MIRAGE: A Framework for Mining, Exploring and Visualizing Minimal Association Rules *

Mohammed J. Zaki and Benjarath Phoophakdee
Computer Science Department
Rensselaer Polytechnic Institute, Troy NY 12180
{zaki,phoopb}@cs.rpi.edu

Abstract

In this paper we propose the concept of minimal association rules, the most general rules that satisfy a given support and confidence threshold. We present MIRAGE, an new framework for mining and visually exploring the minimal rules. MIRAGE uses lattice-based interactive rule visualization approach, displaying the rules in a very compact form; all association rules can also be generated if desired. MIRAGE uses a database back-end to store the state of exploration for easy retrieval at a later point in time.

1 Introduction

An association rule is an implication of the form $X \longrightarrow Y$, where X is a set of antecedent items and Y is the consequent items. Each rule is presented along with its support and confidence, which are the joint and conditional probabilities of the antecedent and consequent items, respectively.

Association rules are traditionally presented as sets of the above implications, which are often difficult and unintuitive to analyze. There are two main problems. First, there are typically too many rules that are mined making human exploration almost impossible. Second, visualization support for multiple item association rules has proved to be hard. These limitations present serious challenges for analysts who need to understand the mined patterns.

This paper presents a novel framework for the mining and exploration of association rules, based on the concept of *minimal association rules*. Minimal association rules are the most general rules (i.e., having most general antecedent and consequent) that satisfy a given support and confidence threshold. Minimal rules are typically a lot less than the the full set of association rules, and this helps address the combinatorial rule explosion problem.

In this paper we propose MIRAGE [1], a system for mining and visually exploring minimal association rules. MIRAGE uses lattice-based interactive rule visualization approach, displaying the rules in a very compact form; all association rules can also be generated if desired. Hence, there is no information loss. MIRAGE uses a database back-end for effective and persistent rule management for easy retrieval at a later point in time.

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

[1] MIRAGE is an anagram of the letters IGEMAR, taken from: **I**nteractive **G**raphical **E**xploration of **M**inimal **A**ssociation **R**ules

2 Related Work

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 are two bodies of relevant work. One deals with the theoretical frameworks for rule reduction, including mining only interesting rules. The other deals with rule visualization. We review the work in both these areas.

This paper builds upon our previous work on non-redundant association rule mining [16]. The key differences are as follows. Minimal rules are a refinement of non-redundant association rules proposed earlier. While the previous work laid the theoretical foundation, this paper presents concrete algorithms for rule generation and exploration. We also formally prove that the set of all minimal rules is a *representative set*, that is, for any (redundant) association rule, there exists an equivalent minimal rule. Finally, we propose MIRAGE, an integrated tool based on the minimal rule framework, for interactive visual rule exploration.

Most of the foundational work on reducing the association rules is based on the concept of closed itemsets, which utilizes the elegant mathematical framework of formal concept analysis [4] (FCA). This includes the early work of Luxenburger [11]. He proposed an approach for obtaining a generating set of rules, but he did not consider frequent rules, and no algorithms were proposed. Like our earlier work in [18], Stumme et al [14] proposed a basis for association rules based on the work of Guigues and Duquenne [5] and Luxenburger [11]. Rule visualization was not addressed in any of these works.

The work by Taouil et al [2] has also addressed the problem of extracting "minimal" association rules. However, their definition of minimal rules is different from ours; they consider a rule to be minimal if it has the most general antecedent and the most specific consequent. This different definition of minimality leads to different, mutually complementary, formulations of the representative sets for association rules. Also they do not discuss any system for visual exploration of rules, as is done in our approach. Another novel aspect of our work is the persistent rule management support, using a database backend.

There has been some work on mining interesting association rules [9, 3, 12]. 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 and/or that of visualization.

There have been very few papers in the past that have presented visualization techniques for association rules [15, 6, 7, 10]. Wong et al [15] present an association-rule visualization system that can deal with large databases of text. Their system presents to the user all association rules that can be generated at once in the form of a matrix between rules and individual items; they use bar graphs to present the support and confidence of each rule. The work of Hofmann et al [7] focuses mainly on visual representation of association rules and presenting the relationships between related rules using interactive Mosaic plots (histograms) and Double Decker plots. In both these approaches, there is no focus on rule reduction and user interaction is limited.

Liu et al [10] present an integrated system for finding interesting associations and visualizing them. We believe that our lattice based system MIRAGE is more intuitive. Further, the two approaches are complementary, since we present minimal rules. It would be an interesting future direction to combine interestingness with minimal rules.

Han and Cercone [6] propose an interactive model for visualizing the entire process of knowledge discovery and data mining. They mainly focus on the "parallel coordinates" technique to visualize rule induction algorithms. Other methods for general purpose visual exploration and mining are presented by Keim [8]. Our approach is specifically designed for visualizing association rules.

3 Preliminaries

The association rule mining framework can be described as follows: Let $I = \{1, 2, \ldots, m\}$ be a set of items, and $T = \{1, 2, \ldots, n\}$ be a set of transaction identifiers. The input dataset is a binary relation $\delta \subseteq I \times T$. For example, consider the input database as shown in Table 1, which will be used as a running example throughout this paper. Here $I = \{A, C, D, T, W\}$, $T = \{1, 2, 3, 4, 5, 6\}$ and the first transaction can be written as $\{A\delta 1, C\delta 1, T\delta 1, W\delta 1\}$. A set $X \subseteq I$ is called an *itemset* and a set $Y \subseteq T$ 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 common to all the tids in Y. The composition of the two functions, namely, t that maps from itemsets to tidsets, and i that maps from tidsets to itemsets, is called a *closure operator* [4], and is given as $c_{it}(X) = i(t(X))$. It can be shown that an itemset X is closed if and only if $c_{it}(X) = X$ [16]. For instance, AW is not closed since $c_{it}(AW) = i(t(AW)) = i(1345) = ACW$. On the other hand ACW is closed since $c_{it}(ACW) = i(t(ACW)) = i(1345) = ACW$.

Sample Dataset

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

Closed Sets

Itemset	Tidset	Sup
C	123456	6
CD	2456	4
CT	1356	4
CW	12345	5
CWA	1345	4
CDW	245	3
CTWA	135	3

Figure 1: Sample Dataset and Closed Sets

The *support* [1] of an itemset X, denoted $\pi(X)$, is the number of transactions in which it occurs as a subset, i.e., $\pi(X) = \mid t(X) \mid$. An itemset is *frequent* if its support is more than or equal to a user-specified *minimum support* ($\pi^{\min}$) value, i.e., if $\pi(X) \geq \pi^{\min}$. A frequent itemset is called *maximal* if it is not a subset of any other frequent itemset. A frequent itemset X is called *closed* if there exists no proper superset $Y \supset X$ with $\pi(X) = \pi(Y)$. Figure 1 shows all the closed itemsets along with their tidsets and supports on the example dataset, using $\pi^{\min} = 3$. These closed sets can be arranged in a lattice, as shown in Figure 2 (which, incidentally, is a screenshot of MIRAGE). The right side shows all the 7 closed sets arranged according to the subset relation. There is an edge connecting two closed sets X and Y, if and only if $X \subset Y$ and $\nexists Z$, such that, $X \subset Z \subset Y$. The levels of the lattice correspond to the length of the itemsets. Thus the smaller sets (with higher support) appear on top and the larger itemsets (with lower support) appear below. The support of a set X is the same as the support of the smallest closed set that contains it [2, 16]. For example, the smallest closed set containing A is ACW, thus $\pi(A) = \pi(ACW) = 3$.

Association Rules Let A and B be any itemsets. An *association rule* is an implication $A \xrightarrow{q,p} B$, where $A \cap B = \emptyset$.

The *support* of the rule is $q = \pi(A \cup B) = \mid t(A \cup B) \mid$, and the *confidence* $p = \frac{\pi(A \cup B)}{\pi(A)} = \frac{|t(A \cup B)|}{|t(A)|}$ (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., $q \geq \pi^{\min}$). A rule is *confident* if $p \geq minconf$, where

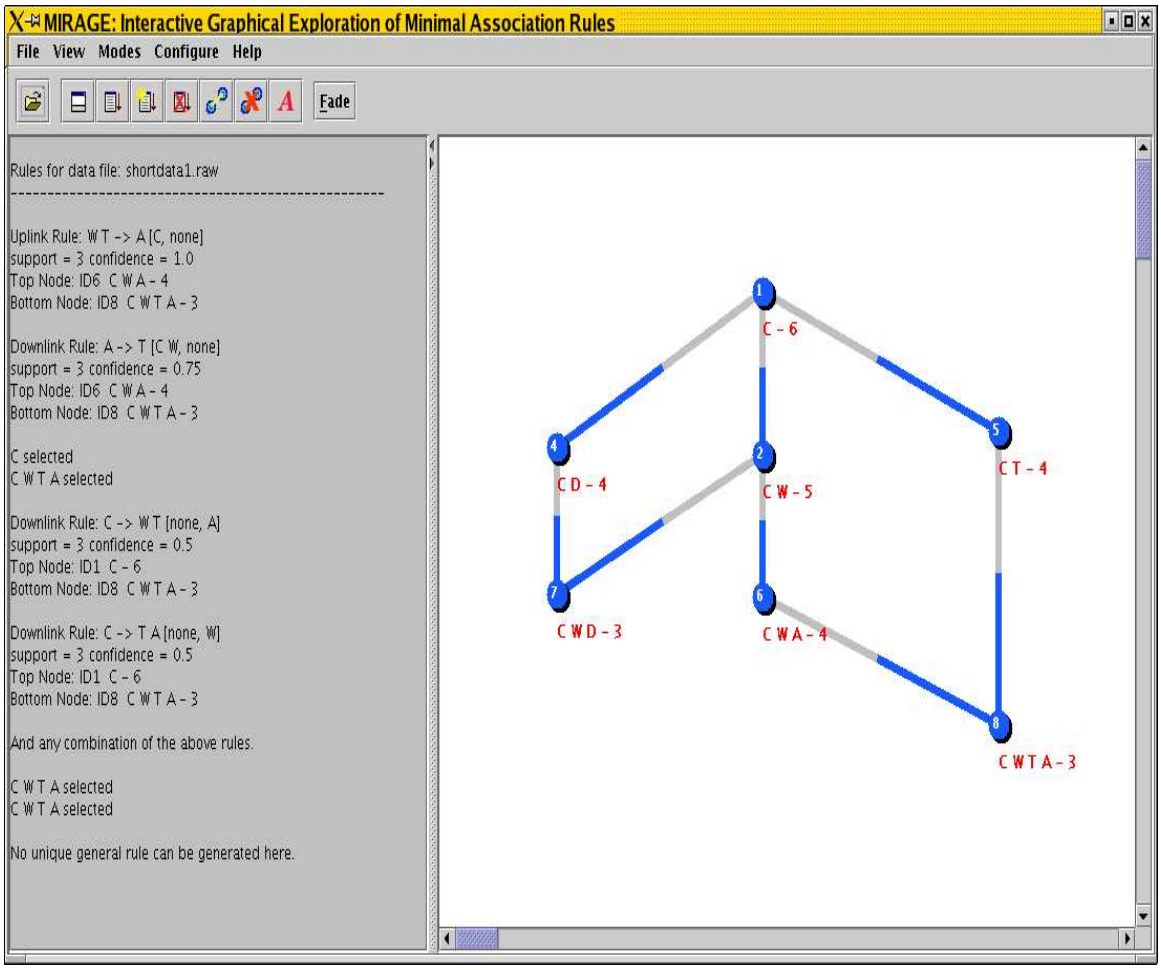

Figure 2: MIRAGE: Lattice-based Itemset and Rules Visualization

where *minconf* is a user-specified minimum threshold. When support is understood, we omit q and write a rule as $A \xrightarrow{p} B$. Please note that the confidence p is the rational number $p = \frac{q}{\pi(A)}$, and we explicitly write it as such when required for emphasis. A rule with $p = q/q = 1.0$ is called an *exact* association rule, and a rule with $q < \pi(A)$, and thus $p < 1.0$ is called an *inexact* association rule.

Once the frequent sets have been mined, association rule generation works as follows [1]: rules of the form $X' \xrightarrow{q,p} X - X'$, are generated for all frequent itemsets X (where $X' \subset X$, and $X' \neq \emptyset$), provided $p \geq minconf$. For example, from the frequent itemset ACW we can generate 6 possible rules (all of them have support of $q = 4$): $A \xrightarrow{4/4} CW, C \xrightarrow{4/6} AW, W \xrightarrow{4/5} AC, AC \xrightarrow{4/4} W, AW \xrightarrow{4/4} C$, and $CW \xrightarrow{4/5} A$. To obtain all possible rules we need to examine each frequent itemset and repeat the rule generation process as shown for ACW.

Minimal Association Rules Let R_i denote the association rule $X_1^i \xrightarrow{q_i,p_i} X_2^i$, and let $\mathcal{R} = \{R_1, \cdots, R_n\}$ be a set of rules. We say that rule $R_i \in \mathcal{R}$ is more *general* than a rule $R_j \in \mathcal{R}$, denoted $R_i \preceq R_j$, if and only if the following conditions are met: 1) $X_1^i \subseteq X_1^j$, 2) $X_2^i \subseteq X_2^j$, 3) $q_i = q_j$, and 4) $p_i = p_j$. In other words R_j has the same support and confidence as R_i, and it can be generated by adding items to either the antecedent or consequent of R_i. It is clear that R_j is

not conveying any new information then R_i. Thus we say that a rule R_j is *redundant* if there exists some rule R_i, such that $R_i \preceq R_j$. We denote by $\mathcal{R}^{\min} = \{R_i \in \mathcal{R} \mid \nexists R_j \in \mathcal{R}, R_j \prec R_i\}$ the set of minimal association rules in $\mathcal{R}$.

As an example consider the rule set $\mathcal{R} = \{R_1 : W \xrightarrow{4,4/5} A, R_2 : W \xrightarrow{4,4/5} AC, R_3 : CW \xrightarrow{4,4/5} A, R_4 : C \xrightarrow{5,5/6} W, R_5 : A \xrightarrow{4,4/4} W, R_6 : A \xrightarrow{4,4/4} CW, R_7 : AC \xrightarrow{4,4/4} W\}$. Then the set of minimal rules is given as $\mathcal{R}^{\min} = \{R_1 : W \xrightarrow{4,4/5} A, R_4 : C \xrightarrow{5,5/6}, R_5 : A \xrightarrow{4,4/4} W\}$, since R_2 and R_3 are redundant with respect to (w.r.t) R_1, and R_6 and R_7 are redundant w.r.t R_5.

4 MIRAGE: Interactive Graphical Exploration of Minimal Association Rules

MIRAGE allows interactive graphical visualization and exploration of minimal association rules. Figure 3 presents a conceptual view of MIRAGE. We assume that the database of interest has already been mined, using any of the efficient closed itemset mining algorithms, such as Charm [17] or Closet [13]. MIRAGE takes either all frequent itemsets or the closed frequent itemsets along with their supports and optional extensions (i.e., tidset) as input. As an example, Figure 1 illustrates the set of closed itemsets that would be input to MIRAGE. If all frequent itemsets are input MIRAGE first infers the closed sets among them, and uses only closed sets for further processing. We shall see later that MIRAGE also accepts a previously stored itemset lattice as input, as well as an entire previous state of exploration (itemsets, lattice and rules) using its database support.

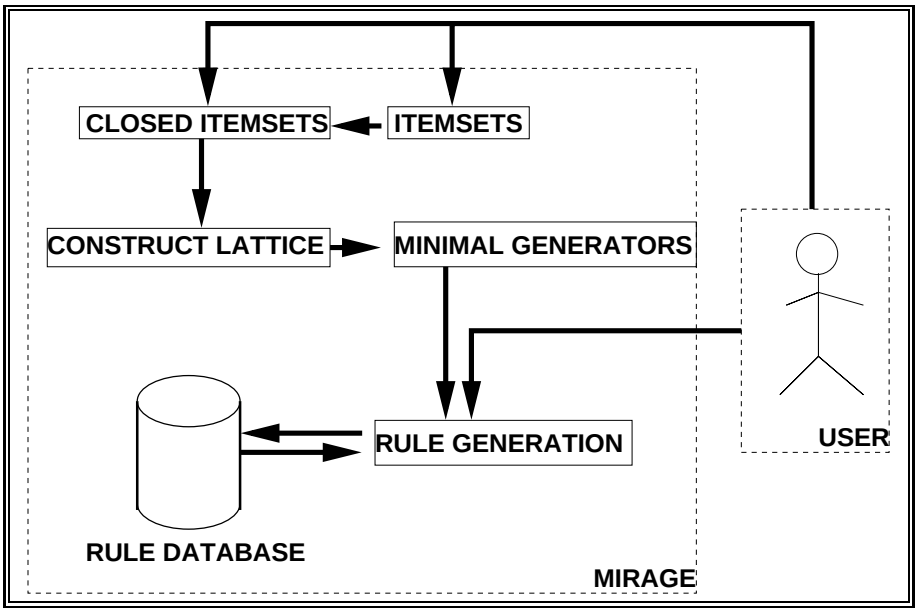

Figure 3: Conceptual Architecture of MIRAGE

From the closed itemsets, MIRAGE creates a lattice, as shown in the screenshot in Figure 2. Each closed itemset is represented as a node, with a unique identifier, in the lattice. There is an edge connecting two nodes if they are related by the subset relation and there is no intermediate set between them. The size of a node is the length of the itemset, which also governs the levels of the lattice. The smallest nodes are represented as roots, which are the only nodes displayed initially. The use can then selectively expand nodes of interest to display the relationships. The

Action#	Mouse Actions	Functions Performed
A1	Left double click	Collapse or expand a node
A2	Right single click and drag on a node	Move the node to another location
A3	Right single click on a canvas	Center the canvas view at the location clicked
A4	Left single click on the gray portion of an edge	Display uplink rules between nodes on this edge
A5	Left single click on the blue portion of an edge	Display downlink rules between nodes on this edge
A6	Ctrl-left click on a node	Node Selection, for displaying rules between any two nodes

Table 1: Some Mouse Commands in MIRAGE

figure shows the full lattice where the user has expanded all nodes. Also shown with each node is its support, e.g., CWA–4.

Once the lattice has been created, MIRAGE computes the minimal generators (defined later in Section 5.2) of all closed sets. It then waits for user interactions to generate and display minimal rules. Minimal rule generation consists of two steps: 1) finding minimal generators of each closed itemset, and 2) generating minimal rules among pairs of minimal generators. Details will be presented later.

MIRAGE maintains a rule database, using a DBMS, to store all mined patterns and rules in an interactive manner. Thus, it provides persistency by allowing the user to store the current state of exploration (the lattice, rules, etc.) and to return to it at a later point in time.

4.1 Interactive Itemset and Rule Exploration

MIRAGE has two main display areas: the left text area for displaying details of minimal association rules generated, and the right canvas for displaying the interactive itemset lattice, as shown in the screenshot in Figure 2. On the top there are some menu buttons and icons for some common rule modes.

4.1.1 Interactive Itemset Exploration

For itemset display there are four choices (the `view` menu button): i) itemset, ii) itemset–support, iii) itemset–tidset, and iv) itemset–support–tidset. The itemsets shown in Figure 2 are in the itemset–support view. The last two modes are typically not useful for market basket datasets, where the individual transaction id is not of interest. However, the tidset mode is very useful in applications like gene expression mining [19], where the tidset represents the experiments that exhibit a similar expression for a given set of genes (the itemset).

MIRAGE supports interactive exploration of association rules between itemsets by providing a variety of mouse commands as shown in Table 1. Left double clicking (A1) either expands a node, or collapses it if already expanded. This allows the user to selectively explore parts of the lattice of interest starting from some root node. A1 collapses the entire lattice under an already expanded node. For example, Figure 4 (left) shows a sequence of node expansions from node C to CW, and then CWA. When we apply A1 on $CWTA$ it becomes black indicating no further expansion is possible. Figure 4 (right) shows what happens when we collapse node CW; the entire lattice under it is removed. Command A2 allows the user to move the node positions to avoid clutter, and A3 allows the user to focus/center the entire lattice on the clicked location.

Another feature for interactive itemset exploration in MIRAGE is the No-Fade/Fade mode, as illustrated in Figure 5 (one can toggle between the two modes by clicking on the Fade icon button on top). In no-fade mode, multiple nodes can be expanded per level of the lattice as space permits. If there is limited space for an expansion, a red star appears on top of the node, and the node is

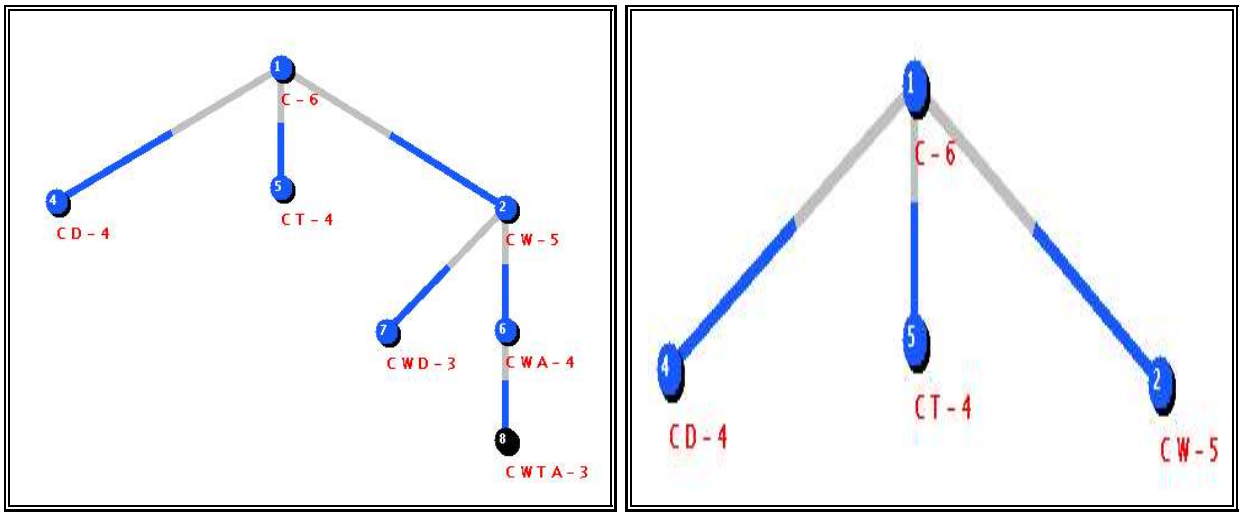

Figure 4: Expanding and Collapsing

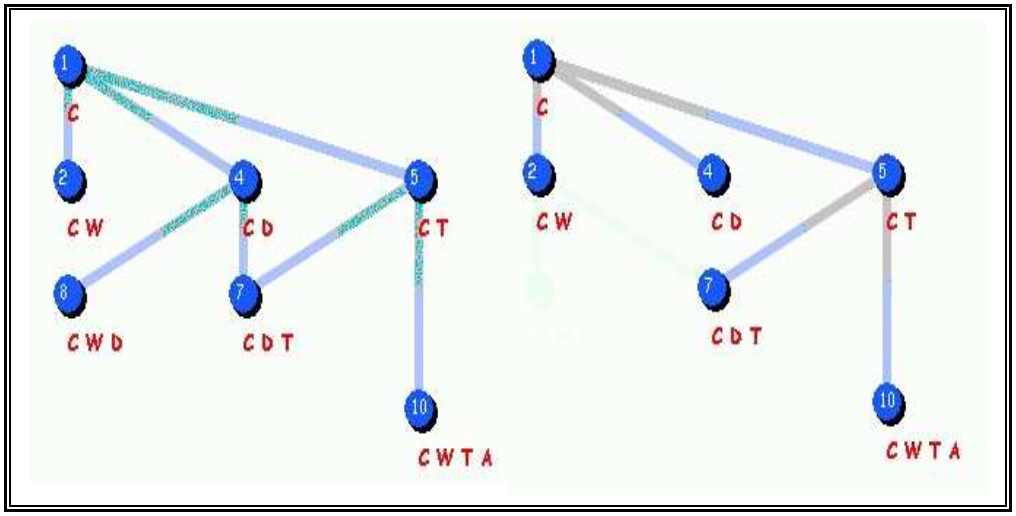

Figure 5: No Fade (left) and Fade (right) Modes

not expanded. Once the user widens the canvas or removes extraneous itemsets, the node can be expanded. In fade mode, we display only the children of the current node, and all other edges from the current node's siblings are faded away. This helps avoid the clutter, and allows the focus to remain on the most recent node and its children. For example, in Figure 5, after expanding C, when we expand CD and CT, both their children are visible in the no-fade node. On the other hand, when we click on CT, in the fade mode, the children of CD are faded out.

4.1.2 Interactive Rule Exploration

It has been shown previously [2, 16] that it suffices to consider association rules among the frequent closed itemsets, since a rule between any two itemsets is exactly the same as the rule between their respective closures. We shall prove later that it is in fact sufficient to consider rules among only the adjacent closed itemsets. MIRAGE allows the user to generate minimal rules among any two nodes, whether adjacent or non-adjacent.

In rule exploration we need to consider two kinds of minimal rules: the exact rules with confidence $p = 1.0$, and inexact rules with confidence $p < 1.0$. There are two cases that lead to exact rules:

- *Self rule*: Directed from a frequent closed itemset to itself.
- *Uplink rule*: Directed from a frequent closed itemset to an adjacent closed subset, i.e., to one of its parents in the lattice.

On the other hand, there is only one case that produces an inexact rule:

- *Downlink rules*: Directed from a frequent closed itemset to an adjacent closed superset, i.e., to one of its children in the lattice.

Commands A4, A5 and A6 in Table 1 allow the user to explore exact and inexact minimal rules between adjacent or non-adjacent nodes. Consider the lattice shown in Figure 2. Each edge in the lattice is broken into two parts. Clicking on the top gray half (A4) generates an uplink exact rule between the nodes, while clicking on the bottom blue half (A5) generates a downlink inexact rule, if such a rule exists. The rule is displayed in the left text area. Each minimal rule is presented in a manner that conveys as much information as possible. A rule is displayed in the text area as follows:

`Uplink/Downlink Rule:` $L \longrightarrow R\ [I^{\top}, I^{\perp}]$
`support =` q`, confidence =` p
`Top Node(`$X^{\top}$`):  NodeID -` $\pi(X)$
`Bottom Node (`$X^{\perp}$`):  NodeID -` $\pi(Y)$

The display indicates whether the rule is uplink or downlink between the Top Node ($\top$) and Bottom Node ($\perp$); their NodeIDs, itemsets ($X^{\top}$ for top node and $X^{\perp}$ for bottom node), and supports are also displayed under the node. Note that the top node by definition is a subset of the bottom node, i.e. $X^{\top} \subset X^{\perp}$. The rule support and confidence are shown separately. The actual minimal rule is given as $L \longrightarrow R\ [I^{\top}, I^{\perp}]$, where L is the itemset on the left hand side (i.e.,antecedent) and R is the itemset on right hand side (i.e., consequent). $I^{\top}$ is the set of items that can be added to both L and R to yield an equivalent rule (i.e., having the same support and confidence). $I^{\perp}$ is the set of items that can be added *only* to L in case of an uplink rule, and *only* to R in case of a downlink rule. Items in $I^{\perp}$ can be added to either the antecedent or the consequent, provided it is derived from the bottom node, while items in $I^{\top}$ can be added to *both* antecedent and consequent. We ensure that $I^{\top} \cap I^{\perp} = \emptyset$.

Some example of rules generated by MIRAGE on clicking different edges/nodes are given in Figure 2. For instance the first two rules on the left show the uplink rule $WT \overset{3,3/3=1.0}{\longrightarrow} A$ produced by clicking on the grey half of the edge (command A4), and the downlink rule $A \overset{3,3/4=0.75}{\longrightarrow} T$, produced by clicking the blue half of the edge (command A5), between `TopNode` : CWA and `BottomNode` : $CWTA$. The downlink rule $A \overset{3,0.75}{\longrightarrow} T$ has $I^{\top} = \{C, W\}$ and $I^{\perp} = \emptyset$, which means that any subset of CW can be added to both the antecedent or the consequent to obtain an equivalent redundant association rule. In other word, the notation conveys that all of the following 8 rules $AW \longrightarrow T, AC \longrightarrow T, ACW \longrightarrow T, A \longrightarrow CT, A \longrightarrow TW, A \longrightarrow CTW, AW \longrightarrow CT, AC \longrightarrow TW$, are equivalent to the minimal rule $A \overset{3,0.75}{\longrightarrow} T$, Since $I^{\perp} = \emptyset$, no additional items can be added to L to yield an equivalent rule. For the uplink rule $WT \overset{3,3/3=1.0}{\longrightarrow} A$, $I^{\top} = C$, which means that the following two rule are redundant: $WT \longrightarrow AC$, and $CWT \longrightarrow A$.

It is also possible to generate rules between non-adjacent nodes. We first select C and then $CWTA$ using command A6; Figure 2 displays the two selected nodes in the left text area (`C selected`, and `CWTA selected`). The first node clicked is top node and the second the bottom node, which corresponds to a request to generate a downlink rule from C to $CWTA$. There are two minimal downlink rules generated. The first one is $C \overset{3,3/6=0.5}{\longrightarrow} WT\ [\emptyset, A]$, meaning that $C \overset{3,0.5}{\longrightarrow} WT$ is the minimal rule, and since $I^{\perp} = \{A\}$, it means that A can be added to the consequent to produce the redundant rule $C \overset{3,0.5}{\longrightarrow} AWT$. The other minimal rule is $C \overset{3,0.5}{\longrightarrow} TA$ with $I^{\perp} = \{W\}$.

As a final example, Figure 2 shows how to generate self-rules. This is done by selecting the same node twice using A6. The node selected is $CWTA$; in this case there are no self-rules possible. It can be seen from the examples above that MIRAGE displays minimal rules, yet it conveys maximum amount of information which allows the user to infer other redundant rules from $I^{\top}$ and $I^{\perp}$.

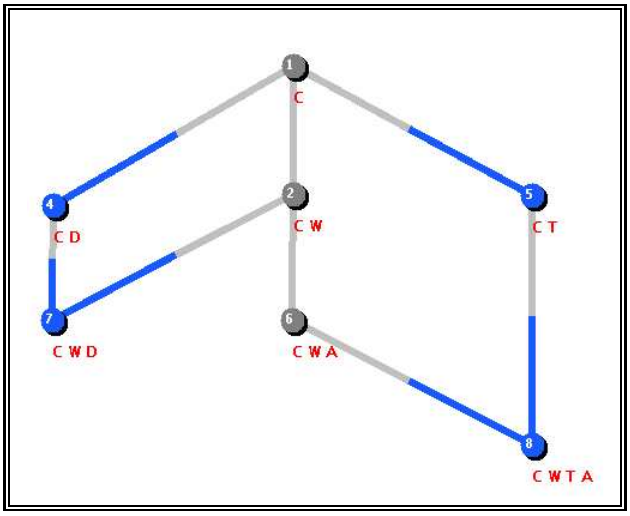

Figure 6: Constraints

4.1.3 Rule and Item Constraints

The system also allows the user to set restrictions on the confidence and support levels of the rules. For example, a user may only be interested in rules where confidence is > 0.8 and support is ≥ 10. Additionally, the system allows constraints on itemsets to be made. For example, a user may be interested in rules that involve strictly some items. For example Figure 6 shows the effect of specifying the constraint to display only those itemsets containing D or T; nodes not meeting the constraint are faded slightly (nodes C, CW and CWA). The system allows any number of items to be specified in the constraint. As part of future work we will allow greater power to specify different kinds of constraints.

4.1.4 Rule Management

There are several rule management options provided in MIRAGE, as shown in Figure 7 (which is a screen-shot of the drop-down menu obtained by clicking on the `Modes` menu button in Figure 2). There are two main rule exploration modes: normal and cached. In the normal (non-cached) mode, minimal rules between itemsets are generated dynamically as the user commands. The system has

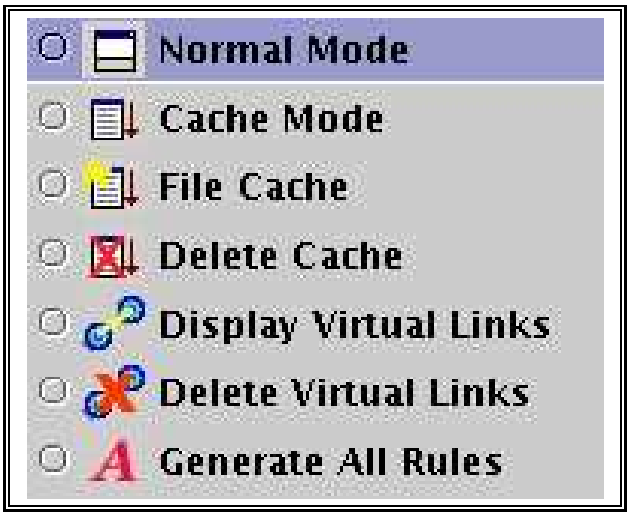

Figure 7: Modes

no knowledge of rules that were generated earlier, i.e., no exploration history is kept; each time the user click on an edge the rules are generated afresh.

In cache mode, all rules that have been generated are stored in a backend database using the open source MySQL DBMS (www.mysql.com). In cache mode, prior to any rule generation, MIRAGE queries the rule database whether it already has rules between the given nodes. If the rules are already in the cache (generated during the current exploration or during some previous exploration, in cache mode), they are retrieved from the database and displayed directly. This greatly speeds up rule exploration. If the rules are not found in the cache, MIRAGE generates the rules, displays them, and also stores them in the cache for possible future references.

There are several other rule management options available in MIRAGE, as shown in Figure 7. The `Delete Cache` button clears the entire state of exploration; all cached rules are deleted from the database. The `File Cache` button allows the user to export all current rules in the database to a flat file. The `Generate All Rules` button generates all the possible association rules, minimal as well as redundant, and stores them in the database (i.e., it can only be invoked in the cache mode). In conjunction with the file cache option this gives the user the ability to export all association rules to a file.

Finally, the user has an option of displaying and deleting *virtual links*, as illustrated in Figure 8. In the cached mode, when a user generates rules between any two nodes, and the `Display Virtual Links` mode is on, a virtual link (i.e., a directed edge) is displayed between those itemsets. The direction of the virtual edge is either up or down depending on whether the generated rules are uplink or downlink rules. Such links are called virtual since they may not correspond to a lattice edge, which exists only between adjacent closed itemsets. The virtual link mode allows the user to see on the lattice which rules have already been generated and cached in the database, as well as the direction. As shown in Figure 8 the user has generated both uplink and downlink rules between C and CT, and downlink rules between C and CD, but rules between C and CW have not been explored.

MIRAGE allows the user to store not only the rules generated via the cached mode, but it also allows the lattice to be persistently stored in the database. A user can later reload it directly from the database instead of having to regenerate the same lattice. If there are any cached rules, they will also be saved along with the lattice in the same database. Lattice and rule saving allows

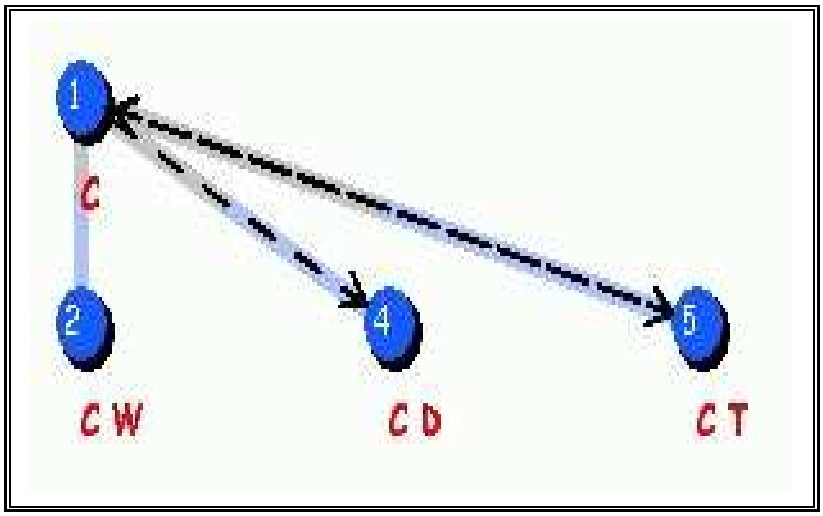

Figure 8: Virtual Links

a user to save tremendous time when working with a large database, since the time required for lattice reloading is a lot faster than the time required for regenerating it.

4.1.5 Database Design

The MySQL database consists of two tables: `Nodes`, and `Rules`. `Nodes` saves the lattice structure, while `Rules` caches the explored rules. The table definitions, and instantiation are shown in Figure 9.

The `Nodes` table stores for each node, its ID, itemset, tidset, support, confidence, and a list of its children and parents node ids in string form. For example, node ID 2 corresponds to CW, with support 5 and level 1. The tidset is empty for this dataset, but it may be useful for other applications. CW has two children, whose ids are 7 and 6, which we store as the string "7 6". It only has one parent: node "1". We adopted the string format for the list of children and parents because i) it is simple, ii) it allows us to store the lattice as a relational table, and iii) it has small space requirements. Converting from a string to a list of node ids is easy via a string tokenizer.

The `Rules` table stores for each rule generated, its direction (Dir: Up or Down), the top and bottom node ids, the confidence and support, and the rule LHS and RHS. The figure shows some of the example rules shown in Figure 2. For instance, the first row corresponds to the uplink rule $WT \stackrel{3,1.0}{\longrightarrow} A\ [C, none]$, between nodes 6 and 8.

These two tables are sufficient to support all of the interactive itemset and rule exploration functionality of MIRAGE.

5 Minimal Association Rules: Theory and Algorithms

Having outlined the interactive rule and itemset visualization and exploration features of MIRAGE, we now present the theory behind minimal association rules as well as algorithms for rule generation. We begin with a formal characterization of the set of minimal exact and inexact association rules.

Table: `Nodes`

ID	Itemset	Parents	Children	Sup	Tidset	Level
1	"C"	' '	"4 2 5"	6	' '	1
2	"CW"	"1"	"7 6"	5	' '	2
4	"CD"	"1"	"7"	4	' '	2
5	"CT"	"1"	"8"	4	' '	2
6	"CWA"	"2"	' '	3	' '	3
7	"CDW"	"4 2"	' '	3	' '	3
8	"CWTA"	"6 5"	' '	3	' '	4

Table: `Rules`

Dir	Top	Bot	LHS	RHS	Conf	Sup
Up	6	8	"WT"	"A [C, none]"	1.0	3
Down	6	8	"A"	"T [CW, none]"	0.75	3
Down	1	8	"C"	"WT [none, A]"	0.5	3
Down	1	8	"C"	"TA [none, W]"	0.5	3

Figure 9: Database Tables in MIRAGE

5.1 Characterizing Minimal Rules

Let X_1 and X_2 be any itemsets. We know that an association rule is an implication $X_1 \xrightarrow{q,p} X_2$, where $X_1 \cap X_2 = \emptyset$. We define an *extended association rule* as an implication $X_1 \xrightarrow{q,p} X_2$, where X_1 and X_2 are not necessarily disjoint. The following lemmas will prove useful.

Lemma 5.1 ([2, 16]) *The support of an itemset X is equal to the support of its closure, i.e., $\pi(X) = \pi(c_{it}(X))$.*

Lemma 5.2 ([4]) *If $X_1 \subseteq X_2$, then $t(X_2) \subseteq t(X_1)$, and $i(t(X_1)) \subseteq i(t(X_2))$, i.e., $c_{it}(X_1) \subseteq c_{it}(X_2)$.*

Lemma 5.3 *The association rule $X_1 \xrightarrow{q,p} X_2$ is equivalent to the extended rule $X_1 \xrightarrow{q_1,p_1} X_1 \cup X_2$, i.e., $q = q_1$ and $p = p_1$.*

PROOF: Obvious, since the rule antecedent X_1 is unchanged, and the rule support $q = \pi(X_1 \cup X_2) = \pi(X_1 \cup (X_1 \cup X_2)) = q_1$. ■

Lemma 5.4 ([2, 16]) *The association rule $X_1 \xrightarrow{q,p} X_2$ is equivalent to the extended rule $c_{it}(X_1) \xrightarrow{q_1,p_1} c_{it}(X_2)$, i.e., $q = q_1$ and $p = p_1$.*

Lemma 5.4 says that a rule between any two itemsets is exactly the same as the extended rule between their respective closures.

Corollary 5.5 *The association rule $X_1 \xrightarrow{q,p} X_2$ is equivalent to the extended 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 by Lemma 5.4 and Lemma 5.3. ■

Lemma 5.6 ([11, 2, 16]) *An extended association rule $X_1 \xrightarrow{p=1.0} X_2$ has confidence $p = 1.0$ if and only if $X_2 \subseteq X_1$, or equivalently if and only if $t(X_2) \subseteq t(X_1)$.*

5.1.1 Minimal Exact Rules

Theorem 5.7 *Let $R_j : X_1^j \xrightarrow{q_j,p_j} X_2^j$ be an association rule (with $X_1^j \cap X_2^j = \emptyset$), and let $\mathcal{R} = \{R_1, \cdots, R_n\}$ be a set of association rules that satisfy the following conditions for all $1 \leq j \leq n$:*

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

Then all association rules $R_j \in \mathcal{R}$ are equivalent to the extended rule $I_1 \xrightarrow{q,1.0} I_2$.

PROOF: Consider any rule $R_j = X_1^j \xrightarrow{q,1.0} X_2^j$. Then the support of the rule is given as $q = |t(X_1^j \cup X_2^j)|$ and its confidence $p = \frac{|t(X_1^j \cup X_2^j)|}{|t(X_1^j)|}$. By Lemma 5.6 we have $t(X_1^j) \subseteq t(X_2^j)$, giving $|t(X_1^j \cup X_2^j)| = |t(X_1^j) \cap t(X_2^j)| = |t(X_1^j)|$. Thus $q = |t(X_1^j)|$, and $p = \frac{|t(X_1^j)|}{|t(X_1^j)|} = 1$.

Since $t(X_1^j) \subseteq t(X_2^j)$, by Lemma 5.2, we have $i(t(X_1^j)) \supseteq i(t(X_2^j))$, i.e, $c_{it}(X_1^j) \supseteq c_{it}(X_2^j)$. Since $X_1 \subseteq (X_1 \cup X_2)$, by Lemma 5.2 we have $c_{it}(X_1^j) \subseteq c_{it}(X_1^j \cup X_2^j)$. By Lemma 5.1 we also have $\pi(c_{it}(X_1^j \cup X_2^j)) = \pi(X_1^j \cup X_2^j)$, i.e., $|t(c_{it}(X_1^j \cup X_2^j))| = |t(X_1^j \cup X_2^j)|$.

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

The theorem above says that all exact confidence rules are equivalent to those that are directed from a closed itemset I_1 to its closed subset I_2 (since $I_2 \subseteq I_1$). There are two cases to consider:

1. $I_2 = I_1$: This leads to generation of self rules directed form I_1 to itself.
2. $I_2 \subset I_1$: This leads to generation of uplink rules directed from I_1 to its closed proper subset I_2.

5.1.2 Minimal Inexact Rules

Theorem 5.8 *Let $R_j : X_1^j \xrightarrow{q_j,p_j} X_2^j$ be an association rule (with $X_1^j \cap X_2^j = \emptyset$), and let $\mathcal{R} = \{R_1, \cdots, R_n\}$ be a set of association rules that satisfy the following conditions for all $1 \leq j \leq n$:*

1. *$q_j = q$ (i.e., all rules have the same support).*
2. *$p_j = p < 1.0$ (i.e., all rules have same confidence).*
3. *$I_1 = c_{it}(X_1^j)$, and $I_2 = c_{it}(X_1^j \cup X_2^j)$.*

Then all association rules $R_j \in \mathcal{R}$ are equivalent to the extended rule $I_1 \xrightarrow{q,p} I_2$.

PROOF: Consider any rule $R_j = X_1^j \xrightarrow{q,p} X_2^j$. Then the support of the rule is given as $q = |t(X_1^j \cup X_2^j)|$ and its confidence as $p = q/d$, where $d = |t(X_1^j)|$. We shall show that the rule $I_1 \longrightarrow I_2$ also has support $|t(I_1 \cup I_2)| = q$ and confidence $\frac{|t(I_1 \cup I_2)|}{|t(I_1)|} = q/d$.

We have $|t(I_1)| = |t(c_{it}(X_1^j))| = |t(X_1^j)| = d$. Now consider the rule support. We have $|t(I_1 \cup I_2)| = |t(c_{it}(X_1^j) \cup c_{it}(X_1^j \cup X_2^j))|$. Since $X_1^j \subseteq (X_1^j \cup X_2^j)$, by Lemma 5.2 $c_{it}(X_1^j) \subseteq c_{it}(X_1^j \cup X_2^j)$. Thus, $|t(I_1 \cup I_2)| = |t(c_{it}(X_1^j \cup X_2^j))| = |t(X_1^j \cup X_2^j)| = q$. ■

The theorem above says that the inexact rules are the equivalent to downlink rules from a closed itemset I_1 to its proper closed superset I_2, since $I_1 \subset I_2$.

If a rule set $\mathcal{R}$ satisfies the conditions in theorem 5.7 or in theorem 5.8, then by definition every association rule $R_i \in \mathcal{R} - \mathcal{R}^{\min}$ is redundant, where $\mathcal{R}^{\min} = \{R_i \in \mathcal{R} \mid \nexists R_j \in \mathcal{R}, R_j \prec R_i\}$ is the set of minimal association rules in $\mathcal{R}$.

5.2 Generating Minimal Rules

In this section we give the algorithms that underly MIRAGE for generating minimal exact and inexact association rules. Algorithms for minimal rule generation rely on the concept of minimal generators [2] of a closed itemset. We begin by defining this concept and then present the rule generation algorithms.

```
//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 Pass
        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

5.2.1 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) $\pi(X') = \pi(X)$. 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 proper 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. For example, consider the closed set $ACTW$ shown in Figure 1; it has support $\pi(ACTW) = 3$. The generators of $ACTW$ are $\mathcal{G}(ACTW) = \{AT, TW, ACT, ATW, CTW\}$, since they are all subsets of $ACTW$ having the same support. The minimal generators in this set are then given as $\mathcal{G}^{\min}(ACTW) = \{AT, TW\}$.

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, i.e., not a subset of any of X's parents in the lattice. 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 or parents of X.

First, any item appearing for the first time in X, and in none of X's parents, 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 [1] level-wise procedure. Note that we do not need access to the database, but rather only to X and $\mathcal{S}$, the set of its parents in the lattice.

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 $G' = i_1 i_2 \cdots i_k i_{k+1}$ 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 generator G, none of G's supersets can ever become candidate generators (this fact guarantees correctness of the algorithm). 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.

Closed Itemsets	**Minimal Generators**
C	C
CW	W
CT	T
CD	D
CWA	A
CDW	DW
CTWA	TA, TW

Table 2: Minimal Generators

As an example consider $X = ACTW$ again. We have $\mathcal{S} = \{CT, CWA\}$. There are no new items in X, i.e., $I = \emptyset$, and thus $\mathcal{G}_1 = \{A, C, T, W\}$. 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 $\mathcal{G}^{\min}(ACTW) = \{AT, TW\}$. The minimal generators for all closed itemsets in our example (from Figure 1) are given in the Table 2.

5.2.2 Rule Generation Algorithms

Given any two closed itemsets $X^\top$ and $X^\perp$, with $X^\top \subseteq X^\perp$, Figures 11 and 12 show algorithms to generate all possible minimal rules between them, based on the notion of minimal generators.

Minimal Exact Rules (Uplink and Self Rules) Exact rules are directed from the bottom node $X^\perp$ to the top node $X^\top$. Note that $X^\top \subseteq X^\perp$. There are two case that lead to an exact rule as we saw earlier: i) when $X^\top \subset X^\perp$, we get an uplink rule, and ii) when $X^\top = X^\perp$, we get a self rule. Exact (uplink and self) rules always have confidence 1.0 since an itemset always implies its own subset.

Figure 11 shows the algorithm for generating minimal exact rules. Since $\mathcal{G}^{\min}(X^\perp)$ consists of the minimal sets that are equivalent to $X^\perp$, every minimal generator $X_g^\perp \in \mathcal{G}^{\min}(X^\perp)$ forms a potential LHS (left hand side) for a minimal rule. Furthermore, as required by Theorem 5.7, $c_{it}(LHS) = c_{it}(X_g^\perp) = X^\perp$. Likewise every minimal set $X_g^\top \in \mathcal{G}^{\min}(X^\top)$ can serve as a potential RHS (right hand side). Since we require RHS to be disjoint from LHS, we set $RHS = X_g^\top - X_g^\perp$ (thus $LHS \cap RHS = \emptyset$). If the remaining two conditions in Theorem 5.7 are met (i.e., $c_{it}(RHS) = X^\top$ and $c_{it}(LHS \cup RHS) = X^\perp$), we can generate a 100% confidence rule $LHS \longrightarrow RHS$.

```
//X^⊤, X^⊥ are closed itemsets, with X^⊤ ⊆ X^⊥
GenerateExactRules (X^⊤, X^⊥):
    G_1 = G^min(X^⊤);
    G_2 = G^min(X^⊥);
    R = ∅; // minimal rule set
    //All exact rules between minimal generators
    for all X_g^⊥ ∈ G_2
        for all X_g^⊤ ∈ G_1
            LHS = X_g^⊥; RHS = (X_g^⊤ − X_g^⊥);
            if c_it(RHS) = X^⊤ ∧ c_it(LHS ∪ RHS) = X^⊥
                q = π(X^⊤);
                (I^⊤, I^⊥) = compute-extension (X^⊤, X_g^⊤, X^⊥, X_g^⊥);
                R = R ∪ (LHS --q,1.0--> RHS[I^⊤, I^⊥])
    return R^min; //remove redundant rules
```

Figure 11: Minimal Exact Rule Generation

Minimal Inexact Rules (Downlink Rules) Figure 12 shows the algorithm for generating minimal inexact rules. Downlink inexact rules are directed from the top node $X^\top$ to the bottom node $X^\perp$. As before, every $X_g^\top \in \mathcal{G}^{\min}(X^\top)$ forms a potential LHS, and every $X_g^\perp \in \mathcal{G}^{\min}(X^\perp)$ forms a potential RHS for a rule. To ensure disjointness, we set $RHS = X_g^\perp - X_g^\top$. As required by Theorem 5.8, $c_{it}(LHS) = c_{it}(X') = X^\top$, and if $c_{it}(LHS \cup RHS) = X^\perp$ then we can generate an inexact rule $LHS \longrightarrow RHS$.

As an example of generating exact and inexact rules, let $X^\top = ACW$ and $X^\perp = ACTW$, be two closed sets from Figure 1. We have $\mathcal{G}^{\min}(ACW) = \{A\}$ and $\mathcal{G}^{\min}(ACTW) = \{AT, TW\}$, as given in Table 2. The possible exact uplink rules are $AT \longrightarrow (A - AT)$, but since $A - AT = \emptyset$ this rule is not possible. The other possibility is $TW \longrightarrow (A - TW)$, which gives us the rule $TW \xrightarrow{3,1.0} A$ (which is the first rule shown in the screenshot of MIRAGE in Figure 2).

$//X^\top, X^\perp$ are closed itemsets, with $X^\top \subseteq X^\perp$
GenerateInexactRules $(X^\top, X^\perp)$:
  $G_1 = \mathcal{G}^{\min}(X^\top)$;
  $G_2 = \mathcal{G}^{\min}(X^\perp)$;
  $\mathcal{R} = \emptyset$; // non-redundant rule set
  //All inexact rules between minimal generators
  for all $X_g^\top \in G_1$
    for all $X_g^\perp \in G_2$
      $LHS = X_g^\top; RHS = (X_g^\perp - X_g^\top)$;
      if $c_{it}(LHS \cup RHS) = X^\perp$
        $q = \pi(X^\perp)$; $p = \frac{\pi(X^\perp)}{\pi(X^\top)}$
        $(I^\top, I^\perp)$ = compute-extension $(X^\top, X_g^\top, X^\perp, X_g^\perp)$;
        $\mathcal{R} = \mathcal{R} \cup (LHS \xrightarrow{q,p} RHS[I^\top, I^\perp])$
  return $\mathcal{R}^{\min}$; //remove redundant rules

Figure 12: Minimal Inexact Rule Generation

A possible downlink inexact rule 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 rules between ACW and $ACTW$ are $\mathcal{R} = \{TW \xrightarrow{3,1.0} A, A \xrightarrow{3,3/4} T, A \xrightarrow{3,3/4} TW\}$. However, since $A \xrightarrow{3,3/4} T$ is more general than $A \xrightarrow{3,3/4} TW$, we get $\mathcal{R}^{\min} = \{A \xrightarrow{3,3/4} T\}$ as the set of minimal rules (which is shown as the second rule in the screenshot of MIRAGE in Figure 2).

Computing $I^\top$ and $I^\perp$: A minimal rule between top node $X^\top$ and bottom node $X^\perp$, with $X^\top \subseteq X^\perp$, is of the form $L \xrightarrow{q,p} R[I^\top, I^\perp]$, where L is the antecedent, and R the consequent. We now give an algorithm to compute $I^\top$ and $I^\perp$.

Recall that $I^\top$ is the set of items that can be added to both top and bottom nodes' representatives, i.e., to both L and R. $I^\perp$ is the set of items that can be added only to the bottom node's representative, i.e., only to L in case of an uplink rule, and only to R in case of a downlink rule. Also $I^\top \cap I^\perp = \emptyset$.

$//X^\top, X^\perp$ are closed itemsets, with $X^\top \subseteq X^\perp$
$//X_g^\top, X_g^\perp$ are minimal generators of $X^\top, X^\perp$, respv.
compute-extension $(X^\top, X_g^\top, X^\perp, X_g^\perp)$:
  $I^\top = X^\top - (X_g^\top \cup X_g^\perp)$;
  $I^\perp = X^\perp - (X_g^\perp \cup X^\top)$;
  return $I^\top, I^\perp$;

Figure 13: Compute $I^\top$ and $I^\perp$

Figure 13 gives the algorithm to compute $I^\top$ and $I^\perp$. The algorithm is straight forward; we give an intuitive explanation before formally proving it correct. We know that every item in $X^\top$ also appears in $X^\perp$, since $X^\top \subseteq X^\perp$. Thus all items in $X^\top$ could potentially be added to L and R, except items that already appear in the minimal generators of $X^\top$ and $X^\perp$. Also, all items in $X^\perp$

can potentially be added exclusively to L or R derived from $X^{\perp}$, except items that already appear in $X^{\top}$ (since they are common to both), and items in $X_g^{\perp}$ (since this is the minimal generator from which L or R has been derived).

Theorem 5.9 *Let $X^{\top}$ and $X^{\perp}$ be closed sets with $X^{\top} \subseteq X^{\perp}$, and let $X_g^{\top} \in \mathcal{G}^{\min}(X^{\top})$ and $X_g^{\perp} \in \mathcal{G}^{\min}(X^{\perp})$. Then $I^{\top} = X^{\top} - (X_g^{\top} \cup X_g^{\perp})$, and $I^{\perp} = X^{\perp} - (X_g^{\perp} \cup X^{\top})$.*

PROOF: Let $L \xrightarrow{q,p} R$ be an uplink rule, where $L = X_g^{\perp}$ and $R = X_g^{\top} - X_g^{\perp}$. Then $I^{\top} = (X^{\top} - R) - L$, i.e., every item in $X^{\top}$ is potentially common, except those that appear in R and L. After expanding we get $I^{\top} = (X^{\top} - (X_g^{\top} - X_g^{\perp})) - X_g^{\perp})$. Simplifying, we get $I^{\top} = X^{\top} - X_g^{\top} - X_g^{\perp} = X^{\top} - (X_g^{\top} \cup X_g^{\perp})$.

For computing $I^{\perp}$ we need to exclude items in L and R, as well as $I^{\top}$, i.e., $I^{\perp} = ((X^{\perp} - L) - R) - I^{\top}$. $I^{\perp} = X^{\perp} - (L \cup R) - (X^{\top} - R - L) = X^{\perp} - ((L \cup R) \bigcup (X^{\top} - (L \cup R))) = X^{\perp} - (L \cup R \; cup X^{\top})$. Plugging in values of L and R, and simplifying we get, $I^{\perp} = X^{\perp} - (X_g^{\perp} \cup (X_g^{\top} - X_g^{\perp}) \; cup X^{\top}) = X^{\perp} - (X_g^{\perp} \cup X_g^{\top} \cup X^{\top}) = X^{\perp} - (X_g^{\perp} \cup X^{\top})$.

Let $L \xrightarrow{q,p} R$ be a downlink rule, where $L = X_g^{\top}$ and $R = X_g^{\perp} - X_g^{\top}$. Then $I^{\top} = (X^{\top} - L) - R$. Substituting and simplifying, we get $I^{\top} = (X^{\top} - X_g^{\top}) - (X_g^{\perp} - X_g^{\top}) = X^{\top} - X_g^{\top} - X_g^{\perp} = X^{\top} - (X_g^{\top} \cup X_g^{\perp})$. We have $I^{\perp} = X^{\perp} - L - R - I^{\top}$. Similar to the uplink case, it simplifies to $I^{\perp} = X^{\perp} - (X_g^{\perp} \cup X^{\top})$.

■

5.3 Representative Rule Set

Let $\mathcal{R} = \{R_1, R_2, \cdots, R_n\}$ be the any set of rules, and let $\mathcal{R}' \subseteq \mathcal{R}$. We say that $\mathcal{R}'$ *entails* $R_i \in \mathcal{R}$, denoted as $\mathcal{R}' \vdash R_i$, if either $\exists R_j \in \mathcal{R}'$ such that $R_j \preceq R_i$, or we can deduce a rule R_j from rules in $\mathcal{R}'$ such that $R_j \preceq R_i$. That is R_i has the same support and confidence as R_j and it can be obtained by adding items to the antecedent and/or consequent of R_j. If $\mathcal{R}' \vdash R_i$ for all $R_i \in \mathcal{R}$, we say that $\mathcal{R}'$ is a *representative set* for $\mathcal{R}$, and denote it as $\mathcal{R}' \vdash \mathcal{R}$.

Let $\mathcal{C}$ be the set of all closed frequent itemsets mined from a database. Let $X_i, X_j \in \mathcal{C}$, and assume without loss of generality that $X_i \subseteq X_j$. Assume further that X_i and X_j are adjacent (or the same), i.e., $\nexists X_k$, such that $(X_i \subset X_k \subseteq X_j)$.

Let $\mathcal{R}_{ij}$ denote the set of exact and inexact association rules that can be generated from any such pair of adjacent closed itemsets X_i and X_j using the algorithms outlined above. Let $\mathcal{R}_{ij}^{\min}$ be the set of minimal rules in $\mathcal{R}_{ij}$, and let $\Re^{\min} = \bigcup_{i,j} \mathcal{R}_{ij}^{\min}$ be the set of all minimal inexact and exact rules generated from adjacent frequent closed itemsets X_i and X_j. Finally, let $\mathcal{R}^a$ denote the set of all possible association rules.

Here we prove that the set of minimal association rules $\Re^{\min}$ taken over all adjacent frequent closed itemsets, constitutes a representative set for the set of all possible association rules $\mathcal{R}^a$, i.e., $\Re^{\min} \vdash \mathcal{R}^a$.

Lemma 5.10 $\Re^{\min} \subseteq \mathcal{R}^a$.

PROOF: Let $R : X \xrightarrow{q,p} Y$ be any minimal rule $R \in \Re^{\min}$, then the rule $R' : X \longrightarrow Z - X$ is an association rule generated from the frequent itemset $Z = XY$. It is obvious that $R = R'$, thus $\mathcal{R} \in \mathcal{R}^a$. ■

Lemma 5.11 *Let $R : (X_1 \xrightarrow{q,p} X_2) \in \mathcal{R}^a$. If $c_{it}(X_1)$ and $c_{it}(X_2)$ are adjacent closed sets, then $\Re^{\min} \vdash R$.*

PROOF: We know that $\exists X_1' \in \mathcal{G}^{\min}(c_{it}(X_1))$, with $X_1' \subseteq X_1$ and $\exists X_2' \in \mathcal{G}^{\min}(c_{it}(X_2))$, with $X_2' \subseteq X_2$. R is an association rule means that $X_1 \cap X_2 = \emptyset$, which implies that $X_1' \cap X_2' = \emptyset$. Since $c_{it}(X_1)$ and $c_{it}(X_2)$ are adjacent closed sets, by Theorems 5.7 and 5.8, $R' : (X_1' \longrightarrow X_2') \in \Re^{\min}$. Since $R' \preceq R$, we obtain $\Re^{\min} \vdash R$. ■

Lemma 5.12 *Let $R : (X_1 \xrightarrow{q,p} X_2) \in \mathcal{R}^a$. Let $c_{it}(X_1) \not\subseteq c_{it}(X_2)$, and $c_{it}(X_2) \not\subseteq c_{it}(X_1)$,i.e., the two closed sets are non-comparable. Then R is equivalent to the rule $R' : I_1 \longrightarrow I_2$ between comparable closed sets $I_1 = c_{it}(X_1)$ and $I_2 = c_{it}(X_1 \cup X_2)$.*

PROOF: Immediate by Corollary 5.5. ■

Lemma 5.13 *Let $R : (X_1 \xrightarrow{q,p} X_2) \in \mathcal{R}^a$. Let $I_1 = c_{it}(X_1)$ and $I_2 = c_{it}(X_2)$ be non-adjacent, but related by the $\subseteq$ relation. Then $\Re^{\min} \vdash R$.*

PROOF: We know that $\exists X_1' \in \mathcal{G}^{\min}(c_{it}(X_1))$, with $X_1' \subseteq X_1$ and $\exists X_2' \in \mathcal{G}^{\min}(c_{it}(X_2))$, with $X_2' \subseteq X_2$. We shall show that the rule $R' : X_1' \longrightarrow X_2'$ can be deduced from $\Re^{\min}$. Since $R' \preceq R$, it will follow that $\Re^{\min} \vdash R$.

There are two cases to consider: 1) $I_1 \subset I_2$, 2) $I_2 \subset I_1$. Note that $I_1 \neq I_2$ since they are non-adjacent.

Case 1) – $\mathbf{I_1 \subset I_2}$: Since they are non-adjacent, there exist a chain of $k \geq 1$ distinct closed sets $I_1 = Z_0 \subset Z_1 \subset \cdots Z_k \subset Z_{k+1} = I_2$, such that Z_i and Z_{i+1} are adjacent for $0 \leq i \leq k$. Let $Z_i' \in \mathcal{G}^{\min}(Z_i)$ denote a minimal generator of the closed set Z_i.

By algorithm `GenerateInexactRules` (Figure 12), if $Z_{i+1}' - Z_i' \neq \emptyset$, then $R_i' : (Z_i' \longrightarrow Z_{i+1}' - Z_i') \in \Re^{\min}$. If $Z_{i+1}' - Z_i' = \emptyset$, then $Z_{i+1}' \subseteq Z_i'$. But this is a contradiction, since the minimal generator of a superset Z_{i+1} cannot be a subset of the minimal generator of a subset Z_i. Thus the rule $R_i' : (Z_i' \longrightarrow Z_{i+1}') \in \Re^{\min}$ for any $Z_i' \in \mathcal{G}^{\min}(Z_i)$ and $Z_{i+1}' \in \mathcal{G}^{\min}(Z_{i+1})$. It follows that there exists a chain of minimal generators $X_1' = Z_0', Z_1', \cdots, Z_k', Z_{k+1}' = X_2'$ such that the rule $Z_i' \longrightarrow Z_{i+1}' - Z_i' \in \Re^{\min}$. From this we can deduce the rule $R' : X_1' \longrightarrow X_2'$. It is clear that $R' \preceq R$.

Case 2) – $\mathbf{I_2 \subset I_1}$: Since they are non-adjacent, there exist a chain of $k \geq 1$ distinct closed sets $I_1 = Z_0 \supset Z_1 \supset \cdots Z_k \supset Z_{k+1} = I_2$, such that Z_{i+1} and Z_i are adjacent for $0 \leq i \leq k$. Let $Z_i' \in \mathcal{G}^{\min}(Z_i)$ denote a minimal generator of the closed set Z_i.

By algorithm `GenerateExactRules` (Figure 11), if $Z_{i+1}' - Z_i' \neq \emptyset$, then $R_i' : (Z_i' \longrightarrow Z_{i+1}' - Z_i') \in \Re^{\min}$. If $Z_{i+1}' - Z_i' = \emptyset$, then $Z_{i+1}' \subseteq Z_i'$. Since minimal generators of two distinct closed sets cannot be equal, we have $Z_{i+1}' \neq Z_i'$. Thus $Z_{i+1}' \subset Z_i'$. But in this case the inexact rule $Z_{i+1}' \longrightarrow Z_i' - Z_{i+1}' \in \Re^{\min}$.

It follows that there exists a chain of minimal generators $X_1' = Z_0', Z_1', \cdots, Z_k', Z_{k+1}' = X_2'$ such that either the exact rule $Z_i' \longrightarrow Z_{i+1}' - Z_i' \in \Re^{\min}$ or the inexact rule $Z_{i+1}' \longrightarrow Z_i' - Z_{i+1}' \in \Re^{\min}$. From this we can deduce the exact rule $R' : X_1' \longrightarrow X_2'$. It is clear that $R' \preceq R$. ■

Theorem 5.14 $\Re^{\min} \vdash \mathcal{R}^a$.

PROOF: Let $R : (X_1 \xrightarrow{q,p} X_2) \in \mathcal{R}^a$. Let $I_1 = c_{it}(X_1)$ and $I_2 = c_{it}(X_2)$. Consider 3 cases:

Case 1): If I_1 and I_2 are adjacent, then by Lemma 5.11 $\Re^{\min} \vdash R$.

Case 2): If I_1 and I_2 are non-adjacent but comparable, then by Lemma 5.13 $\Re^{\min} \vdash R$.

Case 3): If I_1 and I_2 are non-comparable, then by Lemma 5.12 R is equivalent to the rule between comparable closed sets I_1 and $I_2' = c_{it}(X_1 \cup X_2)$. This reduces to case 2). Thus $\Re^{\min} \vdash \mathcal{R}^a$. ■

6 Experimental Evaluation

In this section we show some experimental evidence on the effectiveness as well as the interactive performance of MIRAGE. All experiments described below were performed on a 400MHz Pentium PC with 256MB of memory, running RedHat Linux 6.0. MIRAGE is coded in the Java programming language.

We test the framework suing several real and synthetic datasets to evaluate the system performance. Table 3 shows the characteristics of the real and synthetic datasets used in our evaluation.

All datasets except the PUMS (pumsb and pumsb*) sets are taken from UC Irvine Machine Learning Database Repository. The PUMS datasets contain census data, and pumsb* is the same as pumsb without items of support equals to 80% or more. The connect and chess datasets are obtained from their respective game steps. Finally, the mushroom set contains characteristics of various species of mushrooms. Real datasets are usually very dense, i.e. they produce many long frequent itemsets even for a very high value of support. 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 3) synthetic datasets for our study.

Database	# Items	Record Length	# 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

Table 3: Database Characteristics

6.1 Redundant vs. Minimal Rules

In the first set of experiments we show the effectiveness of the minimal association rule framework in cutting down the number of rules presented to the user, when compared against presenting all (redundant) association rules. Table 4 shows the database used with two different values of minimum support $\pi^{\min}$ (expressed as fraction of database records, instead of an absolute number of records). The next three columns show the number of all association rules ($|\mathcal{R}|$), the number of all minimal rules ($|\mathcal{R}^{\min}|$), and the rule reduction ratio ($\frac{|\mathcal{R}^{\min}|}{|\mathcal{R}|}$). The results indicate that the number of minimal rules can be much smaller than the number of all association rules; the reduction ratio of minimal rules w.r.t all rules, can range from a factor of 2 to more than 3000 times!

6.2 Interactive Performance

Given that MIRAGE is an entire framework for interactive graphical exploration of minimal association rules, it is difficult to quantitatively evaluate it. Here we give some results on the interactive

Database	$\pi^{\min}$	$\lvert\mathcal{R}\rvert$	$\lvert\mathcal{R}^{\min}\rvert$	$\frac{\lvert\mathcal{R}^{\min}\rvert}{\lvert\mathcal{R}\rvert}$
chess	80%	552564	27711	20
chess	70%	8171198	152074	54
connect	97%	8092	1116	7
connect	90%	3640704	18848	193
mushroom	40%	7020	475	15
mushroom	20%	19191656	5741	3343
pumsb*	60%	2358	192	12
pumsb*	40%	5659536	13479	420
pumsb	95%	1170	267	4
pumsb	85%	1408950	44483	32
T20I12D100K	0.5%	40356	2642	15
T40I8D100K	1.5%	1609678	11379	142
T10I4D100K	0.5%	2216	1231	1.8
T10I4D100K	0.1%	431838	86902	5.0
T20I4D100K	1.0%	2736	1738	1.6
T20I4D100K	0.25%	391512	89963	4.4

Table 4: Redundant vs. Minimal Rules

performance of MIRAGE while generating minimal rules. In out experiments we measure the average rule generation time (ARGT) for different values of $\pi^{\min}$. We also measure the height of lattice, which is given as the maximum level difference between any two lattice nodes. As the support is lowered more and longer closed sets are mined, and the lattice height increases.

$\pi^{\min}$	ARGT (msec)	Height
0.94	0.70	4
0.945	0.69	4
0.95	0.45	3
0.955	0.45	3
0.96	0.60	3
0.965	0.71	3
0.97	0.46	2
0.975	0.87	2
0.98	2.73	1

Table 5: Average RGT: Pumsb

$\pi^{\min}$	ARGT (msec)	Height
0.575	2.15	7
0.6	2.58	7
0.625	1.44	6
0.65	0.17	3
0.7	0.12	3
0.75	0.44	2
0.8	0.15	2

Table 6: Average RGT: T10I4D100K

Tables 5 and 6, the interactive rule generation times for one real (pumsb) and one synthetic dataset (T10I4D100K). We can observe that the rule generation is very fast, taking only a few milliseconds per pair of lattice nodes. Similar results were obtained for other datasets.

7 Conclusions

This paper presents a novel framework for the mining and exploration of association rules, based on the concept of minimal association rules. Minimal association rules are the most general rules (i.e., having most general antecedent and consequent) that satisfy a given support and confidence threshold. Minimal rules are typically a lot less than the the full set of association rules, and this helps address the combinatorial rule explosion problem. We show formally that the set of minimal rules is a representative set for all association rules.

We proposed MIRAGE, a tool for interactive graphical exploration of minimal association rules, which uses lattice-based interactive rule visualization approach, displaying the rules in a very compact form; all association rules can also be generated if desired. Hence, there is no information loss. MIRAGE uses a database back-end for effective, persistent rule management for easy retrieval at a later point in time. As part of future work, we plan to implement a more flexible approach to user-specified constraints during rule exploration.

References

[1] 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.

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

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

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

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

[6] Jianchao Han and Nick Cercone. Ruleviz: A model for visualizing knowledge discovery process. In *6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 244 – 253, Boston, MA, USA, August 2000.

[7] Heike Hofmann, Arno P. J. M. Siebes, and Adalbert F. X. Wilhelm. Visualizing association rules with interactive mosaic plots. In *6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 227 – 235, Boston, MA, USA, August 2000.

[8] Daniel Keim. Visual techniques for exploring databases. In *3rd Int. Conf. Knowledge Discovery and Data Mining*, August 1997.

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

[10] B. Liu, W. Hsu, K. Wang, and S. Chen. Visually aided exploration of interesting association rules. In *3rd Pacific-Asia Conference on Methodologies for Knowledge Discovery and Data Mining*, April 1999.

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

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

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

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

[15] Pak Chung Wong, Paul Whitney, and Jim Thomas. Visualizing association rules for text mining. In *Information Visualization*, Richland, WA, USA, 1999. Pacific Northwest National Laboratory.

[16] M. J. Zaki. Generating non-redundant association rules. In *6th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining*, August 2000.

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

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

[19] G. Zweiger. Knowledge discovery in gene-expression microarray data: mining the information output of the genome. *Trends in Biotechnology*, 17:429–436, November 1999.