AN IMPROVED FREQUENT PATTERN-GROWTH APPROACH TO DISCOVER RARE ASSOCIATION RULES

R. Uday Kiran and P. Krishna Reddy
International Institute of Information Technology-Hyderabad, Hyderabad, Andhra Pradesh, India

Keywords: Data mining, Rare knowledge patterns, Rare association rules, Frequent patterns.

Abstract: In this paper we have proposed an improved approach to extract rare association rules. The association rules which involve rare items are called rare association rules. Mining rare association rules is difficult with single minimum support (minsup) based approaches like Apriori and FP-growth as they suffer from "rare item problem" dilemma. At high minsup, frequent patterns involving rare items will be missed and at low minsup, the number of frequent patterns explodes. To address "rare item problem", efforts have been made in the literature by extending the "multiple minimum support" framework to both Apriori and FP-growth approaches. The approaches proposed by extending "multiple minimum support" framework to Apriori require multiple scans on the dataset and generate huge number of candidate patterns. The approach proposed by extending the "multiple minimum support" framework to FP-growth is relatively efficient than Apriori based approaches, but suffers from performance problems. In this paper, we have proposed an improved multiple minimum support based FP-growth approach by exploiting the notions such as "least minimum support" and "infrequent leaf node pruning". Experimental results on both synthetic and real world datasets show that the proposed approach improves the performance over existing approaches.

1 INTRODUCTION

Data mining represents techniques for discovering knowledge patterns hidden in large databases. Several data mining approaches are being used to extract interesting knowledge (G. Melli and Kitts, 2006). Like, association rule mining techniques (R. Agrawal and Swami, 1993) (Agrawal and Srikanth, 1994) discover association between the entities, clustering techniques (Xu, 2005) to group the unlabeled data into clusters such that there exists high inter similarity and low intra similarity between the clusters, classification techniques (Weiss and Kulikowski, 1991) to identify the different classes existing in categorical labeled data.

It can be observed that most of the data mining approaches discover the knowledge pertaining to frequently occurring entities. However, real-world datasets are mostly non-uniform in nature containing both frequent and relatively infrequent or rarely occurring entities. (Referring an entity as either frequent or rare is a subjective matter depending on the user and/or type of application etc.) In literature, it has been reported that rare knowledge patterns i.e., knowledge pertaining to rare entities may contain interesting knowledge useful in decision making process (Weiss, 2004) (B. Liu and Ma, 1999). The rare knowledge patterns are more difficult to detect because they present in fewer data cases. In the literature, research efforts are being made to investigate efficient approaches to extract rare knowledge patterns like rare association rules and rare class identification (Weiss, 2004).

In this paper, we have proposed an improved approach to extract rare association rules. Association rule mining (Agrawal and Srikanth, 1994) is a popular knowledge discovery technique and has been extensively studied in (J. Hipp and Nakhaeizadeh, 2000). The basic model of association rule mining is as follows. Let $I = \{i_1, i_2, ..., i_n\}$ be a set of items. Let T be a set of transactions (dataset), where each transaction t is a set of items such that $t \subseteq I$. A pattern (or an itemset) X is a set of items $\{i_1, i_2, ..., i_k\}$ ($1 \leq k \leq n$) such that $X \subseteq I$. Pattern containing k number of items is called k-pattern. An association rule is an implication of the form, $A \Rightarrow B$, where $A \subset I$, $B \subset I$ and $A \cap B = \emptyset$. The rule $A \Rightarrow B$ holds in T with *support s*, if s% of the transactions in T contain $A \cup B$. Similarly rule $A \Rightarrow B$ holds in T with *con-*

Uday Kiran R. and Krishna Reddy P. (2009).
AN IMPROVED FREQUENT PATTERN-GROWTH APPROACH TO DISCOVER RARE ASSOCIATION RULES.
In *Proceedings of the International Conference on Knowledge Discovery and Information Retrieval*, pages 43-52
DOI: 10.5220/0002299600430052

fidence c, if c% of transactions in T that support A also support B. Given T, the objective of association rule mining is to discover all association rules that have support and confidence greater than the user-specified minimum support (minsup) and minimum confidence (minconf). The patterns which satisfy the minsup value are called frequent patterns. The rules that satisfy the minsup value and minconf value are called strong rules.

Rare association rule refers to an association rule forming between frequent and rare items or among rare items. Rare associations may contain useful knowledge. For example, consider the set of items {bread, jam, bed, pillow} being sold in a super market. It can be observed that the items in the set {bread, jam} are frequently purchased items while the items in the set {bed, pillow} are infrequently or rarely purchased items. Even though {bed, pillow} contains rare items, it is interesting as it may generate more revenue in this case.

Mining rare association rules is an issue, because single minimum support (minsup) based approaches like Apriori (Agrawal and Srikanth, 1994) and FP-growth (H. Jiawei and Runying, 2004) suffer from "rare item problem" dilemma (Mannila, 1997). That is, at high minsup value, frequent patterns involving rare items could not be extracted as rare items fail to satisfy the minsup value. To facilitate participation of rare items in generating frequent patterns, the minsup value has to be set low. However, low minsup may result in combinatorial explosion of frequent patterns.

To address "rare item problem", efforts have been made in the literature by extending the "multiple minimum support" framework to both Apriori and FP-growth approaches. In this framework, each item is specified a minsup value called minimum item support (MIS) and frequent patterns are discovered if a pattern satisfies the lowest MIS value of an item in it.

The approaches (B. Liu and Ma, 1999) (Kiran and Reddy, 2009) proposed by extending "multiple minimum support" framework to Apriori require multiple scans on the dataset and generate huge number of candidate patterns.

In (Ya-Han Hu, 2004) an approach called Conditional Frequent Pattern-growth (CFP-growth) is proposed by extending the "multiple minimum support" framework to FP-growth. In this approach the MIS value of each item is used to construct MIS-tree instead of FP-tree. From the MIS-tree, compact MIS-tree is derived with the items having support values greater than the lowest MIS value among all items in transaction dataset (the details are discussed in Section 2).

The CFP-growth approach improves performance over Apriori based approaches. However, it suffers from performance problems. To generate frequent patterns involving rare items, it carries out computation involving those items which do not generate any frequent patterns. In this paper, we propose an improved CFP-growth approach referred as Improved Conditional Frequent Pattern-growth (ICFP-growth) approach by exploiting the notions such as "least minimum support" and "infrequent leaf node pruning". The notion "least minimum support" is used to consider only those items which generate frequent patterns and the notion "infrequent leaf node pruning" is used to prune the leaf nodes belonging to infrequent items, so that the size of compact MIS-tree can be reduced. Experimental results on both synthetic and real world datasets show that the proposed approach improves the performance over CFP-growth.

The paper is organized as follows. In Section 2, we discuss the mining of frequent patterns using CFP-growth approach. In Section 3, we present the proposed ICFP-growth approach and the algorithm to mine frequent patterns. In Section 4, we present the experiment results conducted on synthetic and real world datasets. In the last section we discuss conclusions and future work.

2 CFP-GROWTH APPROACH

We first define the terms "minimum item support" and "sorted closure property". Next, we briefly explain the CFP-growth approach along with the performance problem.

Minimum Item Support. In multiple minimum support based frequent pattern mining, each item is specified with a minsup value called minimum item support (MIS). Frequent items are the items having support greater than or equal to their respective MIS values. Infrequent items are the items having support less than their respective MIS values.

Frequent pattern is a pattern that satisfies the lowest MIS value of the item in it. (See Equation 1.)

$$S(i_1, i_2, ..., i_k) \geq min \begin{pmatrix} MIS(i_1), MIS(i_2) \\ ..., MIS(i_k) \end{pmatrix} \quad (1)$$

where S(i_1, i_2, $\cdots$, i_k) represents the support for an itemset $\{i_1, i_2, \cdots, i_k\}$ and MIS(i_j) represents the minimum item support for item i_j.

Sorted Closure Property. The frequent patterns discovered using multiple minsup values follow "sorted closure property". The "sorted closure property" says, if a *sorted k-pattern* $\langle i_1, i_2, ..., i_k \rangle$, for k $\geq$

2 and MIS(i_1) $\geq$ MIS(i_2) $\geq$ ... $\geq$ MIS(i_k), is frequent, then all of its subsets involving item having lowest MIS value (i_1) need to be frequent and the subsets involving rest of the items i_j, where j $\geq$ 2, need not necessarily be frequent patterns. So the multiple minsup based frequent pattern mining algorithms have to consider both frequent and infrequent items (complete set of items I) to generate further frequent patterns.

For example, consider a transaction dataset involving three items i_1, i_2 and i_3, each having MIS values 5%, 10% and 20% respectively. If a sorted 3-pattern $\{i_1, i_2, i_3\}$ has support 6% then it is a frequent pattern. In this frequent pattern, the supersets of item i_1 i.e., $\{\{i_1\}, \{i_1, i_2\}, \{i_1, i_3\}\}$ are to be frequent. However, the supersets of items i_2 and i_3, say $\{i_2, i_3\}$ may still be infrequent by having support as 8%. For this pattern to be frequent, the support it should have is 10% (min(10%, 20%)).

The **CFP-growth** approach extends "pattern-growth" methodology to multiple minimum support values.

In this approach, it is assumed that the information regarding the MIS values for the items will be the provided by the user prior to its execution. The MIS-tree is constructed as follows. First, the items are sorted in descending order of their MIS values, say L_1 and their frequency values are set at zero. Next, a root node of the tree is constructed by labeling with "null". Next, for each transaction in the dataset the following steps are performed to generate MIS-tree. They are:

1. The items in each transaction are sorted in L_1 order. Next, update the frequencies of the items which are present in the transaction by incrementing the frequency value of the respective item by 1.
2. A branch is created for each transaction such that nodes represent the items, level of nodes in a branch is based on the sorted order and the count of each node is set to 1. However, in constructing the new branch for a transaction, the count of each node along a common prefix is incremented by 1, and nodes for the items following the prefix are created, linked accordingly and their values are set to 1.

To facilitate tree traversal, an item header table is built so that each item points to its occurrences in the tree via a chain of node-links. From the item frequencies, the respective support values are calculated. Using the lowest MIS value among all the items (MIS), the tree-pruning process is performed on the item header and MIS-tree to remove the items having support less than the lowest MIS value among all items. After tree-pruning, tree-merging process is performed to generate the compact MIS-tree.

Table 1: Transaction dataset.

TID	Items
1	bread, jam
2	bread, jam, ball
3	bread, jam, pen
4	bread, jam, pencils
5	bread, bat, ball
6	bed, pillow
7	bed, pillow
8	ball, bat
9	ball, bat
10	ball, bat

The compact MIS-tree is mined as follows. Each item in L_1 is considered as a suffix pattern, next its conditional pattern base, which is a set of prefix-paths in the MIS-tree is constructed and mining is performed recursively on such a tree. The pattern growth is achieved by the concatenation of the suffix pattern with the frequent patterns generated from the conditional pattern.

For the dataset shown in Table 1, the extraction of frequent patterns using CFP-growth algorithm is illustrated using Example 1. For ease of explaining this example we refer the support and MIS values of the items in terms of support counts and MIS counts.

Example 1. For the transaction dataset shown in Table 1, the itemset I = {bread, ball, jam, bat, pillow, bed, pencil, pen}. Let the MIS values (in count) for bread, ball, jam, bat, pillow, bed, pencil and pen be 4, 4, 3, 3, 2, 2, 2 and 2 respectively. Now, using the MIS values for the items, the CFP-growth approach sorts the items in descending order of their MIS values and assigns the frequency value of zero to every item. Thus, L_1 contain {{bread:0}, {ball:0}, {jam:0}, {bat:0}, {pillow:0}, {bed:0}, {pencil:0}, {pen:0}}. In the first scan of the dataset shown in Table 1, the first transaction "1: bread, jam" containing two items is scanned in L_1 order i.e., {bread, jam} and the frequencies of items "bread" and "jam" are updated by 1 in L_1. Next, a first branch of tree is constructed with two nodes, $\langle$bread: 1$\rangle$ and $\langle$jam: 1$\rangle$, where "bread" is linked as a child of the root and "jam" is linked as a child of "bread". The second transaction "2: bread, jam, ball" containing three items "bread, ball, jam" in L_1 order and the frequencies of the items are updated by incrementing by 1. Next, the items in second transaction, ordered in L_1, will result in a branch where "bread" is linked to root, "ball" is linked to "bread" and "jam" is linked to "ball". However, this branch shares the common prefix, "bread", with the existing path for first transaction. Therefore, the

count of “bread” node is incremented by 1 and new nodes are created for items “ball and jam” such that “ball” is linked to “bread” and “jam” is linked to “ball” and their node values are set to 1. The process is repeated until all the transactions are completed. A node link table is built for traversal. The constructed MIS-tree is shown in Figure 1. Next, CFP-growth identifies the lowest MIS value among all the items i.e., 2 and try to remove the items whose support value is less than 2. From the node table, first, item “pencil” is removed. Next, tree pruning is preformed on MIS-tree to remove all the nodes pertaining to item “pencil”. Next, the item “pen” is removed from the node-link table and MIS-tree. After that, tree-merging is performed to merge the branches. The final and compact MIS-tree is shown in Figure 1 with bold letter and think lines.

Mining the constructed MIS-tree is shown in Table 2 and is summarized as follows. Start from the last item in the item header table i.e., “bed”, as the suffix item. Then construct its conditional pattern base for this suffix item is as follows. In the MIS-tree shown in Figure 1, “bed” occurs in one node. The path formed from this node is ⟨pillow, bed: 2⟩. Therefore, considering “bed” as suffix, its corresponding prefix paths are ⟨pillow: 2⟩, which form its conditional pattern base. Its conditional MIS-tree contains only a single path, ⟨pillow: 2⟩. The single path generates all the combinations of frequent patterns {pillow, bed: 2}. Next, by considering every item one after another in the ascending order of their MIS values, the corresponding conditional pattern bases and conditional MIS-tree are constructed to generate all frequent patterns are generated. The finally generated frequent patterns are {{bread}, {ball}, {jam}, {bat}, {pillow}, {bed}, {bread, jam}, {bat, ball}, {pillow, bed}}.

The **performance problem in the CFP-growth** approach is as follows. The CFP-growth constructs compact MIS-tree involving the items having support greater than the lowest MIS value among all the items. The intuition behind this selection process is that no frequent pattern will have support less than the lowest MIS value among all the items. However, by considering the lowest MIS value among all items this approach considers certain infrequent items which will never generate any frequent patterns. This results in increased memory and runtime requirements. We illustrate this scenario in Example 2.

Example 2. Let the set of items {U, V, W, X, Y, Z} have support values {10%, 8%, 6%, 4%, 2%, 1%} respectively. Let the respective MIS values be {9%, 9%, 7%, 4%, 3%, 2%}. Since the lowest MIS value is 2%, any frequent pattern will have support not less than 2%. Therefore, CFP-growth considers set of items {U, V, W, X, Y} for generating frequent patterns. The CFP-growth do not consider item Z for generating frequent patterns because its support (1%) is less than lowest MIS value among all items (2%). However, it can be observed that the infrequent item Y will never generate any frequent pattern. The reason is that any pattern involving item Y can have support at most equivalent to 2%, which is less than the MIS value of Y i.e., 3%.

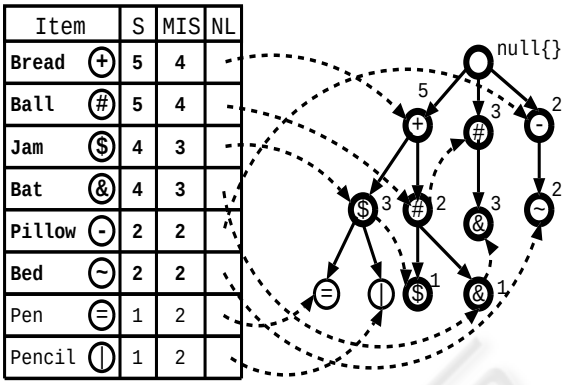

Figure 1: MIS-tree. The compact MIS-tree is represented with bold letters (first six rows in the table) and the corresponding think lines and circles in the tree.

3 PROPOSED APPROACH

In this section, we first present the basic idea of the proposed approach. Next, the algorithm is discussed. Subsequently, we discuss how the proposed approach is different from CFP-growth approach.

3.1 Basic Idea

In the proposed approach, we exploit the following notions: “least minimum support” and “infrequent leaf node pruning”.

Least Minimum Support. The frequent patterns mined using multiple minsup values follow “sorted closure property”. According to “sorted closure property”, all the supersets involving the item having lowest MIS value should be frequent in a frequent pattern. So in every frequent pattern, frequent item represents the item having the lowest MIS value. Therefore, it can be argued that every frequent pattern will have support greater than or equal to lowest MIS value among all the frequent items. Thus, if we remove

Table 2: Mining the MIS-tree by creating conditional pattern bases in CFP-growth.

Item	MIS	Conditional pattern base	Conditional MIS-tree	Frequent patterns generated
bed	2	{{pillow: 2}}	⟨pillow: 2⟩	{pillow, bed: 2}
pillow	2			
bat	3	{{bread, ball: 1}, {ball:3}}	⟨ball:4⟩	{ball, bat: 4}
jam	3	{{bread,ball: 1}, {bread: 3}}	⟨bread:4⟩	{bread, jam: 4}
ball	4	{{bread: 2}}	⟨bread:2⟩	{bread, ball: 2}

all the items whose support is less than the lowest MIS value of the frequent item, no frequent pattern will be missed. This notion is called "least minimum support" (LMS) and it refers to the lowest MIS value among all the frequent items. The significance of this notion is illustrated in Example 3.

Example 3. Continuing with the Example 2, it can be observed that the set of items {U, X} are frequent items. The lowest MIS value among these items is 4%. Therefore, using LMS value as 4%, the proposed approach prunes the set of items {Y, Z} and considers {U, V, W, X} for frequent pattern mining.

Let I be the set of all items in the transaction dataset. Let C be the set of items considered by CFP-growth approach for mining frequent patterns. Let F be the set of items considered by ICFP-growth approach for mining frequent patterns. Then, the relation between I, C and F is as follows: $F \subseteq C \subseteq I$.

Infrequent Leaf Node Pruning: In the process of mining compact MIS-tree using conditional pattern bases, the suffix item (or pattern) represents the item having lowest MIS value. So if a suffix item is an infrequent item, based on "sorted closure property", it can be said that all the prefix paths of the respective suffix item will also be infrequent. Therefore, the ICFP-growth approach skips the construction of conditional pattern bases for the infrequent suffix items. In the compact MIS-tree, the leaf nodes belong to infrequent items have no significance because its prefix paths (conditional pattern bases) are not used. The resultant MIS-tree will still preserve the transaction details pertaining to frequent patterns even if we prune the leaf nodes belonging to infrequent items. Therefore, we propose that "infrequent leaf node pruning" is performed such that every branch ends with the node of a frequent item. We illustrate the "infrequent leaf node pruning" in Example 4.

Example 4. Continuing with the Example 2 and Example 3, let the MIS-tree derived after performing a single scan on the dataset contain three branches, say ⟨U, V⟩, ⟨V, W⟩ and ⟨W, X⟩. Among the set of items {U, V, W, X}, we know that items V and W are infrequent items. First, let us consider the item W, having relatively lowest MIS value for pruning. In the MIS-tree generated, the branch ⟨U, V⟩ do not contain any item W and hence no pruning is performed. In the second branch ⟨V, W⟩, there exists leaf node with item W. Therefore, pruning is performed to generate a new branch ⟨V⟩. In the third branch ⟨W, X⟩, though there exists node of the item W, it is not the leaf node. So no pruning is performed. Thus the resulted MIS-tree contains ⟨U, V⟩, ⟨V⟩ and ⟨W, X⟩. Next, select another infrequent item i.e., V for pruning. In the newly generated MIS-tree, the branch ⟨U, V⟩ contains the leaf node V. So pruning is performed to create a new branch ⟨U⟩. In the second branch ⟨V⟩, the node V is a leaf node. So the node is pruned from compact MIS-tree. Since no node exists in the branch the branch is deleted. In the third branch, ⟨W⟩, exists no node with the item V. So no pruning is performed. Thus the final resulted MIS-tree contains only two branches ⟨U⟩ and ⟨W, X⟩.

3.2 The Algorithm

The ICFP-growth pre-assumes that for every item, user specifies the MIS values priori to its execution. Therefore, using the priori information i.e., MIS values of the items, the frequent patterns are generated with a single scan on the dataset. The proposed algorithm generalizes the CFP-growth algorithm for finding frequent patterns. This approach involves three steps. They are constructing the MIS-tree, extracting compact MIS-tree and mining the compact MIS-tree to mine frequent patterns. We now discuss each of these steps in detail.

3.2.1 Constructing MIS-tree

The construction of MIS-tree in ICFP-growth algorithm is shown in Algorithm 1 and described as fol-

lows. The ICFP-growth algorithm accepts transaction dataset (Trans), Itemset (I) and minimum item support values (MIS) of the items as input parameters. Using the input parameters, the ICFP-growth creates an initial MIS-tree which is similar to MIS-tree created by CFP-growth (Lines 1 to 7 in Algorithm 1). (Refer to Section 2 for knowing the construction of initial MIS-tree using CFP-growth approach.)

3.2.2 Extracting Compact MIS-tree

Next, starting from the last item in the item-header table (i.e., item having lowest MIS value) perform tree-pruning operating by calling MisPruning procedure (See, procedure 3) to remove the infrequent items from the item-header table and MIS-tree. After one item is pruned, move to immediate next item in item-header table and perform tree-pruning. However, stop tree-pruning process when the frequent item is encountered. The MIS value of this frequent item represents the LMS value. Let the resultant item header table be MinFrequentItemHeaderTable. The items in this header table may contain both frequent and infrequent items having support greater than the lowest MIS value among all frequent items. The items in the header table are referred as "quasi-frequent items". Call MisMerge procedure (See, procedure 4) to merge the tree. Finally, call InfrequentLeafNodePruning procedure (See, procedure 5) to prune the infrequent leaf nodes in the MIS-tree. The resultant MIS-tree is the compact MIS-tree.

3.2.3 Mining Frequent Patterns from Compact MIS-tree

Mining the frequent patterns from the compact MIS-tree is shown in Algorithm 6. The process of mining the compact MIS-tree in ICFP-growth is almost same as mining the compact MIS-tree in CFP-growth. However, the variant between the two approaches is that before generating conditional pattern base and conditional MIS-tree for every item in the header of the Tree, the ICFP-growth approach verifies whether the suffix item in the header of the Tree is a frequent item (Line 2 in Algorithm 6). If an suffix item is not a frequent item (or pattern) then the construction of conditional pattern base and conditional MIS-tree are skipped. The reason is as follows. In every frequent pattern, the item having lowest MIS value should be a frequent item (sorted closure property). In constructing the conditional pattern base for a suffix item, the suffix item represents the item having lowest MIS value. Therefore, if the suffix item is an infrequent item then all its prefix-paths (the patterns in which it

Algorithm 1. MIS-tree (Tran:transaction dataset, I: itemset containing *n* items, MIS: minimum item support values for *n* items).

1: Let L represent the set of items sorted in nondecreasing order of their MIS values.
2: create the root of a MIS-tree, T, and label it as "null".
3: **for** each transaction t $\in$ Tran **do**
4: sort all the items in t in L order.
5: count the support values of any item i, denoted as S(i) in t.
6: Let the sorted items in t be [p|P], where p is the first element and P is the remaining list. Call InsertTree([p|P], T).
7: **end for**
8: **for** j=n-1; j $\geq$ 0; –j **do**
9: **if** S[i_j] $<$ MIS[i_j] **then**
10: Delete the item i_j in header table.
11: call MisPruning(Tree, L[i_j]).
12: **else**
13: break; //come out of pruning step.
14: **end if**
15: **end for**
16: Name the resulting table as MinFrequentItemHeaderTable.
17: Call MisMerge(Tree).
18: Call InfrequentLeafNodePruning(Tree).

Procedure 2. InsertTree ([p|P, T).

1: **while** P is nonempty **do**
2: **if** T has a child N such that p.item-name=N.item-name **then**
3: N.count++.
4: **else**
5: create a new node N, and let its count be 1.
6: let its parent link be linked to T.
7: let its node-link be linked to the nodes with the same item-name via the node-link structure;
8: **end if**
9: **end while**

Procedure 3. MisPruning (Tree, i_j).

1: **for** each node in the node-link of i_j in Tree **do**
2: **if** the node is a leaf **then**
3: remove the node directly;
4: **else**
5: remove the node and then its parent node will be linked to its child node(s);
6: **end if**
7: **end for**

Procedure 4. MisMerge (Tree).

1: **for** each item i_j in the MinFrequentItemHeaderTable **do**
2: **if** there are child nodes with the same itemname then **then**
3: merge these nodes and set the count as the summation of these nodes' counts.
4: **end if**
5: **end for**

Procedure 5. InfrequentLeafNodePruning(Tree).

1: choose the last but one item i_j in MinFrequentItemHeaderTable. That is, item having second lowest MIS value.
2: **repeat**
3: **if** i_j item is infrequent item **then**
4: using node-links parse the branches of the Tree.
5: **repeat**
6: **if** i_j node is the leaf of a branch **then**
7: drop the node-link connecting through the child branch.
8: create a new node-link from the node in the previous branch to node in the coming branch.
9: drop the leaf node in the branch.
10: **end if**
11: **until** all the branches in the tree are parsed
12: **end if**
13: choose item i_j which is next in the order.
14: **until** all items in MinFrequentItemHeaderTable are completed

represents the item having lowest MIS value) will also be infrequent.

After mining frequent patterns, the method give in (Agrawal and Srikanth, 1994) can be used to find association rules.

4 EXPERIMENTAL RESULTS

4.1 Experimental Details

In this section, we present the performance comparison of CFP-growth and ICFP-growth approaches. We are not comparing the ICFP-growth with MSApriori and IMSApriori approaches. The reason being that, CFP-growth is relatively efficient than MSApriori approach (Ya-Han Hu, 2004). The IMSApriori approach uses "iterative level-wise search" to discover

Algorithm 6. ICFP-growth (Tree: MIS-tree, L: set of quasi-frequent items, MIS: minimum item support values for the items in L).

1: **for** each item i_j in the header of the Tree **do**
2: **if** i_j is a frequent item **then**
3: generate pattern $\beta = i_j \cup \alpha$ with support = i_j.support;
4: construct β's conditional pattern base and β's conditional MIS-tree Tree β.
5: **if** Tree $\beta \neq \emptyset$ **then**
6: call CpGrowth(Tree β, β, MIS(i_j).
7: **end if**
8: **end if**
9: **end for**

Procedure 7. CpGrowth(Tree, α, MIS(α)).

1: **for** each i_j in the header of Tree **do**
2: generate pattern $\beta = i_j \cup \alpha$ with support = i_j.support.
3: construct β's conditional pattern base and then β's conditional MIS-tree $Tree_\beta$.
4: **if** $Tree_\beta \neq \emptyset$ **then**
5: call CpGrowth($Tree_\beta$, β, MIS(α)).
6: **end if**
7: **end for**

frequent patterns. So, the CFP-growth is relatively efficient than IMSApriori approach.

We have evaluated the performance of proposed approach by considering two kinds of datasets: synthetic and real world datasets. The synthetic dataset T10.I4.D100K, is generated with the data generator (Agrawal and Srikanth, 1994), which is widely used for evaluating association rule mining algorithms. It contains 1,00,000 number of transactions, 886 items, maximum number of items in a transaction is 29 and the average number of items in a transaction is 12. Another dataset is a real world dataset referred as retail dataset. It contains 88,162 number of transactions, 16,470 items, maximum number of items in a transaction is 76 and the average number of items in each transaction is 5.8.

4.2 Experiment 1

In this experiment, we present the results pertaining to construction of compact MIS-tree by only exploiting the "least minimum support" notion and not considering the "infrequent leaf node pruning" notion. In the next subsection, we discuss the results by exploiting both notions.

For this experiment, we need a method to assign

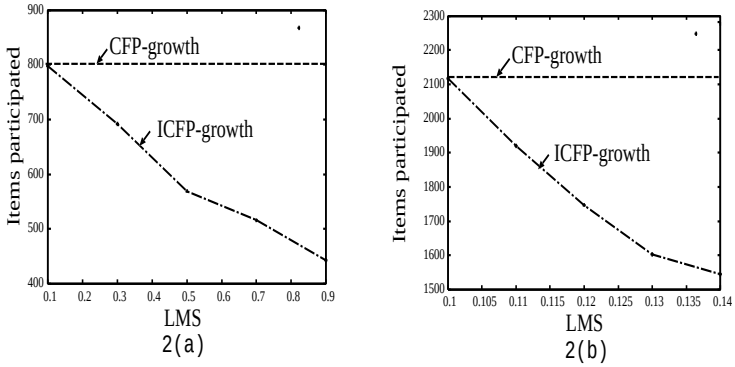

Figure 2: Number of items participated at different LMS values in (a) Synthetic and (b) Retail datasets.

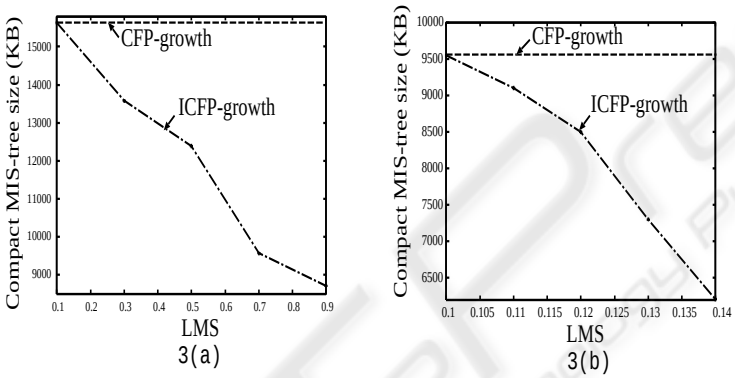

Figure 3: The size of the compact MIS-tree at different LMS values in (a) Synthetic and (b) Retail datasets.

MIS values to items in the dataset. We use the supports of the items in the dataset as the basis for assigning MIS values. Specially, we use the following formulas:

$$MIS(i_j) = \left\{ \begin{array}{cc} M(i_j) & if\ M(i_j) > LMS \\ LMS & else\ if\ M(i_j) < LMS \\ & and\ S(i_j) > LMS \\ LMIS & else \end{array} \right\} \quad (2)$$

$$M(i_j) = S(i_j) - SD$$

where, SD is a user-specified support difference value varied between 0% to 1%, $S(i_j)$ refers to support of an item equal to f(i_j)/N, (f(i_j) represents frequency of i_j and N is the number of transactions in a transaction dataset), LMS corresponds to user-specified lowest minimum support value, which represents lowest MIS value of a frequent item and LMIS corresponds to user-specified least minimum item support value, which represents the lowest MIS value among all items in the transaction dataset. The LMS value will be always be greater than or equal to LMIS value.

For both T10.I4.D100k and Retail datasets, the SD and LMIS values are set at 0.1% and 0.1% respectively. By varying the LMS values, the compact MIS-tree is constructed using the proposed approach. It can be observed that in the CFP-growth approach there is no scope to vary LMS values. (In order to make the chosen MIS value to represent the LMS value, the items which are having support values less than the LMS value are converted into infrequent items.)

Both Figure 2(a) and Figure 2(b) show the number of items participated in generating frequent patterns at different LMS values in both T10.I4.D100k and Retail datasets. It can be observed that as the LMS value increases the number of items which have participated in generating the frequent patterns also got reduced in ICFP-growth approach. However, in CFP-growth the number of items participating in generating frequent patterns remains the same.

Both Figure 3(a) and Figure 3(b) provide the information regarding the size of compact MIS-tree generated at different LMS values in both datasets. It can be observed that as the LMS value increases the size of the compact MIS-tree gets reduced in ICFP-growth approach. The reason being that the number of items participating in generating frequent patterns also get reduced. However, the size of the compact MIS-tree generated by CFP-growth do not get

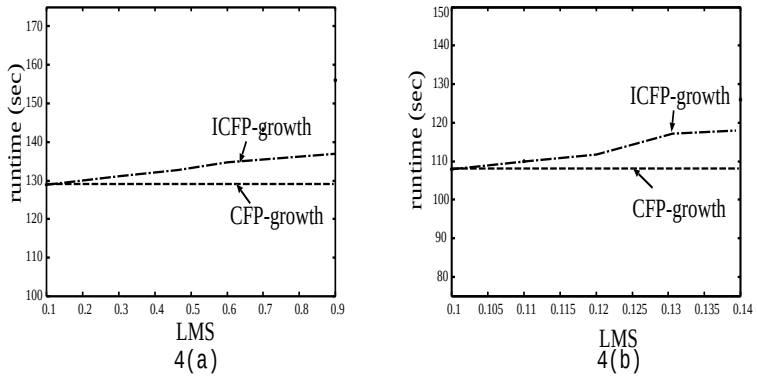

Figure 4: Runtime for generating compact MIS-tree at different LMS values in (a) Synthetic and (b) Retail datasets.

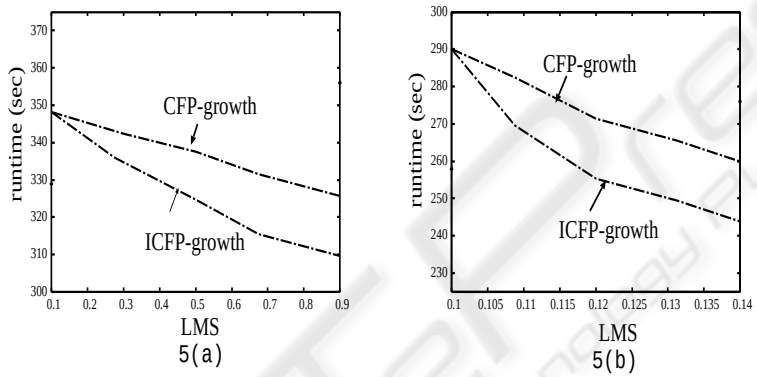

Figure 5: Runtime for generating compact MIS-tree and frequent patterns at different LMS values in (a) Synthetic and (b) Retail datasets.

reduced. The reason is that CFP-growth fails to prune the items which will not generate any frequent patterns.

Both Figure 4(a) and Figure 4(b) provide the information regarding the "runtime" taken to generate compact MIS-tree at different LMS values in both datasets. It can be observed that the ICFP-growth requires relatively more runtime than CFP-growth approach. The reason is ICFP-growth approach has to prune relatively more number of items from the header table and MIS-tree as compared with CFP-growth.

Both Figure 5(a) and Figure 5(b) provide the information regarding the "runtime" taken to construct compact MIS-tree and generate frequent patterns at different LMS values in both datasets. In this experiment the ICFP-growth takes relatively less runtime than CFP-growth approach. It can be noted that even though ICFP-growth takes more time for constructing compact MIS-tree over CFP-growth approach, overall it takes less time to extract frequent patterns. The reason is the ICFP-growth approach skips construction of conditional pattern bases for the suffix items (or patterns) which are infrequent.

4.3 Experiment 2

In this experiment, we present the results by exploiting both "least minimum support" and "infrequent leaf node pruning" notions.

The experiment is conducted as follows. In this experiment, the MIS value for the each item is fixed equal to a random number between 0.1% to 1% of the number of transactions in the dataset. We randomly select some percentage of items which vary from 0% to 20% and made them infrequent by setting MIS values greater than their support values. By fixing LMS values at 0.1% and 0.25%, we compute the size of compact MIS-tree by varying percentage of infrequent items under both the approaches. The corresponding results are shown in Figure 6(a) and Figure 6(b) for the datasets T10.I4.D100k and Retail datasets respectively. The results show that the size of the compact MIS-tree remains the same at different percentage values of infrequent items under CFP-growth approach, whereas the size of the compact MIS-tree reduces significantly at different percentage values of infrequent items under ICFP-growth approach. The reason is the ICFP-growth approach pruned the leaf

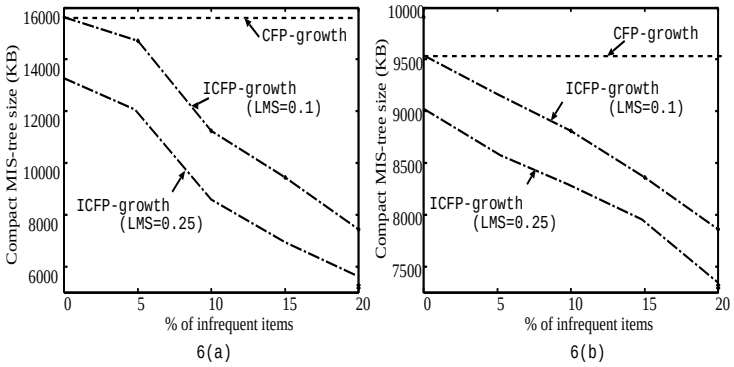

Figure 6: Compact MIS-tree sizes at different percentage of infrequent items in (a) Synthetic and (b) Retail datasets.

nodes belonging to infrequent items.

5 CONCLUSIONS AND FUTURE WORK

To extract rare association rules, efforts are being made in the literature by extending the "multiple minimum support" framework to FP-growth approach. In this paper, we have proposed an improved FP-growth approach with "multiple minimum support" framework by exploiting the notions such as "least minimum support" and "infrequent leaf node pruning". The proposed approach reduces the memory for constructing conditional frequent pattern tree and runtime for generating frequent patterns. The experimental results on both synthetic and real world datasets show that the proposed approach improves the performance over the existing approaches.

As a part of future work, we are going to investigate appropriate methodology for assigning confidence values in a dynamic manner to generate rare association rules. We are also going to investigate appropriate methodology for automatic calculation of MIS values for the items.

ACKNOWLEDGEMENTS

The work has been carried out with the support from Nokia Global University Grant.

REFERENCES

Agrawal, R. and Srikanth, R. (1994). Fast algorithms for mining association rules. In *International Conference on Very Large Databases*.

B. Liu, W. H. and Ma, Y. (1999). Mining association rules with multiple minimum supports. In *ACM Special Interest Group on Knowledge Discovery and Data Mining Explorations*.

G. Melli, R. Z. O. and Kitts, B. (2006). Introduction to the special issues on successful real-world data mining applications. In *ACM Special Interest Group on Knowledge Discovery and Data Mining Explorations*, volume 8, Issue 1.

H. Jiawei, P. Jian, Y. Y. and Runying, M. (2004). Mining frequent patterns without candidate generation: A frequent-pattern tree approach. In *ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*.

J. Hipp, U. G. and Nakhaeizadeh, G. (2000). Algorithms for association rule mining a general survey and comparision. In *ACM Special Interest Group on Knowledge Discovery and Data Mining, Volume 2, Issue 1*.

Kiran, R. U. and Reddy, P. K. (2009). An improved multiple minimum support based approach to mine rare association rules. In *IEEE Symposium on Computational Intelligence and Data Mining*.

Mannila, H. (1997). Methods and problems in data mining. In *International Conference on Database Theory*.

R. Agrawal, T. I. and Swami, A. (1993). Mining association rules between sets of items in large databases. In *ACM Special Interest Group on Management Of Data*.

Weiss, G. M. (2004). Mining with rarity: A unifying framework. In *ACM Special Interest Group on Knowledge Discovery and Data Mining Explorations*.

Weiss, S. and Kulikowski, C. A. (1991). Computer systems that learn: Classification and prediction models from statistics. In *Neural Nets, Machine Learning, and Expert Systems*. Morgan Kaufmann.

Xu, R. (2005). Survey of clustering algorithms. In *IEEE Transactions on Neural Networks*.

Ya-Han Hu, Y.-L. C. (2004). Mining association rules with multiple minimum supports: A new algorithm and a support tuning mechanism. In *Decision Support Systems*.