Concise Representation of Multi-level Association Rules using MinMaxExact Rules

R. Vijaya Prakash, A. Govardhan, SSVN.Sarma

Abstract—Association Rule mining plays an important role in the discovery of knowledge and information. Association Rule mining discovers huge number of rules for any dataset for different support and confidence values, among this many of them are redundant, especially in the case of multi-level datasets. Mining non-redundant Association Rules in multi-level dataset is a big concern in field of Data mining. In this paper, we present a definition for redundancy and a concise representation called Reliable Exact basis for representing non-redundant Association Rules from multi-level datasets. The given non-redundant Association Rules are loss less representation for any datasets.

Index Terms—Association Rules, Itemsets, Non-Redundant Reliable Rules.

I. INTRODUCTION

The huge amount of the extracted rules is a big problem for Association Rule mining. Especially, many of the extracted rules are considered redundant since they produce the same meaning to the user or extracted rules can be replaced by other rules. Many efforts have been made on reducing the size of the extracted rule set. There are number of representations of frequent patterns have been proposed, one of them, is the closed itemsets, is of particular interest as they can be applied for generating non-redundant rules [10,12,18]. The use of frequent closed itemsets presents a clear promise to reduce the number of extracted rules [13, 17, 19].

However, these approaches have only dealt with redundancy in single level datasets. Multi-level datasets in which the items are not all at the same concept level contain information at different abstract levels. The approaches used to find frequent itemsets in single level datasets miss information, as they only look at one level in the dataset. Thus techniques that consider all the levels are needed [6, 7, 8, 9].

However, rules derived from multi-level datasets can have the same issues with redundancy as those from a single level dataset. While approaches used to remove redundancy in single level datasets [13, 17, 19] can be adapted for use in one rule at a given level gives the same information as another rule at a different level.

In this paper, we present a Reliable basis representation of non-redundant Association Rules. We then look into this hierarchical redundancy and propose an approach from which more non-redundant rules can be derived. We use the same definition of non-redundant rules in single level datasets, but to this definition we add a requirement that considers the different levels of the item(s) in determining the redundancy rule. By doing so, more redundant Association Rules can be eliminated. We also show that it is possible to derive all of the Association Rules, without lose of information.

The paper is organized as follows. Section 2 briefly discusses some related work. In Section 3, we first discuss the algorithms for generating frequent patterns in multilevel datasets and then we present the definition of hierarchical redundancy and introduce our approach for deriving multilevel non-redundant exact basis rules. In Section 4, we discuss the redundancy in Association Rules and present a definition to redundant rules. Experiments and results are presented in Section 5. Finally, Section 6 concludes the paper.

II. RELATED WORK

The approaches proposed in [13, 18] make use of the closure of the Galois connection [4] to extract non-redundant rules from frequent closed itemsets instead of from frequent itemsets. One difference between the two approaches is the definition of redundancy. The approach proposed in [18] extracts the rules with shorter antecedent and shorter consequent as well among rules which have the same confidence, while the method proposed in [13] defines that the non-redundant rules are those which have minimal antecedents and maximal consequents. The definition proposed in [17] is similar to that of [13]. However, the requirement to redundancy is relaxed, and the lesser requirement makes more rules to be considered redundant and thus eliminated. Most importantly, [17] proved that the elimination of such redundant rules increase the belief to the extracted rules and the capacity of the extracted non-redundant rules for solving problems. However, the work mentioned above has only focused on datasets where all items are at the same concept level. Thus they do not need to consider redundancy that can occur when there is a hierarchy among the items.

A multi-level dataset is the one which has an implicit taxonomy or concept tree, like shown in Figure 1. The items in the dataset exist at the lowest concept level but are part of a hierarchical structure and organization.
Thus for level of the taxonomy but it also belongs to the example, ‘Old Mills’ is an item at the lowest higher concept category of ‘bread’ and also the more refined category ‘white bread’.

Because of the hierarchical nature of a multi-level dataset, a new approach to finding frequent itemsets for multi-level datasets has to be considered. Work has been done in adapting approaches originally made for single level datasets into techniques usable on multi-level datasets. The paper in [5][7] shows one of the earliest approaches proposed to find frequent itemsets in multi-level datasets and later was revisited in [6]. This work primarily focused on finding frequent itemsets at each of the levels in the dataset and did not focus heavily on cross-level itemsets (those itemsets that are composed of items from two or more different levels). Referring to Figure 1 for an example, the frequent itemset {‘Dairyland-2%-milk’, ‘white-bread’} is a cross-level itemset as the first item is from the lowest level, while the second item is from a different concept level. In fact the cross-level idea was an addition to the work being proposed. Further work proposed an approach which included finding cross-level frequent itemsets [15]. This later work also performs more pruning of the dataset to make finding the frequent itemsets more efficient. However, even with all this work the focus has been on finding the frequent itemsets as efficiently as possible and the issue of quality and/or redundancy in single level datasets. Some brief work presented by [6] discusses removing rules which are hierarchically redundant, but it relies on the user giving an expected confidence variation margin to determine redundancy. There appears to be a void in dealing with hierarchical redundancy in Association Rules derived from multi-level datasets. This work attempts to fill that void and show an approach to deal with hierarchical redundancy without losing any information.

III. MINING FREQUENT PATTERNS

From the beginning of Association Rule mining in [1][3], the first step has always been to find the frequent patterns or itemsets. The simplest way to do this is through the use of the Apriori algorithm [2]. However, Apriori is not designed to work on extracting frequent itemsets at multiple levels in a multi-level dataset. It is designed for use on single level datasets. But, it has been adapted for multi-level datasets.

One adaptation of Apriori to multi-level datasets is the ML_T2L1 algorithm [5, 6]. The ML_T2L1 algorithm uses a transaction table that has the hierarchy information encoded into it. Each level in the dataset is processed individually. Firstly, level 1 (the highest level in the hierarchy) is analyzed for large 1-itemsets using Apriori. The list of level 1 large 1-itemsets is then used to filter and prune the transaction dataset of any item that does not have an ancestor in the level 1 large 1-itemset list and remove any transaction which has no frequent items (thus contains only infrequent items when assessed using the level 1 large 1-itemset list). From the level 1 large 1-itemset list, level 1 large 2-itemsets are derived (using the filter dataset). Then level 1 large 3-itemsets are derived and so on, until there are no more frequent itemsets to discover at level 1. Since ML_T2L1 defines that only the items that are descendant from frequent items at level 1 (essentially they must descend from level 1 large 1-itemsets) can be frequent themselves, the level 2 itemsets are derived from the filtered transaction table. For level 2, the large 1-itemsets are discovered, from which the large 2-itemsets are derived and then large 3-itemsets etc. After all the frequent itemsets are discovered at level 2, the level 3 large 1-itemsets are discovered (from the same filtered dataset) and so on. ML_T2L1 repeats until either all levels are searched using Apriori or no large 1-itemsets are found at a level.

As the original work shows [5,6], ML_T2L1 does not find cross-level frequent itemsets. We have added the ability for it to do this. At each level below 1 (so starting at level 2) when large 2-itemsets or later are derived the Apriori algorithm is not restricted to just using the large n-1-itemsets at the current level, but can generate combinations using the large itemsets from higher levels. The only restrictions on this are that the derived frequent itemset(s) can not contain an item that has an ancestor-descendant relationship with another item within the same itemset and that the minimum support threshold used is that of the current level being processed (which is actually the lowest level in the itemset).

A second, more recent adaptation of Apriori for use in multi-level datasets is a top-down progressive deepening method by [15]. This approach was developed to find level-crossing Association Rules by extending existing multi-level mining techniques and uses reduced support and refinement of the transaction table at every hierarchy level. This algorithm works very similarly to ML_T2L1 presented previously in that it uses a transaction table which has the hierarchy encoded into it and each level is processed individually, one at a time. Initially, level 1 is processed, followed by level 2, 3 and so on until the lowest level is reached and processed, or a level generates no large 1-itemsets. At each level, the large 1-itemsets are first derived and are then used to filter / prune[11] the transaction table (as described for ML_T2L1). This filtering happens at every level, not just level 1, like in ML_T2L1. Then large 2-itemsets, 3-itemsets etc are derived from the filtered table. When it comes to level 2 and lower, the itemsets are not restricted to just the current level, but can include itemsets.
from large itemset lists of higher levels. This is how level crossing Association Rules will be found. For the itemsets that span multiple levels, the minimum support threshold of the lowest level in the itemset is used as the threshold to determine whether the itemset is frequent / large.

IV. GENERATION OF NON-REDUNDANT MULTI-LEVEL ASSOCIATION RULES

The use of frequent itemsets as the basis for Association Rule mining often results in the generation of a large number of rules. This is a widely recognized problem. More recent work has demonstrated that the use of closed itemsets and generators can reduce the number of rules generated [14,16,17,18,20]. This has helped to greatly reduce redundancy in the rules derived from single level datasets. Despite this, redundancy still exists in the rules generated from multi-level datasets even when using some of the methods designed to remove redundancy. This redundancy we call hierarchical redundancy. Here in this section we first introduce hierarchical redundancy in multi-level datasets and then we detail our work to remove this redundancy without losing information.

A. Hierarchical Redundancy

Whether a rule is interesting and/or useful is usually determined through the support and confidence values that it has. However, this does not guarantee that all of the rules that have a high enough support and confidence actually convey new information. Following is an example transaction table for a multi-level dataset (Table 1).

Table 1. Simple multi-level transaction dataset.

<table>
<thead>
<tr>
<th>Transaction ID</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>[1-1-1, 1-2-1, 2-1-1, 2-2-1]</td>
</tr>
<tr>
<td>2</td>
<td>[1-1-1, 2-1-1, 2-2-2, 3-2-3]</td>
</tr>
<tr>
<td>3</td>
<td>[1-1-2, 1-2-2, 2-2-1, 4-1-1]</td>
</tr>
<tr>
<td>4</td>
<td>[1-1-1, 1-2-1]</td>
</tr>
<tr>
<td>5</td>
<td>[1-1-1, 1-2-2, 2-1-1, 2-2-1, 4-1-3]</td>
</tr>
<tr>
<td>6</td>
<td>[1-1-3, 3-2-3, 5-2-4]</td>
</tr>
<tr>
<td>7</td>
<td>[1-3-1, 2-3-1]</td>
</tr>
</tbody>
</table>

This simple multi-level transactional dataset has 3 levels with each item belonging to the lowest level. The item ID in the table store/holds the hierarchy information for each item. Thus the item 1-2-1 belongs to the first category at level 1 and for level 2 it belongs to the second sub-category of the first level 1 category. Finally at level 3 it belongs to the first subcategory of the parent category at level 2. From this transaction set we use the ML_T2L1 algorithm with the cross level add-on and a minimum support value of 4 for level 1 and 3 for levels 2 and 3. We used the ML_T2L1 algorithm to discover frequent itemsets. From these frequent itemsets, the closed itemsets and generators are derived (Table 3). The itemsets, closed itemsets and generators come from all three levels.

Finally from the closed itemsets and generators the Association Rules can be generated. In this example we use the ReliableExactRule approach presented in [17,18] to generate the exact basis rules. The discovered rules are from multiple levels and include cross-level rules (due to cross level frequent itemsets). The ReliableExactRule approach can remove redundant rules, but as we will show, it does not remove hierarchy redundancy. The rules given in Table 4 are derived from the closed itemsets and generators in Table 3 when the minimum confidence threshold is set to 0.5 or 50% (Table 4).

Table 3. Frequent closed itemsets and generators derived from the frequent itemsets in Table 2.

<table>
<thead>
<tr>
<th>No</th>
<th>Items</th>
<th>Generators</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>[1-1-1]</td>
<td>[1-1-1]</td>
</tr>
<tr>
<td>2</td>
<td>[1-1-2]</td>
<td>[1-1-1]</td>
</tr>
<tr>
<td>3</td>
<td>[1-1-3]</td>
<td>[1-1-1]</td>
</tr>
<tr>
<td>4</td>
<td>[1-1-1-1]</td>
<td>[1-1-1-1]</td>
</tr>
<tr>
<td>5</td>
<td>[1-1-1, 1-2-1, 2-1-1, 2-2-1, 4-1-3]</td>
<td>[1-1-1]</td>
</tr>
<tr>
<td>6</td>
<td>[1-1-3, 3-2-3, 5-2-4]</td>
<td>[1-1-1]</td>
</tr>
<tr>
<td>7</td>
<td>[1-3-1, 2-3-1]</td>
<td>[1-1-1]</td>
</tr>
</tbody>
</table>

Table 4. Exact basis Association Rules derived from closed itemsets and generators in Table 3.

<table>
<thead>
<tr>
<th>No</th>
<th>Rule</th>
<th>Supp</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>[2-2-1] =&gt; [1-1-1]</td>
<td>0.571</td>
</tr>
<tr>
<td>2</td>
<td>[2-2-1] =&gt; [1-1-1]</td>
<td>0.571</td>
</tr>
<tr>
<td>3</td>
<td>[2-2-1] =&gt; [1-1-1]</td>
<td>0.571</td>
</tr>
<tr>
<td>4</td>
<td>[2-2-1] =&gt; [1-1-1]</td>
<td>0.428</td>
</tr>
<tr>
<td>5</td>
<td>[2-2-1] =&gt; [1-1-1]</td>
<td>0.428</td>
</tr>
<tr>
<td>6</td>
<td>[2-2-1] =&gt; [1-1-1]</td>
<td>0.428</td>
</tr>
<tr>
<td>7</td>
<td>[2-2-1] =&gt; [1-1-1]</td>
<td>0.428</td>
</tr>
<tr>
<td>8</td>
<td>[2-2-1] =&gt; [1-1-1]</td>
<td>0.428</td>
</tr>
<tr>
<td>9</td>
<td>[2-2-1] =&gt; [1-1-1]</td>
<td>0.428</td>
</tr>
<tr>
<td>10</td>
<td>[2-2-1] =&gt; [1-1-1]</td>
<td>0.428</td>
</tr>
<tr>
<td>11</td>
<td>[2-2-1] =&gt; [1-1-1]</td>
<td>0.428</td>
</tr>
<tr>
<td>12</td>
<td>[2-2-1] =&gt; [1-1-1]</td>
<td>0.428</td>
</tr>
<tr>
<td>13</td>
<td>[2-2-1] =&gt; [1-1-1]</td>
<td>0.428</td>
</tr>
</tbody>
</table>

The ReliableExactRule algorithm lists all of the rules (in Table 4) as important and non-redundant. However, we argue that there are still redundant rules. This type of redundancy is beyond what the ReliableExactRule algorithm was designed for. Looking at the rules in Table 4 we claim that rule 4 is redundant to rule 1, rule 7 is redundant to rule 5, rule 8 is redundant to rule 6 and rule 12 is redundant to rule 10. For example, the item 2-2-1 (from rule 4) is a child of the more general/abstract item 2-2-* (from rule 1). Thus rule 4 is in fact a more specific version of rule 1. Because we know that rule 1 says 2-2-* is enough to fire the rule with consequent C, whereas rule 4 requires 2-2-1 to fire with consequent C, any item that is a descendant of 2-2-* will cause a rule to fire with...
consequent C. It does not have to be 2-2-1. Thus rule 4 is more restrictive. Because 2-2-1 is part of 2-2-* having rule 4 does not actually bring any new information to the user, as the information contained in it is actually part of the information contained in rule 1. Thus rule 4 is redundant. We define hierarchical redundancy in exact Association Rules through the following definition.

Definition 1: Let R1 = X1 => Y and R2 = X2 => Y be two exact Association Rules, with exactly the same itemset Y as the consequent. Rule R1 is redundant to rule R2 if (1) the itemset X1 is made up of items where at least one item in X1 is descendant from the items in X2 and (2) the itemset X2 is entirely made up of items where at least one item in X2 is an ancestor of the items in X1 and (3) the other non-ancestor items in X2 are all present in itemset X1.

From this definition, if for an exact Association Rule X1 => Y1 there does not exist any other rule X2 => Y2 such that at least one item in X1 shares an ancestor-descendant relationship with X2 containing the ancestor(s) and all other items X2 are present in X1, then X1 => Y1 is a non-redundant rule. To test for redundancy, we take this definition and add another condition for a rule to be considered valid. A rule X => Y is valid if it has no ancestor-descendant relationship between any items in itemsets X and Y. Thus for example 1-2-1 => 1-2-* is not a valid rule, but 1-2-1 => 1-1-3 is a valid rule. If this condition is not met by any rule X2 => Y2 when testing to see if X1 => Y1 is redundant to X2 => Y2, then X1 => Y1 is a non-redundant rule as X2 => Y2 is not a valid rule. Submit your manuscript electronically for further review.

B. Generating Exact Basis Rules

As previous work has shown [14, 17, 18] using frequent closed itemsets in the generation of Association Rules can reduce the quantity of discovered rules. Because we wish to remove redundancy on top of the redundancy already being removed, our approach uses the closed itemsets and generators to discover the non-redundant rules. [14, 17, 18] have both proposed condensed/more concise bases to represent non-redundant exact rules. Exact rules refer to rules whose confidence is 1. The proposed approach will be extended to other rules (i.e., so called approximate rules). The following definitions outline these two bases:

Definition 2: For the Min-MaxExact (MME) basis, C is the set of the discovered frequent closed itemsets. For each closed itemset c in C, Gc is the set of generators for c. From this the exact basis for min-max is:

\[ \text{MMEHR} = \{ g \rightarrow c | c \in C, g \in G_c, g \neq c \text{ and there exists no rules } g' \rightarrow c' \text{ where } c' \in C, g' \in G_{c'}, c \neq c', g' \neq c' \text{ and } g \text{ is descendant set of } g', g' \text{ has no ancestor or descendant of } c' \text{ or } g' \} \]

Definition 8: (Reliable Exact Basis without Hierarchy Redundancy) Let C be the set of frequent closed itemsets. For each frequent closed itemset c, let Gc be the set of minimal generators of c. The Reliable exact basis is:

\[ \text{REHR} = \{ g \rightarrow c | c \in C, g \in G_c, \neg(c \text{ or } c' \cup g'), c' \in C, c' \subseteq c, g' \in G_{c'}, \text{ and there exists no rules } g' \rightarrow c' \text{ where } c \neq c', g' \neq c' \text{ and } g \text{ is descendant set of } g', g' \text{ has no ancestor or descendant of } c' \text{ or } g' \} \]

Thus the algorithms to extract non-redundant multi-level rules using either MinMaxExactHR or ReliableExactHR are given as follows:

Algorithm 1: MinMaxExactHR()

Input: C a set of frequent closed itemsets G; a set of minimal generators.

Output: A set of non-redundant multilevel rules.
1. \( \text{MinMaxExact} := \emptyset \)
2. for each \( k=1 \) to \( v \)
3. for each \( k \)-generator \( g \in G_k \)
4. \( \text{nonRedundant} = \text{true} \)
5. if \( g \neq g \text{ closure} \)
6. for all \( g' \in G \)
7. if \( (g' \neq g) \)
8. if \( (g' \)is ancestor set of \( g \) and \((c' = c) \text{ or } (g = g')) \) and \((g' \)is not ancestor set of \( c') \)
9. then \( \text{nonRedundant} = \text{false} \)
10. break
11. end if
12. end if
13. end for
14. if \( \text{nonRedundant} = \text{true} \)
15. insert \( \{ (g \rightarrow c), \text{supp} \} \) in MinMaxExact
16. end if
17. end if
18. end for
19. end for
20. return MinMaxExact

Algorithm 2: ReliableExactHR()

Input: C a set of frequent closed itemsets G; a set of minimal generators.

Output: A set of non-redundant multilevel rules.
1. \( \text{ReliableExact} := \emptyset \)
2. for all \( c \in C \)
3. for all \( g \in G_c \)
4. \( \text{nonRedundant} = \text{false} \)
5. if \( \forall c \in C \text{ such that } c' \subseteq c \text{ and } g' \in G_{c'} \), we have \((c \text{ or } c') \cup g' \subseteq g\)
6. then \( \text{nonRedundant} = \text{true} \)
7. else
8. \( \text{nonRedundant} = \text{false} \)
9. break
10. end if
11. for all \( g' \in G \)
12. if \( g' \neq g \)
13. if \( (g' \)is ancestor set of \( g \) and \((c'=c \text{ or } g'=g) \) and \((g' \)is not ancestor set of \( c' \text{ or } g') \) and \((g' \)is not descendant set of \( c' \text{ or } g') \)
14. then \( \text{nonRedundant} = \text{true} \)
15. break
16. end if
17. end if
18. end for
19. if \( \text{nonRedundant} = \text{true} \)
20. insert \( \{ (g \rightarrow c), \text{supp} \} \) in ReliableExact
21. end if
22. end for
23. end for
24. return ReliableExact

All Rights Reserved © 2012 IJARCSEE
C. Deriving Exact Rules from the Exact Basis Rules

The Min-MaxExact approach and ReliableExact approach have proven that they can deduce all of the exact rules from their basis set [17]. Comparing with the Min-MaxExact approach and ReliableExact approach, our work results in a smaller exact basis set by not only removing the redundant rules that are removed by the Min-MaxExact approach and ReliableExact approach, but also removing the hierarchically redundant rules. If we can recover all of the hierarchically redundant rules, then we can derive all the exact rules by using the Min-MaxExact or ReliableExact recovery algorithm. This will ensure that all the exact rules can still be derived and by achieving this, our approach will be a lossless representation of the exact Association Rules. The following algorithm is designed to recover the hierarchically redundant rules from the exact basis. By adding it to the algorithms used by Min-MaxExact and ReliableExact to derive the exact rules it is then able for the existing ReliableExact recovery algorithm to derive all of the exact rules. This is because our algorithm will give them a basis set that includes the hierarchically redundant rules (which the ReliableExact approach would not have removed in the first place). The basic idea is that, for each exact basis rule, first from generators to construct all possible exact basis rules whose antecedent is a descendant of the exact basis rule (steps 4 to 7 in Algorithm 3). These rules are potential exact basis rules that might have been eliminated due to the ancestor-descendant relationship. Then check to make sure these potential rules are valid (steps 8 to 12), finally, from the potential exact rules to find exact basis rules. These exact basis rules have been eliminated due to the ancestor-descendant relationship (steps 13 to 18).

Algorithm 3: DeriveExactHR()

Input: Set of exact basis rules denoted as Exact basis, set of frequent closed itemsets C and generators G.

Output: Set of rules that covers the exact basis and the hierarchically redundant rules.

1. Recovered := φ
2. ∀r ∈ Exact basis
3. CandidateBasis:= φ
4. for all generator g in G
5. if any of the item x in the antecedent X of rule r: X → Y is the ancestor of g.
6. then add all of the possible subsets of g into S
7. end for
8. for all s in S, check every , x ∈ X if x doesn’t have a descendant in s, add x to s to make s a descendant set of X
9. if s has no ancestors in Y and s has no descendants in Y and for all items i ∈ s there are no ancestor-descendant relations with item i ∈ s and for all item i ∈ Y there are no ancestor-descendant relation with item i ∈ Y
10. then insert s → Y in CandidateBasis
11. end if
12. end for
13. for all B → D ∈ CandidateBasis
14. if BUD= itemset i ∈ C and B = g ∈ G_i
15. insert (B → D, g.supp) in Recovered
16. end if
17. end for
18. end for
19. return Exactbasis U Recovered

V. EVALUATION

Experiments were conducted to test and evaluate the effectiveness of the proposed hierarchically non-redundant exact basis and to confirm that it is also a lossless basis set. This section presents and details the experiments and their results.

A. Datasets

We used 6 datasets to test our approach to discover whether it reduced the size of the exact basis rule set and to test that the basis set was lossless, meaning all the rules could be recovered. These datasets were composed of 10, 20, 50, 200 and 500 transactions and are named A to F respectively. The key statistics for these built datasets are detailed in Table 5.

Table 5 Obtained Frequent itemsets using Exact Basis

<table>
<thead>
<tr>
<th>Dataset</th>
<th>MME</th>
<th>MMEHR</th>
<th>RE</th>
<th>REHR</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>15</td>
<td>10</td>
<td>13</td>
<td>9</td>
</tr>
<tr>
<td>B</td>
<td>106</td>
<td>68</td>
<td>80</td>
<td>58</td>
</tr>
<tr>
<td>C</td>
<td>174</td>
<td>134</td>
<td>113</td>
<td>89</td>
</tr>
<tr>
<td>D</td>
<td>577</td>
<td>429</td>
<td>383</td>
<td>305</td>
</tr>
<tr>
<td>E</td>
<td>450</td>
<td>405</td>
<td>315</td>
<td>287</td>
</tr>
<tr>
<td>F</td>
<td>725</td>
<td>602</td>
<td>91</td>
<td>80</td>
</tr>
</tbody>
</table>

Fig. 2. Multi Level association Rules

VI. CONCLUSION

Redundancy in Association Rules affects the quality of the information presented and this affects and reduces the use of the rule set. The goal of redundancy elimination is to improve the quality, thus allowing them to better solve problems being faced. Our work aims to remove hierarchical redundancy in multi-level datasets, thus reducing the size of the rule set to improve the quality and usefulness, without causing the loss of any information. We have proposed an approach which removes hierarchical redundancy through the use of frequent closed itemsets and generators. This allows it to be added to other approaches which also remove redundant rules, thereby allowing a user to remove as much redundancy as possible. The next step in our work is to apply this approach to the approximate basis rule set to remove redundancy there. We will also review our work to see if there are other hierarchical
redundancies in the basis rule sets that should be removed and will investigate what should and can be done to further improve the quality of multi-level Association Rules.

REFERENCES


R. Vijaya Prakash is presently working as Associate Professor in Department of Computer science & Engineering, SR Engineering College., Pursuing Ph.D from Kakatiya University, Warangal. He completed his MCA from Kakatiya University and M.Tech (IT) form Punjabi University, Patiala.

Dr. A. Govardhan is working as professor in Department of Computer science & Engineering, JNT University, Hyderabad. He Received his B.Tech in 1992 from Osmania University, M.Tech in 1994 from Jawahar Lal Nehru University, Delhi and Ph.D from JNT University in 2003. He is very eminent person in the field of teaching he guided 4 Ph.D scholars.

Prof. SSVN. Sarma is an eminent professor in Mathematics. He Worked in Department of Mathematics and Department of Informatics, Kakatiya University over period of 30 years. He is well known person in the field of Mathematics and Computer Science. He guided many Ph.D Scholars.