) = t(X_1) \cap t(X_2) = t(X_1) \neq t(X_2)$. Here every occurrence of X_1 can be replaced with $X_1 \cup X_2$, since if X_1 occurs in any transaction, then X_2 always occurs there too. However, since t (X1) ≠ t (X2), X_2 cannot be removed from consideration; it generates a different closure.
3. If $t(X_1) \supset t(X_2)$, then $t(X_1 \cup X_2) = t(X_1) \cap t(X_2) = t(X_1) \neq t(X_2)$. In this, every occurrence of X_2 can be replaced with $X_1 \cup X_2$, since wherever X_2 occurs X_1 always occurs. X_1, however, produces a different

closure, and it must be retained.

4. If $t(X_1) \neq t(X_2)$, then $t(X_1 \cup X_2) = t(X_1) \cap t(X_2) \neq t(X_2) \neq t(X_1)$. In this case, nothing can be eliminated; both X_1 and X_2 lead to different closures [1].

5.2 CHARM Algorithm

The algorithm starts by initializing the set of nodes to be examined to the frequent single items and their tidsets in Line 1. The main computation is performed in CHARM-EXTEND, which returns the set of closed frequent itemsets *C*.

```
CHARM ($\delta \subseteq \mathcal{I} \times \mathcal{T}$, minsup):
  1.  Nodes = $\{I_j \times t(I_j) : I_j \in \mathcal{I} \wedge |t(I_j)| \geq$ minsup$\}$
  2.  CHARM-EXTEND (Nodes, $\mathcal{C}$ )

CHARM-EXTEND (Nodes, )
  3.  for each $X_i \times t(X_i)$ in nodes
  4.      New N=0 and X= $X_i$
  5.      for each $X_j \times t(X_j)$ in nodes, with $f(j) < f(i)$
  6.          $X = X \cup X_j$ and $Y = X_i \cap t(X_j)$
  7.          CHARM-PROPERTY(Nodes, New N)
  8.      if New N $\neq 0$ then CHARM-EXTEND (New N)
  9.  $\mathcal{C} = \mathcal{C} \cup X$ //if X is not subsumed
CHARM-PROPERTY (Nodes, New N)
  10. if ($|Y| \geq$minsup) then
  11.     if $t(X_i) = t(X_j)$ then //property 1
  12.         Remove $X_j$ from Nodes
  13.         Replace all $X_i$ with X
  14.     else if $t(X_i) \subset t(X_j)$ then //property 2
  15.         Replace all $X_i$ with X
  16.     else if $t(X_i) \supset t(X_j)$ then //property 3
  17.         Remove $X_j$ from Nodes
  18.         Add $X \times Y$ to NewN
  19.     else if $t(X_i) \neq t(X_j)$ then //property 4
  20.         Add $X \times Y$ to NewN
```

Figure 2. CHARM Algorithm

CHARM-EXTEND is responsible for testing each branch for viability. It extracts each itemset-tidset pair in the current node set *Nodes* ($X_i \times t(X_i)$), Line 3), and combines it with the other pairs that come after it ($X_i \times t(X_i)$) Line 5) according to the total order f. The combination of the two itemset-tidset pairs is computed in Line 6. The routine CHARM-PROPERTY tests the resulting set for required support and applies the four properties discussed above. Note that this routine may modify the current node set by deleting itemset-tidset pairs that are already contained in other pairs. It also inserts the newly generated children frequent pairs in the set of new nodes *NewN*. If this set is non-empty it is recursively processed in depth-first manner (Line 8). The possibly extended itemset **X** of X_i is then inserted in the set of closed itemsets, since it cannot be processed further; at this stage any closed itemset containing X_i has already been generated. The control then returns to Line 3 to process the next (unpruned) branch. The routine CHARM-PROPERTY simply tests if a new pair is frequent, discarding it if it is not. It then tests each of the four basic properties of itemset-tidset pairs, extending existing itemsets, removing some subsumed branches from the current set of nodes, or inserting new pairs in the node set for the next depth-first) step [1].

6. GENERATING ASSOCIATION RULES

The set of all association rules can rapidly grow to be unwieldy. The larger the set of frequent itemsets the more the number of rules presented to the user. However, since most of these rules turn out to be redundant, it is not necessary to mine rules from all frequent itemsets. In fact, it is sufficient to consider only the rules among closed frequent itemsets. Given a closed frequent itemset L, rule generation examines each non-empty subset **a** and generates the rule **a** ⇒ (**L** – **a**) with support = support (**L**) and confidence = support (**L**)/support (**a**). This computation can efficiently be done by examining the largest subsets of L first and only proceeding to smaller subsets if the generated rules have the required minimum confidence [4].

7. EXPERIMENTAL ANALYSIS OF THE SYSTEM

Experimental analysis of a sample dataset is described though the system can efficiently work with the sale dataset and the supply chain dataset, each of which has about 2000 transactions. Figure 3 shows the sample dataset, which is the input to the system. Figure 4 shows the rules generated from the set of all non-closed frequent itemsets. Figure 5 shows association rules which are generated by the system from the set of all closed frequent itemsets. There are 28 rules, which can be generated from the set of all non-closed frequent itemsets while 32 rules can be generated from the set of all closed frequent itemsets. As seen in the figures, all of the rules that can be obtained from the set of all non-closed frequent itemsets are all equivalent to those rules which can be obtained from the set of all closed frequent itemsets. This is the same in the case of sale and supply chain datasets.

Sample Dataset	
TRANSACTION	ITEMS
T1	A,C,T,W
T2	C,D,W
T3	A,C,T,W
T4	A,C,D,W
T5	A,C,D,T,W
T6	C,D,T

Figure 3. Sample Data

CTW	
CT=>W	75%
CW=>T	60%
TW=>C	100%
W=>CT	60%
T=>CW	75%
C=>TW	50%

ACT	
AC=>T	75%
AT=>C	100%
CT=>A	75%
A=>CT	75%
C=>AT	50%
T=>AC	75%

ATW	
AT=>W	100%
AW=>T	75%
TW=>A	100%
A=>TW	75%
T=>AW	75%
W=>AT	60%

TW	
T=>W	75%
W=>T	60%

DW	
D=>W	75%
W=>D	60%

AC	
A=>C	100%
C=>A	67%

AW	
A=>W	100%
W=>A	80%

AT	
T=>A	75%
A=>T	75%

Figure 4. Association Rules Generated From Non-Closed Frequent Itemsets

ACTW	
ACT=>W	100%
ACW=>T	100%
ATW=>C	100%
CTW=>A	100%
AC=>TW	75%
AT=>CW	100%
AW=>CT	75%
TW=>AC	100%
CW=>AT	60%
CT=>AW	75%
A=>CTW	75%
C=>ATW	50%
T=>ACW	75%
W=>ACT	60%

ACW	
AC=>W	100%
AW=>C	100%
CW=>A	80%
A=>CW	100%
C=>AW	67%
W=>AC	80%

CDW	
CD=>W	75%
CW=>D	60%
DW=>C	100%
C=>DW	50%
D=>CW	75%
W=>CD	60%

CT	
C=>T	66.67%
T=>C	100%

CD	
C=>D	66.67%
D=>C	100%

CW	
C=>W	83.33%
W=>C	100%

Figure 5. Association Rules Generated From Closed Frequent Itemsets

8. CONCLUSION

This paper describes the closed association rule mining concepts and techniques used in implementing this generator. It presents how closed association rule mining differs from traditional association rule mining. It also describes how CHARM algorithm works. In addition, this paper describes the technical feasibility of closed association rule mining concepts and techniques in general. The main advantage of the generator is that it is not tied to any specific application. It can be used with datasets of various application domains. As a result, it can be used by a wide variety of business applications. In addition, rule generating mechanism used in the system is based on the concepts and techniques of closed association rule mining. Thus, the overall performance of the system is much better than that of systems, which are implemented using traditional association rule mining concepts and techniques.

REFERENCES

[1] Mohammed J. Zaki and Ching-JuiHsiao, *CHARM An Efficient Algorithm for Closed Association Rule Mining*, Computer Science Department, Rensselaer Polytechnic Institute, 2001.

[2] Jiawei Han, Micheline Kamber, *Data Mining Concepts and Techniques*, Morgan Kaufmann, San Francisco, 2001.

[3] B. A. Davey and H. A. Priestley. *Introduction to Lattices and Order*. Cambridge University Press, 1990.

[4] A.M.J. Md. Zubair Rahman and P. Balasubramanie, *Weighted Support Association Rule Mining using Closed Itemset Lattices in Parallel*, IJCSNS International Journal of Computer Science and Network Security, VOL.9 No.3, March 2009.

[5] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, and A. Inkeri Verkamo. Fast discovery of association rules. In U. Fayyad and et al, editors, *Advances in Knowledge Discovery and Data Mining*, pages 307–328. AAAI Press, Menlo Park, CA, 1996.

[6] R. J. Bayardo. Efficiently mining long patterns from databases. In *ACM SIGMOD Conf. Management of Data*, June 1998.

[7] D-I. Lin and Z. M. Kedem. Pincer-search: A new algorithm for discovering the maximum frequent set. In 6th *Intl. Conf. Extending Database Technology*, March 1998