Data Mining and Knowledge Discovery, 9, 223–248, 2004

Mining Non-Redundant Association Rules*

MOHAMMED J. ZAKI zaki@cs.rpi.edu
Computer Science Department, Rensselaer Polytechnic Institute, Troy NY 12180, USA

Editors: Fayyad, Mannila, Ramakrishnan

Received July 24, 2001; Revised July 6, 2003

Abstract. The traditional association rule mining framework produces many redundant rules. The extent of redundancy is a lot larger than previously suspected. We present a new framework for associations based on the concept of *closed* frequent itemsets. The number of non-redundant rules produced by the new approach is exponentially (in the length of the longest frequent itemset) smaller than the rule set from the traditional approach. Experiments using several "hard" as well as "easy" real and synthetic databases confirm the utility of our framework in terms of reduction in the number of rules presented to the user, and in terms of time.

Keywords: association rule mining, frequent closed itemsets, formal concept analysis

1. Introduction

Association rule discovery, a successful and important mining task, aims at uncovering all frequent patterns among transactions composed of data attributes or items. Results are presented in the form of rules between different sets of items, along with metrics like the joint and conditional probabilities of the antecedent and consequent, to judge a rule's importance.

It is widely recognized that the set of association rules can rapidly grow to be unwieldy, especially as we lower the frequency requirements. The larger the set of frequent itemsets the more the number of rules presented to the user, many of which are redundant. This is true even for sparse datasets, but for dense datasets it is simply not feasible to mine all possible frequent itemsets, let alone to generate rules, since they typically produce an exponential number of frequent itemsets; finding long itemsets of length 20 or 30 is not uncommon (Bayardo, 1998).

Prior research has mentioned that the traditional association rule mining framework produces too many rules, but the extent of redundancy is a lot larger than previously suspected. More concretely, the number of redundant rules are exponential in the length of the longest frequent itemset. We present a new framework for association rule mining based on the concept of *closed* frequent itemsets. The set of all closed frequent itemsets can be orders of magnitude smaller than the set of all frequent itemsets, especially for real (dense) datasets.

*This work was supported in part by NSF CAREER Award IIS-0092978, DOE Career Award DE-FG02-02ER25538 and NSF NGSP grant EIA-0103708.

At the same time, we don't loose any information; the closed itemsets uniquely determine the set of all frequent itemsets and their *exact* frequency. Note that using the maximal frequent itemsets results in loss of information, since subset frequency is not available. We show that the new framework produces exponentially (in the length of the longest frequent itemset) fewer rules than the traditional approach, again without loss of information. Our framework allows us to mine even dense datasets, where it is not feasible to find all frequent itemsets.

Experiments using several "hard" or dense, as well as sparse databases confirm the utility of our framework in terms of reduction in the number of rules presented to the user, and in terms of time. We show that closed itemsets can be found in a fraction of the time it takes to mine all frequent itemsets (with improvements of more than 100 times), and the number of rules returned to the user can be smaller by a factor of 3000 or more! (the gap widens for lower frequency values).

1.1. Related work

There has been a lot of research in developing efficient algorithms for mining frequent itemsets (Agrawal et al., 1996; Bayardo, 1998; Brin et al., 1997; Lin and Kedem, 1998; Savasere et al., 1995; Zaki et al., 1997). Most of these algorithms enumerate all frequent itemsets. Using these for rule generation produces many redundant rules, as we will show later. Some methods only generate maximal frequent itemsets (Bayardo, 1998; Lin and Kedem, 1998). Maximal itemsets cannot be used for rule generation, since support of subsets is required for confidence computation. While it is easy to make one more data scan to gather the supports of all subsets, we still have the problem of many redundant rules. Further, for all these methods it is simply not possible to find rules in dense datasets which may easily have frequent itemsets of length 20 and more (Bayardo, 1998). In contrast the set of *closed* frequent itemsets can be orders of magnitude smaller than the set of all frequent itemsets, and it can be used to generate rules even in dense domains.

In general, most of the association mining work has concentrated on the task of mining frequent itemsets. Rule generation has received very little attention. There has been some work in pruning discovered association rules by forming rule covers (Toivonen et al., 1995). Other work addresses the problem of mining interesting association rules (Klemettinen et al., 1994; Bayardo and Agrawal, 1999; Liu et al., 1999; Ng et al., 1998). The approach taken is to incorporate user-specified constraints on the kinds of rules generated or to define objective metrics of interestingness. As such these works are complimentary to our approach here. Furthermore, they do not address the issue of rule redundancy.

A preliminary study of the idea of using closed frequent itemsets to generate rules was presented by us in Zaki and Ogihara (1998). This paper substantially improves on those ideas, and also presents experimental results to support our claims. Independently, Pasquier et al. (1999a, 1999b) have also used closed itemsets for association mining. However, they mainly concentrate on the discovery of frequent closed itemsets, and do not report any experiments on rule mining. We on the other hand are specifically interested in generating a smaller non-redundant rule set, after mining the frequent closed itemsets. Furthermore, we recently proposed the CHARM algorithm (Zaki and Hsiao, 2002) for mining all closed frequent itemsets. This algorithm outperforms, by orders of magnitude, the AClose method

proposed by Pasquier et al. (1999b), as well as the Apriori (Agrawal et al., 1996) method for mining all frequent itemsets. CHARM was also shown to outperform other recent closed set mining algorithms like Closet (Pei et al., 2000), Mafia (Burdick et al., 2001) and Pascal (Bastide et al., 2000).

The notion of closed frequent sets has its origins in the elegant mathematical framework of formal concept analysis (FCA). A number of algorithms have been proposed within FCA for generating all the closed sets of a binary relation (Ganter and Wille, 1999). However, these methods have only been tested on very small datasets. Further, these algorithms generate all the closed sets, and thus have to be adapted to enumerate only the frequent concepts. The foundations of rule generation (in FCA) were studied in Luxenburger (1991), but no experimentation on large sets was done. They also proposed an approach for obtaining a generating set of rules, but did not consider frequent rules, and no algorithms were proposed. Our characterization of the non-redundant rule set of association rules is different, and we also present an experimental verification. Other work has extended the FCA approach to incorporate incremental rule mining (Godin et al., 1995).

Like our earlier work in Zaki and Ogihara (1998), Taouil et al. (2000) proposed a basis for association rules based on the work of Guigues and Duquenne (1986) and Luxenburger (1991). The work by Bastide et al. (2000a) has independently addressed the problem of extracting minimal association rules. Their definition of minimal rules is different from ours, but based on similar ideas. The different definition of minimal and non-redundant rules leads to different, mutually complementary, notions of smaller association rule sets.

2. Association rules

The association mining task can be stated as follows: Let $\mathcal{I} = \{1, 2, \ldots, m\}$ be a set of items, and let $\mathcal{T} = \{1, 2, \ldots, n\}$ be a set of transaction identifiers or *tids*. The input database is a binary relation $\delta \subseteq \mathcal{I} \times \mathcal{T}$. If an item i occurs in a transaction t, we write it as $(i, t) \in \delta$, or alternately as $i\delta t$. Typically the database is arranged as a set of transactions, where each transaction contains a set of items. For example, consider the database shown in figure 1, used as a running example in this paper. Here $\mathcal{I} = \{A, C, D, T, W\}$, and $\mathcal{T} = \{1, 2, 3, 4, 5, 6\}$. The second transaction can be represented as $\{C\delta 2, D\delta 2, W\delta 2\}$; all such pairs from all transactions, taken together form the binary relation δ. A set $X \subseteq \mathcal{I}$ is called an *itemset*, and a set $Y \subseteq \mathcal{T}$ is called a *tidset*. For convenience we write an itemset $\{A, C, W\}$ as ACW, and a tidset $\{2, 4, 5\}$ as 245.

For an itemset X, we denote its corresponding tidset as $t(X)$, i.e., the set of all tids that contain X as a subset. For a tidset Y, we denote its corresponding itemset as $i(Y)$, i.e., the set of items that appear in every transaction (tid) in Y. Note that $t(X) = \bigcap_{x \in X} t(x)$, and $i(Y) = \bigcap_{y \in Y} i(y)$. For example, $t(ACW) = t(A) \cap t(C) \cap t(W) = 1345 \cap 123456 \cap 12345 = 1345$ and $i(12) = i(1) \cap i(2) = ACTW \cap CDW = CW$. We use the notation $X \times t(X)$ to refer an itemset and the corresponding tidset where it appears. The properties of the mappings $t(X)$ and $i(Y)$ will be studied in detail in Section 3.

The *support* of an itemset X, denoted $\sigma(X)$, is the number of transactions in which it occurs as a subset, i.e., $\sigma(X) = |t(X)|$. An itemset is *frequent* if its support $\sigma(X) \geq minsup$, where *minsup* is a user-specified minimum support threshold.

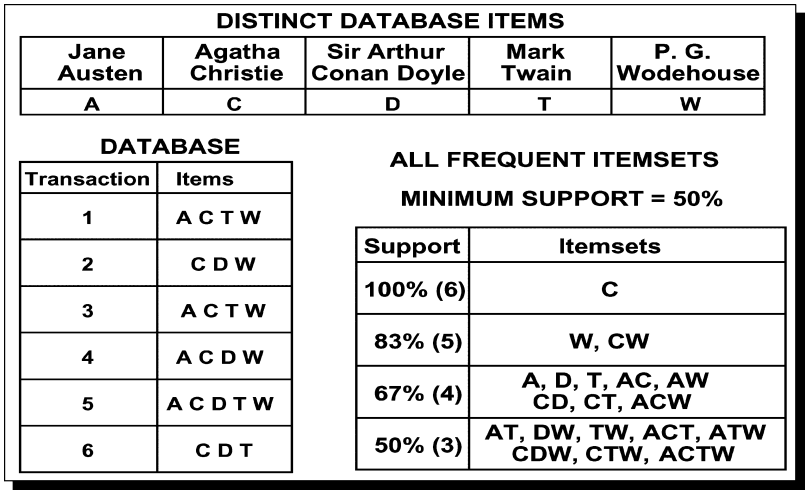

Figure 1. Generating frequent itemsets.

An *association rule* is an expression $A \xrightarrow{q,p} B$, where A and B are itemsets. The *support* of the rule is $q = \sigma(A \cup B) = |t(A \cup B)|$ (i.e., the joint probability of a transaction containing both A and B), and the *confidence* $p = \frac{\sigma(A \cup B)}{\sigma(A)} = \frac{|t(A \cup B)|}{|t(A)|}$ (i.e., the conditional probability that a transaction contains B, given that it contains A). A rule is frequent if the itemset $A \cup B$ is frequent (i.e., $q \geq minsup$). A rule is *confident* if $p \geq minconf$, where where *minconf* is a user-specified minimum threshold. When support is understood, we omit q and write a rule as $A \xrightarrow{p} B$.

Association rule mining consists of two steps (Agrawal et al., 1996): (1) Find all frequent itemsets, and (2) Generate high confidence rules.

2.1. *Finding frequent itemsets*

This step is computationally and I/O intensive. Consider figure 1, which shows a bookstore database with six customers who buy books by different authors. It shows all the frequent itemsets with $minsup = 50\%$ (i.e., 3 transactions). *ACTW* and *CDW* are the maximal frequent itemsets (i.e., not a subset of any other frequent itemset).

Let $|\mathcal{I}| = m$ be the number of items. The search space for enumeration of all frequent itemsets is 2^m, which is exponential in m. One can prove that the problem of finding a frequent set of a certain size is NP-Complete, by reducing it to the balanced bipartite clique problem, which is known to be NP-Complete (Zaki and Ogihara, 1998). However, if we assume that there is a bound on the transaction length, the task of finding all frequent itemsets is essentially linear in the database size, since the overall complexity in this case is given as $O(r \cdot n \cdot 2^l)$, where $|\mathcal{T}| = n$ is the number of transactions, l is the length of the longest frequent itemset, and r is the number of maximal frequent itemsets.

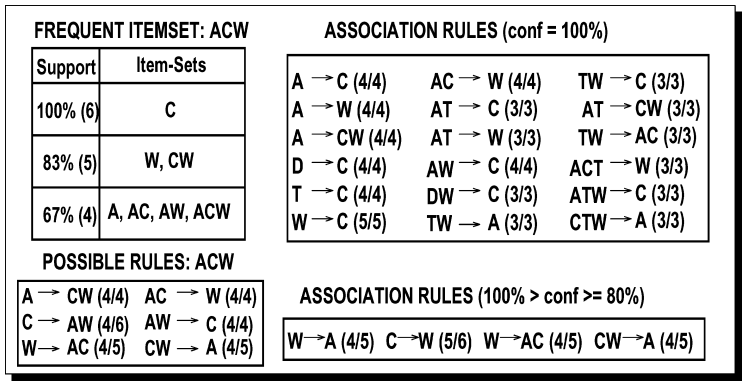

Figure 2. Generating Confident Rules (for a rule $X \longrightarrow Y(n/d)$, its support is given by n and its confidence by n/d).

2.2. *Generating confident rules*

This step is relatively straightforward; rules of the form $Y \xrightarrow{p} X - Y$, are generated for all frequent itemsets X, for all $Y \subset X$, $Y \neq \emptyset$, and provided $p \geq minconf$. Since X is frequent, the rule is guaranteed to be frequent. For example, from the frequent itemset ACW we can generate 6 possible rules (all of them have support of 4): $A \xrightarrow{1.0} CW$, $C \xrightarrow{0.67} AW$, $W \xrightarrow{0.8} AC$, $AC \xrightarrow{1.0} W$, $AW \xrightarrow{1.0} C$, and $CW \xrightarrow{0.8} A$. This process is also shown pictorially in figure 2. Notice that we need access to the support of all subsets of ACW to generate rules from it. To obtain all possible rules we need to examine each frequent itemset and repeat the rule generation process shown above for ACW. Figure 2 shows the set of all other association rules with confidence above or equal to $minconf = 80\%$.

For an itemset of size k there are $2^k - 2$ potentially confident rules that can be generated. This follows from the fact that we must consider each subset of the itemset as an antecedent, except for the empty and the full itemset. The complexity of the rule generation step is thus $O(f \cdot 2^l)$, where f is the number of frequent itemsets, and l is the longest frequent itemset.

3. Closed frequent itemsets

In this section we describe the concept of closed frequent itemsets, and show that this set is necessary and sufficient to capture all the information about frequent itemsets, and has smaller cardinality than the set of all frequent itemsets. We refer the reader to Davey and Priestley (1990) and Ganter and Wille (1999) for more details on lattice theory and formal concept analysis, respectively.

Let $(P, \leq)$ be an ordered set with the binary relation $\leq$, and let S be a subset of P. An element $u \in P$ ($l \in P$) is an *upper bound* (*lower bound*) of S if $s \leq u$ ($s \geq l$) 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 $S = \{x, y\}$, we also write $x \vee y$ for

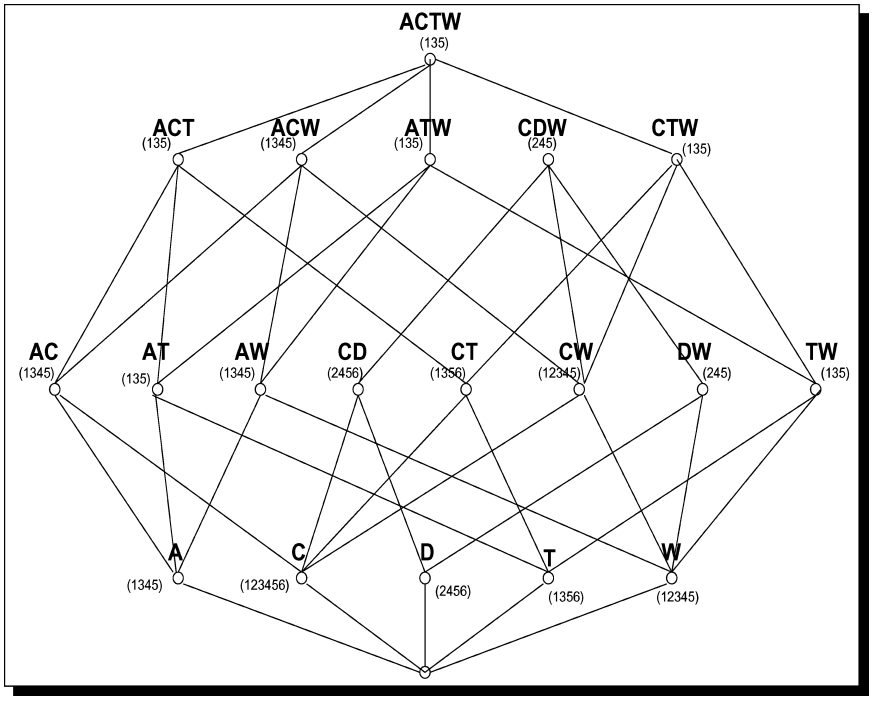

Figure 3. Frequent itemsets (corresponding tidsets are shown in brackets).

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 (Davey and Priestley, 1990).

Let $\mathcal{P}$ denote the power set of S (i.e., the set of all subsets of S). The ordered set $(\mathcal{P}(S), \subseteq)$ 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 $(\mathcal{P}(\mathcal{I}), \subseteq)$, the set of all possible itemsets, and $(\mathcal{P}(\mathcal{T}), \subseteq)$, the set of all possible tidsets are both complete lattices. Figure 3 shows the lattice[1] of all frequent itemsets we found in our example database.

Let the binary relation $\delta \subseteq \mathcal{I} \times \mathcal{T}$ be the input database for association mining. Let $X \subseteq \mathcal{I}$, and $Y \subseteq \mathcal{T}$. The following two mappings together define a *Galois connection* between $\mathcal{P}(\mathcal{I})$ and $\mathcal{P}(\mathcal{T})$ (Ganter and Wille, 1999):

1. $t : \mathcal{I} \mapsto \mathcal{T},\ t(X) = \{y \in \mathcal{T} \mid \forall x \in X,\ x\delta y\}$
2. $i : \mathcal{T} \mapsto \mathcal{I},\ i(Y) = \{x \in \mathcal{I} \mid \forall y \in Y,\ x\delta y\}$

Figure 4 illustrates the two mappings. Recall that the mapping $t(X)$ is the set of all transactions (tidset) which contain the itemset X, and $i(Y)$ is the itemset that is contained in all the transactions in Y. We denote an itemset X and its corresponding tidset $t(X)$ as $X \times t(X)$. Similarly a tidset Y and its corresponding itemset $i(Y)$ is denoted as $i(Y) \times Y$. For example,

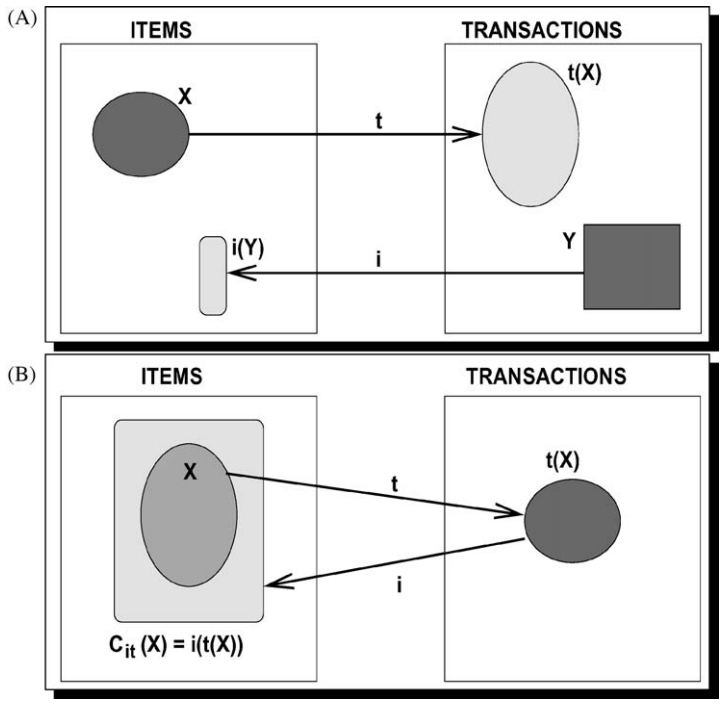

Figure 4. (A) Galois connection and (B) Closed itemset: Round-trip.

$t(ACW) = 1345$, and $i(245) = CDW$. In terms of individual elements $t(X) = \bigcap_{x \in X} t(x)$, and $i(Y) = \bigcap_{y \in Y} i(y)$. For example $t(ACW) = t(A) \cap t(C) \cap t(W) = 1345 \cap 123456 \cap 12345 = 1345$. Also $i(245) = i(2) \cap i(4) \cap i(5) = CDW \cap ACDW \cap ACDTW = CDW$.

The Galois connection satisfies the following properties (Ganter and Wille, 1999) (where $X, X_1, X_2 \in \mathcal{P}(\mathcal{I})$ and $Y, Y_1, Y_2 \in \mathcal{P}(\mathcal{T})$):

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 $Y \subseteq t(i(Y))$.

For example, for $ACW \subseteq ACTW$, $t(ACW) = 1345 \supseteq 135 = t(ACTW)$. For $245 \subseteq 2456$, $i(245) = CDW \supseteq CD = i(2456)$. Also, $AC \subseteq i(t(AC)) = i(1345) = ACW$.

Let S be a set. A function $c : \mathcal{P}(S) \mapsto \mathcal{P}(S)$ is a *closure operator* on S if, for all $X, Y \subseteq S$, c satisfies the following properties (Ganter and Wille, 1999):

1. Extension: $X \subseteq c(X)$
2. Monotonicity: if $X \subseteq Y$, then $c(X) \subseteq c(Y)$
3. Idempotency: $c(c(X)) = c(X)$. A subset X of S is called *closed* if $c(X) = X$.

Lemma 3.1 (Ganter and Wille, 1999). *Let $X \subseteq \mathcal{I}$ and $Y \subseteq \mathcal{T}$. Let $c_{it}(X)$ denote the composition of the two mappings $i \circ t(X) = i(t(X))$. Dually, let $c_{ti}(Y) = t \circ i(Y) = t(i(Y))$. 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.*

We define a *closed itemset* as an itemset X that is the same as its closure, i.e., $X = c_{it}(X)$. For example the itemset ACW is closed. A *closed tidset* is a tidset $Y = c_{ti}(Y)$. For example, the tidset 1345 is closed.

The mappings c_{it} and c_{ti}, being closure operators, satisfy the three properties of extension, monotonicity, and idempotency. We also call the application of $i \circ t$ or $t \circ i$ a *round-trip*. Figure 4 illustrates this round-trip starting with an itemset X. For example, let $X = AC$, then the extension property say that X is a subset of its closure, since $c_{it}(AC) = i(t(AC)) = i(1345) = ACW$. Since $AC \neq c_{it}(AC) = ACW$, we conclude that AC is not closed. On the other hand, the idempotency property say that once we map an itemset to the tidset that contains it, and then map that tidset back to the set of items common to all tids in the tidset, we obtain a closed itemset. After this no matter how many such round-trips we make we cannot extend a closed itemset. For example, after one round-trip for AC we obtain the closed itemset ACW. If we perform another round-trip on ACW, we get $c_{it}(ACW) = i(t(ACW)) = i(1345) = ACW$.

For any closed itemset X, there exists a closed tidset given by Y, with the property that $Y = t(X)$ and $X = i(Y)$ (conversely, for any closed tidset there exists a closed itemset). We can see that X is closed by the fact that $X = i(Y)$, then plugging $Y = t(X)$, we get $X = i(Y) = i(t(X)) = c_{it}(X)$, thus X is closed. Dually, Y is closed. For example, we have seen above that for the closed itemset ACW the associated closed tidset is 1345. Such a closed itemset and closed tidset pair $X \times Y$ is called a *concept*.[2]

A concept $X_1 \times Y_1$ is a *subconcept* of $X_2 \times Y_2$, denoted as $X_1 \times Y_1 \leq X_2 \times Y_2$, iff $X_1 \subseteq X_2$ (iff $Y_2 \subseteq Y_1$). Let $\mathcal{B}(\delta)$ denote the set of all possible concepts in the database. Then the ordered set $(\mathcal{B}(\delta), \leq)$ is a complete lattice, called the *Galois* lattice. For example, figure 5 shows the Galois lattice for our example database, which has a total of 10 concepts. The least element is $C \times 123456$ and the greatest element is $ACDTW \times 5$. The mappings between the closed pairs of itemsets and tidsets are anti-isomorphic, i.e., concepts with large cardinality itemsets have small tidsets, and vice versa.

The concept generated by a single item $x \in \mathcal{I}$ is called an *item concept*, and is given as $\mathcal{C}_i(x) = c_{it}(x) \times t(x)$. Similarly, the concept generated by a single transaction $y \in \mathcal{T}$ is called a *tid concept*, and is given as $\mathcal{C}_t(y) = i(y) \times c_{ti}(y)$. For example, the item concept $\mathcal{C}_i(A) = i(t(A)) \times t(A) = i(1345) \times 1345 = ACW \times 1345$. Further, the tid concept $\mathcal{C}_t(2) = i(2) \times t(i(2)) = CDW \times t(CDW) = CDW \times 245$. An item concept x is the smallest item that generates the closed set $c_{it}(x)$, and a tid concept y is the smallest tid that generates the closed tidset $c_{it}(y)$ or the closed itemset $i(y)$. These two are useful for labeling the closed itemset lattice.

In figure 5 if we relabel each node with the item concept or tid concept that it is equivalent to, then we obtain a lattice with *minimal labeling* (Ganter and Wille, 1999), with item or tid labels, as shown in the figure in bold letters. Such a relabeling reduces clutter in the lattice diagram, which provides an excellent way of visualizing the structure of the patterns and

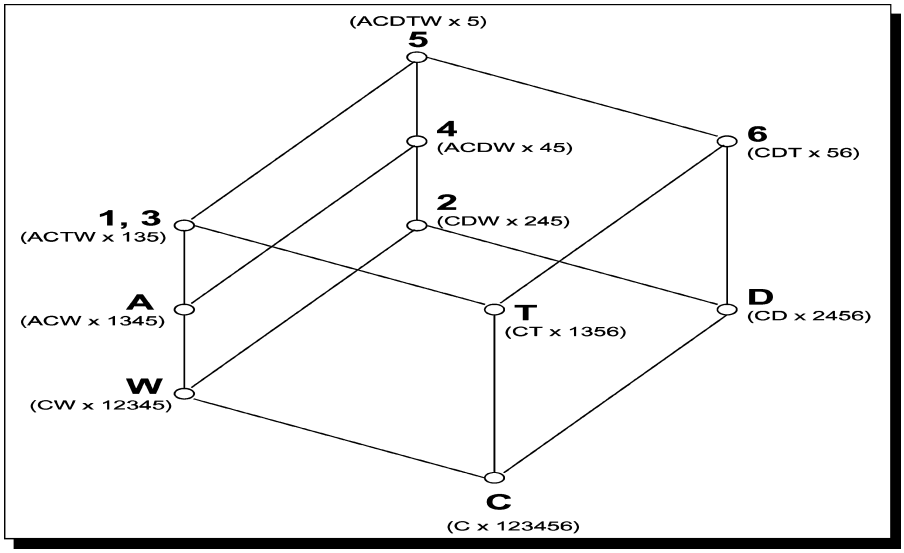

Figure 5. Galois lattice of concepts.

relationships that exist between items. We shall see its benefit in the next section when we talk about high confidence rules extraction.

It is easy to reconstruct the concepts from the minimal labeling. Consider the tid concept $\mathcal{C}_t(2) = X \times Y$. To obtain the closed itemset X, we append all item labels reachable below it. Conversely, to obtain the closed tidset Y we append all labels reachable above $\mathcal{C}_t(2)$. Since W, D and C are all the labels reachable by a path below it, $X = CDW$ forms the closed itemset. Since 4 and 5 are the only labels reachable above $\mathcal{C}_t(2)$, $Y = 245$; this gives us the concept $CDW \times 245$, which matches the concept shown in the figure.

3.1. Frequent closed itemsets vs. frequent itemsets

We begin this section by defining the join and meet operation on the concept lattice (see Ganter and Wille, 1999 for the formal proof): The set of all concepts in the database relation δ, given by $(\mathcal{B}(\delta), \leq)$ is a (complete) lattice with join and meet given by

$$\textit{join}: (X_1 \times Y_1) \vee (X_2 \times Y_2) = c_{it}(X_1 \cup X_2) \times (Y_1 \cap Y_2)$$
$$\textit{meet}: (X_1 \times Y_1) \wedge (X_2 \times Y_2) = (X_1 \cap X_2) \times c_{ti}(Y_1 \cup Y_2)$$

For the join and meet of multiple concepts, we simply take the unions and joins over all of them. For example, consider the join of two concepts, $(ACDW \times 45) \vee (CDT \times 56) = c_{it}(ACDW \cup CDT) \times (45 \cap 56) = ACDTW \times 5$. On the other hand their meet is given as, $(ACDW \times 45) \wedge (CDT \times 56) = (ACDW \cap CDT) \times c_{ti}(45 \cup 56) = CD \times c_{ti}(456) = CD \times 2456$. Similarly, we can perform multiple concept joins or meets; for example, $(CT \times$

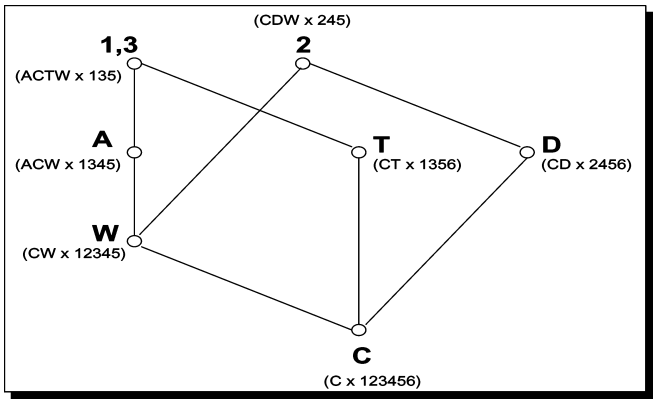

Figure 6. Frequent concepts.

$1356) \vee (CD \times 2456) \vee (CDW \times 245) = c_{it}(CT \cup CD \cup CDW) \times (1356 \cap 2456 \cap 245) = c_{it}(CDTW) \times 5 = ACDTW \times 5$.

We define the support of a closed itemset X or a concept $X \times Y$ as the cardinality of the closed tidset $Y = t(X)$, i.e, $\sigma(X) = |Y| = |t(X)|$. A closed itemset or a concept is *frequent* if its support is at least *minsup*. Figure 6 shows all the frequent concepts with *minsup* = 50% (i.e., with tidset cardinality at least 3). All frequent itemsets can be determined by the join operation on the frequent item concepts. For example, since join of item concepts D and T, $\mathcal{C}_i(D) \vee \mathcal{C}_i(T)$, doesn't exist, DT is not frequent. On the other hand, $\mathcal{C}_i(A) \vee \mathcal{C}_i(T) = ACTW \times 135$, thus AT is frequent. Furthermore, the support of AT is given by the cardinality of the resulting concept's tidset, i.e., $\sigma(AT) = |t(AT)| = |135| = 3$.

Lemma 3.2. *The support of an itemset X is equal to the support of its closure, i.e., $\sigma(X) = \sigma(c_{it}(X))$.*

Proof: The support of an itemset X is the number of transactions where it appears, which is exactly the cardinality of the tidset $t(X)$, i.e., $\sigma(X) = |t(X)|$. Since $\sigma(c_{it}(X)) = |t(c_{it}(X))|$, to prove the theorem, we have to show that $t(X) = t(c_{it}(X))$.

Since c_{ti} is closure operator, it satisfies the extension property, i.e., $t(X) \subseteq c_{ti}(t(X)) = t(i(t(X))) = t(c_{it}(X))$. Thus $t(X) \subseteq t(c_{it}(X))$. On the other hand since c_{it} is also a closure operator, $X \subseteq c_{it}(X)$, which in turn implies that $t(X) \supseteq t(c_{it}(X))$, due to property 1) of Galois connections. Thus $t(X) = t(c_{it}(X))$. □

This lemma, independently reported in Pasquier et al. (1999a), states that all frequent itemsets are uniquely determined by the frequent closed itemsets (or frequent concepts). Furthermore, the set of frequent closed itemsets is bounded above by the set of frequent itemsets, and is typically much smaller, especially for dense datasets. For very sparse datasets, in the worst case, the two sets may be equal. To illustrate the benefits of closed itemset mining, contrast figure 3, showing the set of all frequent itemsets, with figure 6, showing the set of all closed frequent itemsets (or concepts). We see that while there are

only 7 closed frequent itemsets, in contrast there are 19 frequent itemsets. This example clearly illustrates the benefits of mining the closed frequent itemsets.[3]

3.2. *Mining closed frequent itemsets*

Here we briefly describe CHARM, an efficient algorithm for mining closed itemsets (Zaki and Hsiao, 2002). We say that two itemsets X, Y of length k belong to the same *prefix equivalence class*, $[P]$, if they share the $k-1$ length prefix P, i.e., $X = Px$ and $Y = Py$, where $x, y \in \mathcal{I}$. More formally, $[P] = \{X_i \mid Px_i, x_i \in \mathcal{I}\}$, is the class of all itemsets sharing P as a common prefix.

In CHARM there is no distinct candidate generation and support counting phase. Rather, counting is simultaneous with candidate generation. For a given prefix class, one performs intersections of the tidsets of all pairs of itemsets in the class, and checks if the resulting tidsets have cardinality at least *minsup*. Each resulting frequent itemset generates a new class which will be expanded in the next step. That is, for a given class of itemsets with prefix P, $[P] = \{Px_1, Px_2, \ldots, Px_n\}$, one performs the intersection of Px_i with all Px_j with $j > i$ to obtain a new class $[Px_i] = [P']$ with elements $P'x_j$ provided the itemset Px_ix_j is frequent. The computation progresses recursively until no more frequent itemsets are produced.

Figure 7 shows the pseudo-code for CHARM, which performs a novel search for *closed* itemsets using subset properties of tidsets. The initial invocation is with the class of frequent single items (the class [∅]). All tidset intersections for pairs of class elements are computed.

```
[∅] = {x_i | x_i ∈ I, σ(x_i) ≥ minsup} // frequent single items
Invoke as Charm ([∅]);

Charm ([P]):
  for all Px_i ∈ [P]
    for all Px_j ∈ [P] with j > i
      R = Px_ix_j and t(R) = t(Px_i) ∩ t(Px_j)
      if |t(R)| ≥ minsup then
        if t(Px_i) = t(Px_j) then Remove Px_j from [P]; Replace all Px_i with R
        if t(Px_i) ⊂ t(Px_j) then Replace all Px_i with R
        if t(Px_i) ⊃ t(Px_j) then Remove Px_j from [P]; Add R to [P']
        if t(Px_i) ≠ t(Px_j) then Add R to [P']
    if [P'] ≠ ∅ then Charm ([P'])
```

Figure 7. Pseudo-code for CHARM.

However in addition to checking for frequency, CHARM eliminates branches that cannot lead to closed sets, and grows closed itemsets using subset relationships among tidsets. There are four cases: if $t(Px_i) \subset t(Px_j)$ or if $t(Px_i) = t(Px_j)$ we replace every occurrence of Px_i with Px_ix_j, since whenever Px_i occurs Px_j also occurs, which implies that $c_{it}(Px_i) \subseteq c_{it}(Px_ix_j)$. If $t(Px_i) \supset t(Px_j)$ then we replace Px_j for the same reason. Finally, R is processed further if $t(Px_i) \neq t(Px_j)$. These four properties allow CHARM to efficiently prune the search tree (for additional details see Zaki and Hsiao, 2002).

4. Non-redundant association rules

Recall that an association rule is of the form $X_1 \xrightarrow{q,p} X_2$, where $X_1, X_2 \subseteq \mathcal{I}$. Its support is given as $q = |t(X_1 \cup X_2)|$, and its confidence is given as $p = P(X_2|X_1) = |t(X_1 \cup X_2)|/|t(X_1)|$. We are interested in finding all high support and high confidence rules. It is widely recognized that the set of such association rules can rapidly grow to be unwieldy. In this section we will show how the closed frequent itemsets help us form a non-redundant set of rules. Thus, only a small and easily understandable set of rules can be presented to the user, who can later selectively derive other rules of interest.

Before we proceed, we need to formally define what we mean by a redundant rule. Let R_i denote the rule $X_1^i \xrightarrow{q_i,p_i} X_2^i$. We say that a rule R_1 is more general than a rule R_2, denoted $R_1 \preceq R_2$ provided that R_2 can be generated by adding additional items to either the antecedent or consequent of R_1, i.e., if $X_1^1 \subseteq X_1^2$ and $X_2^1 \subseteq X_2^2$.

Let $\mathcal{R} = \{R_1, \ldots, R_n\}$ be a set of rules, such that all their supports and confidences are equal, i.e., $q_i = q$ and $p_i = p$ for all $1 \leq i \leq n$. Then we say that a rule R_j is *redundant* if there exists some rule R_i, such that $R_i \preceq R_j$. Since all the rules in the collection $\mathcal{R}$ have the same support and confidence, the simplest rules in the collection should suffice to represent the whole set. Thus the *non-redundant rules* in the collection $\mathcal{R}$ are those that are most general, i.e., those having minimal antecedents and consequents, in terms of subset relation. We now show how to eliminate the redundant association rules, i.e., rules having the same support and confidence as some more general rule.

Lemma 4.1. *The rule $X_1 \xrightarrow{q,p} X_2$ is equivalent to the rule $X_1 \xrightarrow{q_1,p_1} X_1 \cup X_2$, i.e., $q = q_1$ and $p = p_1$.*

Proof: For support we have $q = \sigma(X_1 \cup X_2) = \sigma(X_1 \cup (X_1 \cup X_2)) = q_1$. For confidence, we have $p = \frac{q}{\sigma(X_1)} = \frac{q_1}{\sigma(X_1)} = p_1$. □

Lemma 4.2. *The rule $X_1 \xrightarrow{q,p} X_2$ is equivalent to the rule $c_{it}(X_1) \xrightarrow{q_1,p_1} c_{it}(X_2)$, i.e., $q = q_1$ and $p = p_1$.*

Proof: For support we have $q = \sigma(X_1 \cup X_2) = |t(X_1 \cup X_2)| = |t(X_1) \cap t(X_2)|$. By Lemma 3.2, we get $q = |t(c_{it}(X_1)) \cap t(c_{it}(X_2))|$, since the support of an itemset and its closure is the same. The last expression can be rewritten as $q = |t(c_{it}(X_1) \cup c_{it}(X_2))| = \sigma(c_{it}(X_1) \cup c_{it}(X_2)) = q_1$. For confidence, we have $p = \frac{q}{|t(X_1)|} = \frac{q_1}{|t(X_1)|} = \frac{q_1}{|t(c_{it}(X_1))|} = p_1$. □

Corollary 4.1. *The rule* $X_1 \xrightarrow{q,p} X_2$ *is equivalent to the rule* $c_{it}(X_1) \xrightarrow{q_1,p_1} c_{it}(X_1 \cup X_2)$, *i.e.,* $q = q_1$ *and* $p = p_1$.

Proof: Follows directly by applying Lemma 4.2 to Lemma 4.1. □

Lemma 4.2 says that it suffices to consider rules *only* among the frequent concepts, i.e., a rule between any two itemsets is exactly the same as the rule between their respective closures. Another observation that follows from the concept lattice is that for closed itemsets related by the subset relation, it is sufficient to consider rules among adjacent concepts, since other rules can be inferred by transitivity (Luxenburger, 1991), that is:

Lemma 4.3. Transitivity: *Let* X_1, X_2, X_3 *be frequent closed itemsets, with* $X_1 \subseteq X_2 \subseteq X_3$. *If* $X_1 \xrightarrow{q_1,p_1} X_2$ *and* $X_2 \xrightarrow{q_2,p_2} X_3$, *then* $X_1 \xrightarrow{q_2,(p_1 p_2)} X_3$.

Proof: First let's consider support of $X_1 \longrightarrow X_3$. Since $X_2 \subseteq X_3$, $q_2 = \sigma(X_2 \cup X_3) = |t(X_2 \cup X_3)| = |t(X_3)|$. Since $X_1 \subseteq X_3$, we get $\sigma(X_1 \cup X_3) = |t(X_1 \cup X_3)| = |t(X_3)| = q_2$.

Now let's consider the confidence of $X_1 \longrightarrow X_3$, given as $r = \frac{|t(X_1 \cup X_3)|}{|t(X_1)|} = \frac{|t(X_3)|}{|t(X_1)|}$. From first rule we have $p_1 = \frac{|t(X_1 \cup X_2)|}{|t(X_1)|} = \frac{|t(X_2)|}{|t(X_1)|}$, and from the second rule we have $p_2 = \frac{|t(X_2 \cup X_3)|}{|t(X_2)|} = \frac{|t(X_3)|}{|t(X_2)|}$. Thus $(p_1 \cdot p_2) = \frac{|t(X_2)|}{|t(X_1)|} \cdot \frac{|t(X_3)|}{|t(X_2)|} = \frac{|t(X_3)|}{|t(X_1)|} = r$. □

In the discussion below, we consider two cases of association rules, those with 100% confidence, i.e., with $p = 1.0$, and those with $p < 1.0$.

4.1. Rules with confidence = 100%

Lemma 4.4 (Luxenburger, 1991)**.** *An association rule* $X_1 \xrightarrow{p=1.0} X_2$ *has confidence* $p = 1.0$ *if and only if* $t(X_1) \subseteq t(X_2)$ (*or equivalently if and only if* $c_{it}(X_2) \subseteq c_{it}(X_1)$).

Proof: If $X_1 \xrightarrow{1.0} X_2$, it means that X_2 always occurs in a transaction, whenever X_1 occurs in that transaction. Put another way, the tidset where X_1 occurs must be a subset of the tidset where X_2 occurs. But this is precisely given as $t(X_1) \subseteq t(X_2)$. More formally, the confidence of the rule $X_1 \xrightarrow{p} X_2$ is given as $p = |t(X_1 \cup X_2)|/|t(X_1) = |t(X_1) \cap t(X_2)|/|t(X_1)|$. Since $t(X_1) \subseteq t(X_2)$, we have $p = |t(X_1)|/|t(X_1)| = 1.0$. This also implies that the support of the rule is the same as support of X_1. Finally, by Property 2 of Galois connection, $t(X_1) \subseteq t(X_2)$ implies $c_{it}(X_2) \subseteq c_{it}(X_1)$. □

By Lemma 4.2, $X_1 \xrightarrow{1.0} X_2$ is equivalent to $c_{it}(X_1) \xrightarrow{1.0} c_{it}(X_2)$. By Lemma 4.4 we get $c_{it}(X_2) \subseteq c_{it}(X_1)$. Since a rule between frequent itemsets is equivalent to the rule between their closures, we can assume without loss of generality in the rest of this section that X_1 and X_2 are closed itemsets. The lemma above then says that all 100% confidence rules are those that are directed from a super-concept $(X_1 \times t(X_1))$ to a sub-concept $(X_2 \times t(X_2))$ since it is in precisely these cases that $t(X_1) \subseteq t(X_2)$ (or $c_{it}(X_1) \supseteq c_{it}(X_1)$). Combining this fact with Lemma 4.3, we conclude that it is sufficient to consider 100% confidence

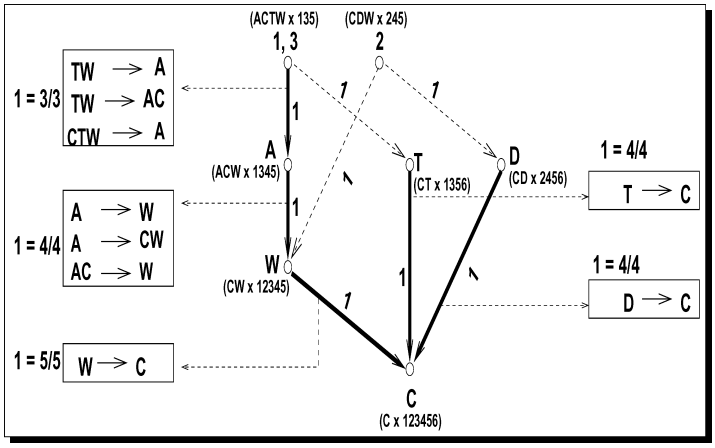

Figure 8. Rules with 100% Confidence (the support of each rule can be taken from the numerator, e.g., for rules with label $1 = 3/3$, their support is 3, and so on): The arrows denote rules between adjacent closed itemsets.

rules only among *adjacent* closed itemsets (not just any closed itemsets), since the other rules can be derived by transitivity.

There are two cases that lead to 100% confidence rules:

1. *Self-rules*: These rules are generated when $X_1 = X_2$ (or equivalently $t(X_1) = t(X_2)$). That is, the rule is directed from a frequent closed itemset to itself.
2. *Down-rules*: These rules are generated when $X_1 \subset X_2$ (or equivalently $t(X_1) \supset t(X_2)$). In other words, the rule is directed from a frequent closed itemsets to its closed proper subset.

Consider the item concepts $\mathcal{C}_i(W) = CW \times 12345$ and $\mathcal{C}_i(C) = C \times 123456$. The rule $W \xrightarrow{1.0} C$ is a 100% confidence rule with support of 5. Note that if we take the itemset closure on both sides of the rule, we obtain $CW \xrightarrow{1.0} C$, i.e., a rule between closed itemsets, but since the antecedent and consequent are not disjoint in this case, we prefer to write the rule as $W \xrightarrow{1.0} C$, although both rules are exactly the same.

Figure 8 shows some of the other rules among adjacent concepts with 100% confidence. We notice that some down-arcs are labeled with more than one rule. In such cases, all rules within a box are equivalent, and we prefer the rule that is most general. For example, consider the rules (all with support 3): $TW \xrightarrow{1.0} A$, $TW \xrightarrow{1.0} AC$, and $CTW \xrightarrow{1.0} A$. $TW \xrightarrow{1.0} A$ is more general than the latter two rules, since the latter two are obtained by adding one (or more) items to either the antecedent or consequent of $TW \xrightarrow{1.0} A$. In fact, we can say that the addition of C to either the antecedent or the consequent has no effect on the support or confidence of the rule. Thus, according to our definition, we say that the other two rules are redundant.

Theorem 4.1. *Let* $\mathcal{R} = \{R_1, \ldots, R_n\}$ *be the set of all possible rules that satisfy the following conditions*:

1. $q_i = q$ *for all* $1 \leq i \leq n$ (*i.e., all rules have the same support*).
2. $p_i = p = 1.0$ *for all* $1 \leq i \leq n$ (*i.e., all rules have 100% confidence*).
3. $I_1 = c_{it}(X_1^i) = c_{it}(X_1^i \cup X_2^i)$, *and* $I_2 = c_{it}(X_2^i)$ *for all* $1 \leq i \leq n$.

Let $\mathcal{R}^G = \{R_i \mid \nexists R_j \in \mathcal{R}, R_j \prec R_i\}$, *denote the most general rules in* $\mathcal{R}$. *Then all rules* $R_i \in \mathcal{R}$ *are equivalent to the rule* $I_1 \stackrel{q,1.0}{\longrightarrow} I_2$, *and all rules in* $\mathcal{R} - \mathcal{R}^G$ *are redundant.*

Proof: Consider any rule $R_i = X_1^i \stackrel{q,1.0}{\longrightarrow} X_2^i$. Then the support of the rule is given as $q = |t(X_1^i \cup X_2^i)|$ and its confidence $p = \frac{|t(X_1^i \cup X_2^i)|}{|t(X_1^i)|}$. By Lemma 4.4 we have $t(X_1^i) \subseteq t(X_2^i)$, giving $|t(X_1^i \cup X_2^i)| = |t(X_1^i) \cap t(X_2^i)| = |t(X_1^i)|$. Thus $q = |t(X_1^i)|$, and $p = \frac{|t(X_1^i)|}{|t(X_1^i)|} = 1$.

Since $t(X_1^i) \subseteq t(X_2^i)$, by Property 2 of Galois connections, we have $i(t(X_1^i)) \supseteq i(t(X_2^i))$, i.e, $c_{it}(X_1^i) \supseteq c_{it}(X_2^i)$. By monotonicity of closure we have $c_{it}(X_1^i) \subseteq c_{it}(X_1^i \cup X_2^i)$. By Lemma 3.2 we also have $\sigma(c_{it}(X_1^i \cup X_2^i)) = \sigma(X_1^i \cup X_2^i)$, i.e., $|t(c_{it}(X_1^i \cup X_2^i))| = |t(X_1^i \cup X_2^i)|$.

The support of the rule $I_1 \longrightarrow I_2$ is given as $|t(I_1 \cup I_2)| = |t(c_{it}(X_1^i \cup X_2^i) \cup c_{it}(X_2^i))| = |t(c_{it}(X_1^i \cup X_2^i))| = |t(X_1^i \cup X_2^i)| = |t(X_1^i) \cap t(X_2^i)| = |t(X_1^i)| = q$. The confidence of the rule $I_1 \longrightarrow I_2$ is given as $\frac{|t(I_1 \cup I_2)|}{|t(I_1)|} = \frac{|t(X_1^i)|}{|t(X_1^i)|} = 1$. □

Let's apply this theorem to the three rules we considered above. For the first rule $c_{it}(TW \cup A) = c_{it}(ATW) = ACTW$. Similarly for the other two rules we see that $c_{it}(TW \cup AC) = c_{it}(ACTW) = ACTW$, and $c_{it}(CTW \cup A) = c_{it}(ACTW) = ACTW$. Thus for these three rules we get the closed itemset $I_1 = ACTW$. By the same process we obtain $I_2 = ACW$. All three rules correspond to the arc between the tid concept $\mathcal{C}_t(1, 3)$ and the item concept $\mathcal{C}_i(A)$. Finally $TW \stackrel{1.0}{\longrightarrow} A$ is the most general rule, and so the other two are redundant.

The set of all non-redundant rules constitutes a *generating set*, i.e., a rule set, from which other 100% confidence rules can inferred (Guigues and Duquenne, 1986; Luxenburger, 1991; Taouil et al., 2000; Zaki and Phoophakdee, 2003). Figure 8 shows the generating set in bold arcs, which includes the 5 most general rules $\{TW \stackrel{1.0}{\longrightarrow} A, A \stackrel{1.0}{\longrightarrow} W, W \stackrel{1.0}{\longrightarrow} C, T \stackrel{1.0}{\longrightarrow} C, D \stackrel{1.0}{\longrightarrow} C\}$ (the down-arcs that have been left out produce rules that cannot be written with disjoint antecedent and consequent. For example, between $\mathcal{C}_t(2)$ and $\mathcal{C}_i(D)$, the most general rule is $DW \stackrel{1.0}{\longrightarrow} D$. Since the antecedent and consequent are not disjoint, as required by definition, we discard such rules).

4.2. Rules with confidence <100%

We now turn to the problem of finding a non-redundant rule set for association rules with confidence less than 100%. By Lemmas 4.1 and 4.3, we need to consider the rules only between *adjacent* concepts. But this time the rules correspond to the up-arcs, instead of the down-rules (or self-rules) for the 100% confidence rules, i.e., the rules go from sub-concepts to super-concepts.

Consider figure 9. The edge between item concepts $\mathcal{C}_i(C)$ and $\mathcal{C}_i(W)$ corresponds to $C \stackrel{0.83}{\longrightarrow} W$. Rules between non-adjacent concepts can be derived by transitivity. For example, for $C \stackrel{p}{\longrightarrow} A$ we can obtain the value of p using the rules $C \stackrel{p_1=5/6}{\longrightarrow} W$ and $W \stackrel{p_2=4/5}{\longrightarrow} A$. We have $p = p_1 p_2 = 5/6 \cdot 4/5 = 2/3 = 0.67$.

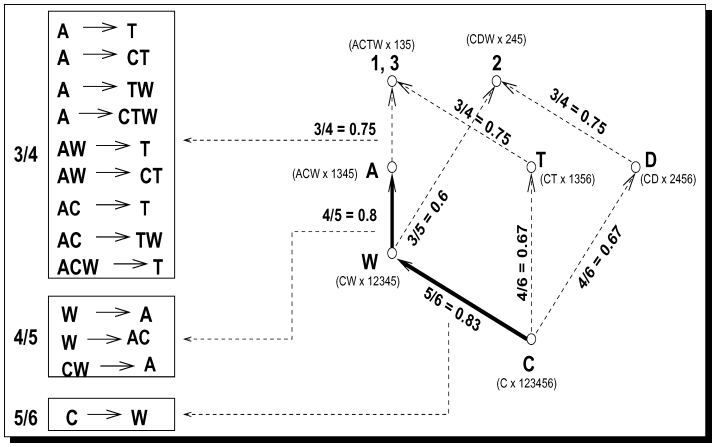

Figure 9. Rules with Confidence <100% (the support of each rule can be taken from the numerator, e.g., for rules with label 3/4, their support is 3, and so on): The arrows denote rules between adjacent closed itemsets.

Theorem 4.2. *Let* $\mathcal{R} = \{R_1, \ldots, R_n\}$ *be the set of all possible rules that satisfy the following conditions*:
1. $q_i = q$ *for all* $1 \leq i \leq n$ (*i.e., all rules have the same support*).
2. $p_i = p < 1.0$ *for all* $1 \leq i \leq n$ (*i.e., all rules have same confidence*).
3. $I_1 = c_{it}(X_1^i)$, *and* $I_2 = c_{it}(X_1^i \cup X_2^i)$ *for all* $1 \leq i \leq n$.
Let $\mathcal{R}^G = \{R_i \mid \nexists R_j \in \mathcal{R}, R_j \prec R_i\}$, *denote the most general rules in* $\mathcal{R}$. *Then all rules* $R_i \in \mathcal{R}$ *are equivalent to the rule* $I_1 \stackrel{q,p}{\longrightarrow} I_2$, *and all rules in* $\mathcal{R} - \mathcal{R}^G$ *are redundant.*

Proof: Consider any rule $R_i = X_1^i \stackrel{p}{\longrightarrow} X_2^i$. Then the support of the rule is given as $q = |t(X_1^i \cup X_2^i)|$ and its confidence as $p = q/d$, $d = |t(X_1^i)|$. We will show that the $I_1 \longrightarrow I_2$ also has support $|t(I_1 \cup I_2)| = q$ and confidence $\frac{|t(I_1 \cup I_2)|}{|t(I_1)|} = q/d$.

Let's consider the denominator first. We have $|t(I_1)| = |t(c_{it}(X_1^i))| = |t(X_1^i)| = d$. Now consider the numerator. We have $|t(I_1 \cup I_2)| = |t(c_{it}(X_1^i) \cup c_{it}(X_1^i \cup X_2^i))|$. Since $X_1^i \subseteq (X_1^i \cup X_2^i)$, we have, from the property of closure operator, $c_{it}(X_1^i) \subseteq c_{it}(X_1^i \cup X_2^i)$. Thus, $|t(I_1 \cup I_2)| = |t(c_{it}(X_1^i \cup X_2^i))| = |t(X_1^i \cup X_2^i)| = q$. □

This theorem differs from that of the 100% confidence rules to account for the up-arcs. Consider the rules produced by the up-arc between item concepts $\mathcal{C}_i(W)$ and $\mathcal{C}_i(A)$. We find that for all three rules, $I_1 = c_{it}(W) = c_{it}(CW) = CW$, and $I_2 = c_{it}(W \cup A) = c_{it}(W \cup AC) = c_{it}(CW \cup A) = ACW$. The support of the rule is given by $|t(I_1 \cup I_2)| = |t(ACW)| = 4$, and the confidence given as $|t(I_1 \cup I_2)|/|t(I_1)| = 4/5 = 0.8$. Finally, since $W \stackrel{0.8}{\longrightarrow} A$ is the most general rule, the other two are redundant. Similarly for the up-arc between $\mathcal{C}_i(A)$ and $\mathcal{C}_t(1, 3)$, we get the general rule $A \stackrel{0.75}{\longrightarrow} T$. The other 8 rules in the box are redundant! The two bold arrows in figure 9 constitute a generating set for all rules with $0.8 \leq p < 1.0$. Due to the transitivity property, we only have to consider arcs with confidence at least *minconf* = 0.8. No other rules can be confident at this level.

By combining the generating set for rules with $p = 1.0$, shown in figure 8 and the generating set for rules with $1.0 > p \geq 0.8$, shown in figure 9, we obtain a generating set for all association rules with *minsup* $= 50\%$, and *minconf* $= 80\%$: $\{TW \xrightarrow{1.0} A, A \xrightarrow{1.0} W, W \xrightarrow{1.0} C, T \xrightarrow{1.0} C, D \xrightarrow{1.0} C, W \xrightarrow{0.8} A, C \xrightarrow{0.83} W\}$.

Using the closed itemset approach we produce 7 rules versus the 22 rules produced in traditional association mining. To see the contrast further, consider the set of all possible association rules we can mine. With *minsup* $= 50\%$, the least value of confidence can be 50% (since the maximum support of an itemset can be 100%, but any frequent subset must have at least 50% support; the least confidence value is thus $50/100 = 0.5$). There are 60 possible association rules versus only 13 in the generating set (5 rules with $p = 1.0$ in figure 8, and 8 rules with $p < 1.0$ in figure 9).

4.3. Non-redundant rule generation

In this section we give the algorithms for non-redundant rule generation. They rely on the concept of minimal generators (Bastide et al., 2000) of a closed itemset. We begin by defining this concept and then present the rule generation algorithms.

4.3.1. Minimal generators. Let X be a closed itemset. We say that an itemset X' is a *generator* of X if and only if (1) $X' \subseteq X$, and (2) $\sigma(X') = \sigma(X)$. X' is called a *proper* generator if $X' \subset X$ (i.e., $X' \neq X$). A proper generator cannot be closed, since by definition, no closed subset of X can have the same support as X. Let $\mathcal{G}(X)$ denote the set of generators of X. We say that $X' \in \mathcal{G}(X)$ is a *minimal* generator if it has no subset in $\mathcal{G}(X)$. Let $\mathcal{G}^{\min}(X)$ denote the set of all minimal generators of X. By definition $\mathcal{G}^{\min}(X) \neq \emptyset$, since if there is no proper generator, X is its own minimal generator. Consider the closed set $ACTW$. The generators of $ACTW$ are $\mathcal{G}(ACTW) = \{AT, TW, ACT, ATW, CTW\}$, and the minimal generators are $\mathcal{G}^{\min}(ACTW) = \{AT, TW\}$.

An algorithm to find minimal generators is shown in figure 10. It is based on the fact that the minimal generators of a closed itemset X are the minimal itemsets that are subsets of X but not a subset of any of X's (immediate) closed subsets. Let $\mathcal{S} = \{X_i \mid (c_{it}(X_i) = X_i) \wedge (X_i \subset X) \wedge (\nexists X_j : (c_{it}(X_j) = X_j) \wedge (X_i \subset X_j \subset X))\}$, be the set of immediate closed subsets of X.

First, any item appearing for the first time in X, given as $I = X - \bigcup_{X_i \in \mathcal{S}} X_i$ is a minimal generator by definition. From the remaining items, i.e., those that appear in subsets of X, we find all minimal generators using an Apriori-style (Agrawal et al., 1996) level-wise procedure. We initialize the candidate generators to be all single items of size one appearing in X's subsets, i.e., $\mathcal{G}_1 = \{i \mid i \in X - I\}$. For any current candidate generator $G \in \mathcal{G}_k$ we test if G is a subset of any itemset in $\mathcal{S}$. If true, G is not a generator for X. If false, then G is a minimal generator, and it is added to $\mathcal{G}^{\min}(X)$, and removed from $\mathcal{G}_k$. After we have seen all $G \in G_k$, we have found all minimal generators of length k. The next step is to generate candidate generators for the next iteration. For each possible generator $G' \in \mathcal{G}_{k+1}$, all its immediate subsets must be present in $\mathcal{G}_k$. Let $G' = i_1 i_2 \ldots i_k i_{k+1}$ be a possible candidate in $\mathcal{G}_{k+1}$. The subset check is done by checking whether the subset G_j of length k obtained by removing item i_j from G' is present in $\mathcal{G}_k$. Since we remove from $\mathcal{G}_k$ any minimal

```
//X is a closed itemset, S = {X_i} is the set of immediate closed subsets of X
MinimalGenerators(X, S):
    G^min(X) = ∅;
    I = X − ⋃_{X_i∈S} X_i; //new items in X
    for all i ∈ I //each item is minimal generator
        G^min(X) = G^min(X) ∪ {i};
    G_1 = {i | i ∈ X − I}; //remaining items
    k = 1;
    while G_k ≠ ∅
        for all G ∈ G_k
            if G ⊈ X_j for all X_j ∈ S
                G^min(X) = G^min(X) ∪ {G};
                G_k = G_k − G;
        //Candidate Minimal Generators for Next Iteration
        G_{k+1} = {G' = i_1 ··· i_k i_{k+1} | ∀1 ≤ j ≤ k + 1,
                    ∃G_j ∈ G_k, G_j = i_1 ··· i_{j−1} i_{j+1} ··· i_{k+1}}
        k = k + 1;
```

Figure 10. Find minimal generators.

generator G, none of G's supersets can ever become candidate generators. We next repeat the whole process with $\mathcal{G}_{k+1}$ as the current set of candidates. The process stops when no more candidates can be generated.

As an example consider $X = ACTW$ again. We have $\mathcal{S} = \{CW, CT, AW\}$. We thus get $I = \emptyset$ and $\mathcal{G}_1 = \{A, C, T, W\}$. We find that all these items are subsets of some set in $\mathcal{S}$, so there can be no single item generators. For the next pass we get $G_2 = \{AC, AT, AW, CT, CW, TW\}$. From these, we find that AT, TW are not subsets of any itemset in $\mathcal{S}$, so we add them to $\mathcal{G}^{\min}(X)$ and remove them from $\mathcal{G}_2$, giving $\mathcal{G}_2 = \{AC, AW, CT, CW\}$. Now for the next pass we get $\mathcal{G}_3 = \{ACW\}$. Since this is a subset of an itemset in $\mathcal{S}$, it cannot be a generator. Finally, we get $\mathcal{G}_4 = \emptyset$, and the computation stops. The final answer is $\mathcal{G}^{\min}(ACTW) = \{AT, TW\}$.

4.3.2. Rule generation. Given any two closed itemsets X_1 and X_2, with $X_1 \subseteq X_2$, figure 11 shows an algorithm to generate all possible non-redundant rules between them, based on the notion of minimal generators. The first step is to compute the minimal generators of X_1 and X_2.

The 100% confidence rules are directed from X_2 to X_1. Every set in $X' \in \mathcal{G}^{\min}(X_2)$ forms a potential *LHS* (left hand side) for a rule, and by definition X' is the most general itemset

```
//X_1, X_2 are closed itemsets, with X_1 ⊆ X_2
GenerateRules (X_1, X_2):
    G_1 = G^min(X_1);
    G_2 = G^min(X_2);
    R = ∅; // non-redundant rule set
    //Rules with 100% confidence
    for all X' ∈ G_2
        for all X'' ∈ G_1
            LHS = X'; RHS = (X'' − X');
            if c_it(RHS) = X_1 and c_it(LHS ∪ RHS) = X_2
                q = σ(X_2);
                R = R ∪ (LHS −q,1.0→ RHS)
    //Rules with confidence < 100%
    for all X' ∈ G_1
        for all X'' ∈ G_2
            LHS = X'; RHS = (X'' − X');
            if c_it(LHS ∪ RHS) = X_2
                q = σ(X_2); p = σ(X_2)/σ(X_1)
                R = R ∪ (LHS −q,p→ RHS)
    //Find general rules
    R^G = {R_i | ∄R_j ∈ R, R_j : (X_1^j −q_j,p_j→ X_2^j) ≺ R_i : (X_1^i −q_i,p_i→ X_2^i) and q_j = q_i and p_j = p_i};
```

Figure 11. Non-redundant rule generation algorithm.

that can represent X_2. Furthermore, as required by Theorem 4.4, $c_{it}(LHS) = c_{it}(X') = X_2$. Likewise every minimal set $X'' \in \mathcal{G}^{\min}(X_1)$ can serve as a potential *RHS* (right hand side). Since we require *RHS* to be disjoint from *LHS*, we set $RHS = X'' - X'$ (thus $LHS \cap RHS = \emptyset$). If the remaining two conditions in Theorem 4.4 are met (i.e., $c_{it}(RHS) = X_1$ and $c_{it}(LHS \cup RHS) = X_2$), we can generate a 100% confidence rule $LHS \longrightarrow RHS$.

Rules with confidence <100% are directed from X_1 to X_2. As before every $X' \in \mathcal{G}^{\min}(X_1)$ forms a potential *LHS*, and every $X'' \in \mathcal{G}^{\min}(X_2)$ forms a potential *RHS* for a rule. To ensure disjointness, we set $RHS = X'' - X'$. As required by Theorem 4.1, $c_{it}(LHS) = c_{it}(X') = X_1$, and if $c_{it}(LHS \cup RHS) = X_2$ then we can generate a <100% confidence rule $LHS \longrightarrow RHS$.

The final step is to find the most general rules among rules having the same support and confidence, $\mathcal{R}^G$, which then represents all possible non-redundant rules between X_1 and X_2.

As an example let $X_1 = ACW$ and $X_2 = ACTW$. We have $\mathcal{G}^{\min}(ACW) = \{A\}$ and $\mathcal{G}^{\min}(ACTW) = \{AT, TW\}$. The possible 100% confidence rules are $AT \longrightarrow (A - AT)$, but since $A - AT = \emptyset$ this rules is not possible. The other possibility is $TW \longrightarrow (A - TW)$, which gives us the rule $TW \xrightarrow{3,1.0} A$. A possible $< 100\%$ confidence rules is $A \longrightarrow$

$(AT - A)$, giving us $A \xrightarrow{3,3/4} T$. The second possibility is $A \longrightarrow (TW - A)$, resulting in the rule $A \xrightarrow{3,3/4} TW$. Thus the possible non-redundant rules between ACW and $ACTW$ are $\mathcal{R} = \{TW \xrightarrow{3,1.0} A, A \xrightarrow{3,3/4} T, A \xrightarrow{3,3/4} TW\}$. Since $A \xrightarrow{3,3/4} T$ is equivalent to (i.e., has same support and confidence) and more general than $A \xrightarrow{3,3/4} TW$, we finally get $\mathcal{R}^G = \{TW \xrightarrow{3,1.0} A, A \xrightarrow{3,3/4} T\}$ as the final answer.

4.4. Complexity of rule generation: Traditional vs. new framework

The complexity of rule generation in the traditional framework is $O(f \cdot 2^l)$, exponential in the length l of the longest frequent itemset (f is the total number of frequent itemsets). On the other hand using the closed itemset framework, the number of non-redundant rules is linear in the number of closed itemsets. To see how much savings are possible using closed frequent itemsets, lets consider the case where the longest frequent itemset has length l; with all 2^l subsets also being frequent.

In the traditional association rule framework, we would have to consider for each frequent itemset all its subsets as rule antecedents. The total number of rules generated in this approach is given as $\sum_{i=0}^{l} \binom{l}{i} \cdot 2^{l-i} \leq \sum_{i=0}^{l} \binom{l}{i} \cdot 2^l = 2^l \sum_{i=0}^{l} \binom{l}{i} = 2^l \cdot 2^l = 0(2^{2l})$.

On the other hand the number of non-redundant rules produced using closed itemsets is given as follows. Let's consider two extreme cases: In the best case, there is only one closed itemset of length l, i.e., all 2^l subsets have the same support as the longest frequent itemset. Thus all rules between itemsets must have 100% confidence. The closed itemset approach produces the most general rules for the single itemset. These simple rules would be the rules between all single items. There are $l \cdot (l-1) = l^2 - l$ possible rules, all with 100% confidence. This corresponds to a reduction in the number of rules by a factor of $O(2^{2l}/l^2)$.

In the worst case, all 2^l frequent itemsets are also closed. In this case there can be no 100% confidence rules and all (<100% confidence) rules point upward, i.e., from subsets to their immediate supersets. For each subset of length k we have k rules from each of its $k-1$ length subsets to that set. The total number of rules generated is thus $\sum_{i=0}^{l} \binom{l}{i} \cdot (l-i) \leq \sum_{i=0}^{l} \binom{l}{i} \cdot l = O(l \cdot 2^l)$. Thus we get a reduction in the number of rules by of a factor of $O(2^l/l)$, i.e., asymptotically exponential in the length of the longest frequent itemset.

5. Experimental evaluation

All experiments described below were performed on a 400 MHz Pentium PC with 256 MB of memory, running RedHat Linux 6.0. Algorithms were coded in C++. Table 1 shows the characteristics of the real and synthetic datasets used in our evaluation. The real datasets were obtained from IBM Almaden (www.almaden.ibm.com/cs/quest/demos.html). All datasets except the PUMS (pumsb and pumsb*) sets, are taken from the UC Irvine Machine Learning Database Repository. The PUMS datasets contain census data. pumsb* is the same as pumsb without items with 80% or more support. The mushroom database contains characteristics of various species of mushrooms. Finally the connect and chess datasets are derived from their respective game steps. Typically, these real datasets are very dense, i.e., they produce many long frequent itemsets even for very high values of support.

Table 1. Database characteristics.

Database	No. of items	Record length	No. of records
Chess	76	37	3,196
Connect	130	43	67,557
Mushroom	120	23	8,124
pumsb*	7117	50	49,046
pumsb	7117	74	49,046
T20I12D100K	1000	20	100,000
T40I8D100K	1000	40	100,000
T10I4D100K	1000	10	100,000
T20I4D100K	1000	20	100,000

We also chose a few synthetic datasets (also available from IBM Almaden), which have been used as benchmarks for testing previous association mining algorithms. These datasets mimic the transactions in a retailing environment. Usually the synthetic datasets are sparse when compared to the real sets. We used two dense and two sparse (the last two rows in Table 1) synthetic datasets for our study.

5.1. Traditional vs. closed framework

Consider Tables 2 and 3, which compare the traditional rule generation framework with the closed itemset approach proposed in this paper. The tables shows the experimental results along a number of dimensions: (1) total number of frequent itemsets vs. closed frequent itemsets, (2) total number of rules in the traditional vs. new approach, and (3) total time taken for mining all frequent itemsets (using Apriori) and the closed frequent itemsets (using CHARM).

Table 2 shows that the number of closed frequent itemsets can be much smaller than the set of all frequent itemsets. For the support values we look at here, we got reductions (shown in the Ratio column) in the cardinality upto a factor of 45. For lower support values the gap widens rapidly (Zaki and Hsiao, 2002). It is noteworthy, that CHARM finds these closed sets in a fraction of the time it takes Apriori to mine all frequent itemsets as shown in Table 2. The reduction in running time ranges upto a factor of 145 (again the gap widens with lower support). For the sparse sets, and for high support values, the closed and all frequent set coincide, but CHARM still runs faster than Apriori. Table 3 shows that the reduction in the number of rules (with all possible consequent lengths) generated is drastic, ranging from a factor of 2 to more than 3000 times!

We also computed how many single consequent rules are generated by the traditional approach. We then compared these with the non-redundant rule set from our approach (with possibly multiple consequents). The table also shows that even if we restrict the traditional rule generation to a single item consequent, the reduction with the closed itemset approach is still substantial, with upto a factor of 66 reduction (once again, the reduction is more

Table 2. Number of itemsets and running time (Sup = *minsup*, Len = longest frequent itemset).

			Number of itemsets			Running time		
Database	Sup (%)	Len	#Freq	#Closed	Ratio	Apriori	CHARM	Ratio
Chess	80	10	8227	5083	1.6	18.54	1.92	9.7
Chess	70	13	48969	23991	2.0	213.03	8.17	26.1
Connect	97	6	487	284	1.7	19.7	4.15	4.7
Connect	90	12	27127	3486	7.8	2084.3	43.8	47.6
Mushroom	40	7	565	140	4.0	1.56	0.28	5.6
Mushroom	20	15	53583	1197	44.7	167.5	1.2	144.4
pumsb*	60	7	167	68	2.5	11.4	1.0	11.1
pumsb*	40	13	27354	2610	10.5	847.9	17.1	49.6
pumsb	95	5	172	110	1.6	19.7	1.7	11.7
pumsb	85	10	20533	8513	2.4	1379.8	76.1	18.1
T20I12D100K	0.5	9	2890	2067	1.4	6.3	5.1	1.2
T40I8D100K	1.5	13	12088	4218	2.9	41.6	15.8	2.6
T10I4D100K	0.5	5	1073	1073	1	2.0	1.1	1.8
T10I4D100K	0.1	10	27532	26806	1.03	32.9	8.3	4.0
T20I4D100K	1.0	6	1327	1327	1	6.7	4.8	2.6
T20I4D100K	0.25	10	30635	30470	1.01	32.8	10.7	3.1

Table 3. Number of rules (all vs. consequent of length 1) (Sup = *minsup*, Len = longest itemset).

			All possible rules			Rules with one consequent	
Database	Sup (%)	Len	#Traditional	#Closed	Ratio	#Traditional	Ratio
Chess	80	10	552564	27711	20	44637	2
Chess	70	13	8171198	152074	54	318248	2
Connect	97	6	8092	1116	7	1846	1.7
Connect	90	12	3640704	18848	193	170067	9
Mushroom	40	7	7020	475	15	1906	4.0
Mushroom	20	15	19191656	5741	3343	380999	66
pumsb*	60	7	2358	192	12	556	3
pumsb*	40	13	5659536	13479	420	179638	13
pumsb	95	5	1170	267	4	473	2
pumsb	85	10	1408950	44483	32	113089	3
T20I12D100K	0.5	9	40356	2642	15	6681	3
T40I8D100K	1.5	13	1609678	11379	142	63622	6
T10I4D100K	0.5	5	2216	1231	1.8	1231	1.0
T10I4D100K	0.1	10	431838	86902	5.0	90350	1.04
T20I4D100K	1.0	6	2736	1738	1.6	1738	1.0
T20I4D100K	0.25	10	391512	89963	4.4	90911	1.01

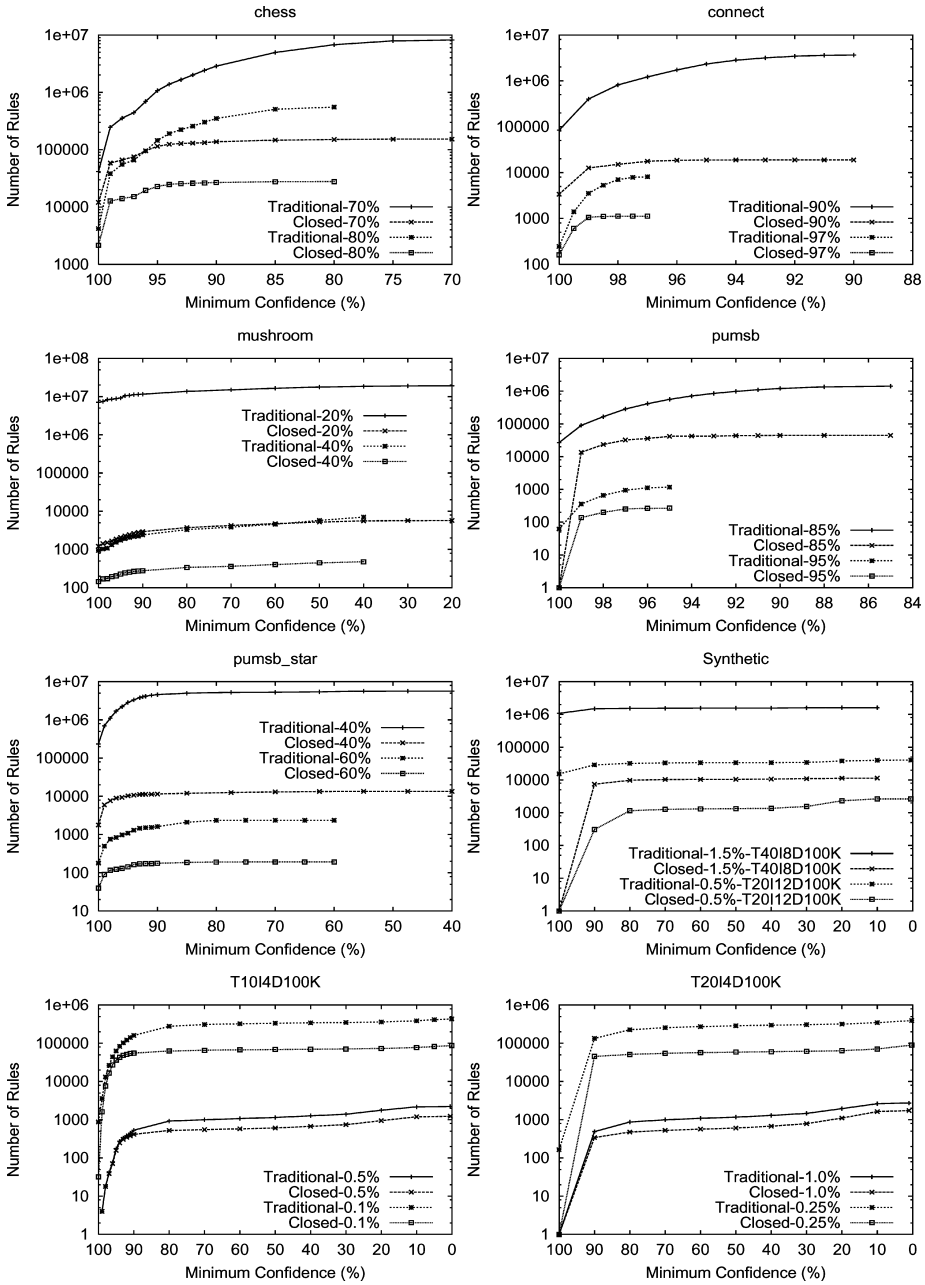

Figure 12. Number of rules: Traditional vs. closed itemset framework.

for lower supports). It is worth noting that, even though for sparse synthetic sets the closed frequent itemsets is not much smaller than the set of all frequent itemsets, we still get upto a factor of 5 reduction in the number of rules generated.

The results above present all possible rules that are obtained by setting *minconf* equal to the *minsup*. Figure 12 shows the effect of *minconf* on the number of rules generated. It shows that most of the rules have very high confidence; as the knee of the curves show, the vast majority of the rules have confidences between 95 and 100 percent! This is a particularly distressing result for the traditional rule generation framework. The new approach produces a rule set that can be orders of magnitude smaller. In general it is possible to mine closed sets using CHARM for low values of support, where it is infeasible to find all frequent itemsets. Thus, even for dense datasets we can generate rules, which may not be possible in the traditional approach.

6. Conclusions

This paper has demonstrated in a formal way, supported with experiments on several datasets, the well known fact that the traditional association rule framework produces too many rules, most of which are redundant. We proposed a new framework based on closed itemsets that can drastically reduce the rule set, and that can be presented to the user in a succinct manner.

This paper opens a lot of interesting directions for future work. For example we plan to use the concept lattice for interactive visualization and exploration of a large set of mined associations. Keep in mind that the frequent concept lattice is a very concise representation of all the frequent itemsets and the rules that can be generated from them. Instead of generating all possible rules, we plan to generate the rules on-demand, based on the user's interests. Finally, there is the issue of developing a theory for extracting a base, or a minimal generating set, for all the rules.

Notes

1. Only meet is defined on frequent sets, while the join may not exist. For example, $AC \wedge AT = AC \cap AT = A$ is frequent. But, while $AC \vee AT = AC \cup AT = ACT$ is frequent, $AC \cup DW = ACDW$ is not frequent.
2. The term *formal concept* was introduced by Wille in the 80's (Ganter and Wille, 1999) to formalize the notion of a concept. The set of tids denotes the extent of the concept, i.e., the objects that share some attributes, while the set of items denotes the intent of the concept, i.e., the attributes or properties shared by the objects.
3. One possible objection that can be raised to the closed itemset framework is that a small change in the data can change the number of closed itemsets. However, the frequency requirement makes the framework robust to small changes, i.e., while the set of closed itemsets can still change, the set of frequent closed itemsets is resilient to change.

References

Agrawal, R., Mannila, H., Srikant, R., Toivonen, H., and Inkeri Verkamo, A. 1996. Fast discovery of association rules. In Advances in Knowledge Discovery and Data Mining, U. Fayyad et al. (Eds.), Menlo Park, CA: AAAI Press, pp. 307–328.

Bastide, Y., Pasquier, N., Taouil, R., Stumme, G., and Lakhal, L. 2000a. Mining minimal non-redundant association rules using frequent closed itemsets. In 1st International Conference on Computational Logic.

Bastide, Y., Taouil, R., Pasquier, N., Stumme, G., and Lakhal, L. 2000b. Mining frequent patterns with counting inference. SIGKDD Explorations, 2(2).

Bayardo, R.J. 1998. Efficiently mining long patterns from databases. In ACM SIGMOD Conf. Management of Data.

Bayardo, R.J. and Agrawal, R. 1999. Mining the most interesting rules. In 5th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining.

Brin, S., Motwani, R., Ullman, J., and Tsur, S. 1997. Dynamic itemset counting and implication rules for market basket data. In ACM SIGMOD Conf. Management of Data.

Burdick, D., Calimlim, M., and Gehrke, J. 2001. MAFIA: A maximal frequent itemset algorithm for transactional databases. In Intl. Conf. on Data Engineering.

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

Ganter, B. and Wille, R. 1999. Formal Concept Analysis: Mathematical Foundations. Springer-Verlag.

Godin, R., Missaoui, R., and Alaoui, H. 1995. Incremental concept formation algorithms based on galois (concept) lattices. Computational Intelligence, 11(2):246–267.

Guigues, J.L. and Duquenne, V. 1986. Familles minimales d'implications informatives resultant d'un tableau de donnees binaires. Math. Sci. hum., 24(95):5–18.

Klemettinen, M., Mannila, H., Ronkainen, P., Toivonen, H., and Verkamo, A.I. 1994. Finding interesting rules from large sets of discovered association rules. In 3rd Intl. Conf. Information and Knowledge Management, pp. 401–407.

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

Liu, B., Hsu, W., and Ma, Y. 1999. Pruning and summarizing the discovered associations. In 5th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining.

Luxenburger, M. 1991. Implications partielles dans un contexte. Math. Inf. Sci. hum., 29(113):35–55.

Ng, R.T., Lakshmanan, L., Han, J., and Pang, A. 1998. Exploratory mining and pruning optimizations of constrained association rules. In ACM SIGMOD Intl. Conf. Management of Data.

Pasquier, N., Bastide, Y., Taouil, R., and Lakhal, L. 1999a. Discovering frequent closed itemsets for association rules. In 7th Intl. Conf. on Database Theory.

Pasquier, N., Bastide, Y., Taouil, R., and Lakhal, L. 1999b. Efficient mining of association rules using closed itemset lattices. Information Systems, 24(1):25–46.

Pei, J., Han, J., and Mao, R. 2000. Closet: An efficient algorithm for mining frequent closed itemsets. In SIGMOD Int'l Workshop on Data Mining and Knowledge Discovery.

Savasere, A., Omiecinski, E., and Navathe, S. 1995. An efficient algorithm for mining association rules in large databases. In 21st VLDB Conf.

Taouil, R., Bastide, Y., Pasquier, N., and Lakhal, L. 2000. Mining bases for association rules using closed sets. In 16th IEEE Intl. Conf. on Data Engineering.

Toivonen, H., Klemettinen, M., Ronkainen, P., Hätönen, K., and Mannila, H. 1995. Pruning and grouping discovered association rules. In MLnet Wkshp. on Statistics, Machine Learning, and Discovery in Databases.

Zaki, M.J. and Hsiao, C.-J. 2002. CHARM: An efficient algorithm for closed itemset mining. In 2nd SIAM International Conference on Data Mining.

Zaki, M.J. and Ogihara, M. 1998. Theoretical foundations of association rules. In 3rd ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery.

Zaki, M.J., Parthasarathy, S., Ogihara, M., and Li, W. 1997. New algorithms for fast discovery of association rules. In 3rd Intl. Conf. on Knowledge Discovery and Data Mining.

Zaki, M.J. and Phoophakdee, B. 2003. MIRAGE: A framework for mining, exploring and visualizing minimal association rules. Technical Report 03-4, Computer Science Dept., Rensselaer Polytechnic Institute.

Mohammed J. Zaki is an associate professor of Computer Science at RPI. He received his Ph.D. degree in computer science from the University of Rochester in 1998. His research interests include the design of efficient,

scalable, and parallel algorithms for various data mining techniques and he is specially interested in developing novel data mining techniques for bioinformatics.

Dr. Zaki has published over 100 papers on data mining, co-edited 10 books/proceedings, and served as guest-editor for Information Systems, SIGKDD Explorations, and Distributed and Parallel Databases: An International Journal. He is the founding co-chair for the ACM SIGKDD Workshop on Data Mining in Bioinformatics (2001–2003), and has co-chaired several workshops on High Performance Data Mining. He is currently an associate editor for IEEE Transactions on Knowledge and Data Engineering and an editor for International Journal of Data Warehousing and Mining. He received the US National Science Foundation's CAREER Award in 2001 and the US Department of Energy's Early Career Principal Investigator Award in 2002. He also received ACM Recognition of Service Award in 2003.