Rare Association Rule Mining using Improved FP-Growth algorithm

T. Ravi Kumar [1]
[1]*Associate Professor, Department of CSE*
Swarna Bharathi Institute of Science & Technology, Khammam,
[1]trkumar5@gmail.com

K. Raghava Rao[2]
[2]*Professor in CSE, School of Computing, KL University, Vaddeswaram, Guntur, A.P. INDIA*[12]
[2]raghavarao1@yahoo.com

Abstract*: Rare association rule refers to an association rule forming between frequent and rare items or among rare items. CFP-growth approach is used to mine frequent patterns using multiple minimum support (minsup) values. This approach is an extension of FP-growth approach to multiple minsup values. This approach involves construction of MIS-tree and generating frequent patterns from the MIS-tree. The issue in CFP-growth is constructing the compact MIS-tree because CFP-growth considers certain items, which will generate neither frequent patterns nor rules. In this paper, we propose an efficient approach for constructing the compact MIS-tree. To do so, the proposed approach explores the notions "least minimum support" and "infrequent child node pruning. The proposed approach improves the performance over CFP-growth approach.*

Keywords: Data mining, Association rules, IMSApriori, CFP-growth and ICFP-Growth

1.Introduction

Data mining represents techniques for extracting patterns hidden in large datasets by combining methods from statistics and artificial intelligence with database management. In Data mining approaches mainly focuseson discovering the frequent occurring items and the knowledge pertaining to it. However, real-world datasets are mostly non-uniform in nature containing both frequent and relatively infrequent or rarely occurring entities. But rare knowledge patterns are more difficult to detect because they present in fewer data cases.

In this paper, we are proposing an improved approach to extract rare association rules. Rare association rule refers to an association rule forming between frequent and rare items or among rare items. Association rule generation is usually split up into two separate steps:

1. First, minimum support is applied to find all frequent item sets in a database.
2. Second, these frequent item sets and the minimum confidence constraint are used to form rules.

Minsup controls the minimum number of data cases that each set of items must cover. Minconf controls the predictive strength of a rule. The main issue in mining rare association rules is finding the frequent patterns involving both frequent and rare items. And the later are not found in the algorithms like Apriori and FP-Growth as they suffer with "rare item problem". That is, at high minsup value, frequent patterns involving rare items could not be extracted as rare items fail to satisfy the minsup value. Therefore in order to find the rare items the minsup value would be set to low which also results in producing many frequent items.

1.1System model and notations

The problem of association rule mining would be defined as: Let $I = \{i_1, i_2, \ldots, i_n\}$ be a set of n binary attributes called items. Let $D = \{t_1, t_2, \ldots, t_m\}$ be a set of transactions called the database. Each transaction in D has a unique transaction ID and contains a subset of the items in I. A rule is defined as an implication of the form $X \Rightarrow Y$ where $X, Y \subseteq I$ and $X \cap Y = \emptyset$. The sets of items X and Y are called antecedent and consequent of the rule respectively. The rule X ! Y holds in D with support s, if s% of the transactions in D contain X ! Y. Similarly rule X ! Y holds in D with confidence c, if c% of transactions in D that support A also support B.

2.MULTIPLE MINIMUM SUPPORT APPROACHES

2.1 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.

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

MIS for each item is calculated as follows

$$MIS(i_j) = \beta S(i_j)), \; if \; \beta S(i_j) > LS$$
$$= LS \quad else$$

Where SD refers to support difference

SD = ! (1" #)

Where ! represents the parameter like mean, median, mode, maximum support of the item supports and # is the parameter ranging between 0 to 1. SD takes values from (0, !).

2.2 CFP-Growth approach

In order to address the performance problems involved in MSApriori approach, an efficient approach known as CFP-growth. The CFP-growth is an extension of single minsup based FP-growth approach to multiple minsup values. This approach involves two steps. They are construction of MIS-tree and mining frequent patterns from the MIS-tree using conditional pattern bases. This approach assumes that information regarding the MIS values for the items will be the provided by the user priori to its execution., the MIS-tree is constructed as follows.

1. First, the items are sorted in descending order of their MIS values, say L1 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:

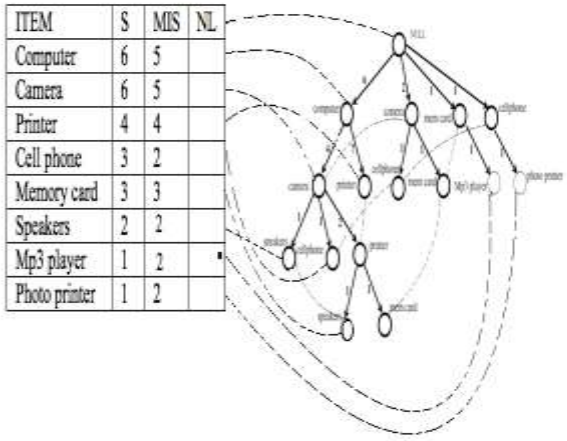
2. A branch is created for each transaction such that nodes represent the items, level of nodes in a branch is based on the sorted order and the count

TID	Items
1	Computer, camera, speakers
2	Camera, cell phone
3	Camera, memory card, printer
4	Computer, camera, cell phone
5	Computer, printer
6	Camera, memory card
7	Computer, printer
8	Computer, camera, printer, speakers
9	Computer,printer, camera, memory card
10	Memory card, mp3 player
11	Cell phone, photo printer

of each node is set to 1. However, in constructing the new branch for a transaction, the count of1. The items in each transaction are sorted in L1 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.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 is less than the lowest MIS value. After tree pruning, tree-merging process is

ITEM	S	MIS	NL
Computer	6	5	
Camera	6	5	
Printer	4	4	
Cell phone	3	2	
Memory card	3	3	
Speakers	2	2	
Mp3 player	1	2	
Photo printer	1	2	



Mining the constructed MIS-tree is shown in Table2.

In multiple minsup based frequent pattern mining infrequent items can still generate frequent patterns. For a pattern to be frequent, it should satisfy the lowest MIS value among the items in a pattern. So, given the MIS values of every item, any frequent pattern will have support greater than or equivalent to the lowest MIS value among all the items. Therefore, if an infrequent item has support less than the lowest MIS value, it will not generate any frequent pattern and hence can be pruned. This notion is exploited by CFP-growth approach to construct the efficient MIS-tree. We illustrate this scenario in Example 1

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
Mem card	3	{{camera:1,printer:1,computer:1}, {camera:1}}	⟨camera: 2⟩	{camera, mem card: 2}
Cell phone	2	{{camera:1,computer:1}, {camera}}	⟨camera: 2⟩	{Cell phone, camera:2}
Printer	4	{{Computer:2},{printer:1}, {computer:2,camera:2}}	⟨computer: 4⟩	{Printer, computer: 4}
Camera	5	{{Computer:4}	⟨computer: 4⟩	{Computer, camera: 4}

Example1: Let the set of items {i,j,k,l,m,n,o,p} have support values {10%,8%,6%, 6%,4%,3%, 2%, 1%} respectively. Let the respective MIS values be {9%, 9%, 7%,7%, 4%, 3%, 3%,2%}. Since the lowest MIS value is 2%, any frequent pattern will have support not less than 2%. So based on downward closure property, the item 'p' will not generate frequent pattern and henceforth can be removed. Therefore, CFP-growth considers set of items {i,j,k,l,m,n,o,p} for generating frequent patterns. However, it can be observed that the infrequent item 'o' will never generate any frequent pattern. The reason is that any pattern involving item 'o' can have support at most equivalent to 2%, which is less than the MIS value of 'o' i.e., 3% constructed by CFP-growth approach.

3 PROPOSED APPROACH

The proposed algorithm generalizes the CFP-growth algorithm for finding frequent patterns .We call the new algorithm as Improved Complete Frequent Pattern-growth (ICFP-growth). The ICFP-growth approach involves two steps. They are constructing the MIS-tree and generating frequent patterns from the MIS-tree. The ICFP-growth explorer the notions "least minimum support" and "infrequent child node pruning" for constructing the MIS-tree, so that the size of the resultant MIS-tree may be less than or equivalent to the MIS-tree constructed by CFP-growth approach.

3.1 Least Minimum Support

The frequent patterns mined using multiple minsup values follow "sorted closure property" says, in a frequent pattern, all the supersets involving the item having lowest MIS value should be frequent. So in every frequent pattern, frequent item represents the item having the lowest MIS value. Therefore, every frequent pattern will have support greater than or equal to lowest MIS value among all the frequent items. Thus, if we remove 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) refers to the lowest MIS value among all the frequent items.

The significance of this notion is illustrated in example3

Example2: Continuing with the Example 1, it can be observed that the set of items {i,m,n} 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 {o,p} and considers {i,j,k,l,m} 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 '' C '' I.

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.

3.2.1 Constructing MIS-tree

The construction of MIS-tree in ICFP-growth algorithm is shown in Algorithm 1 and described as follows. The ICFP-growth algorithm accepts transaction dataset (Trans), Item set (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 procedure 1). Next, starting from the last item in the item-header table (i.e., item having lowest MIS value) perform tree-pruning 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, then move to 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 is the LMS value. Let the resultant item header table be MinFrequentItemHeaderTable. Call MisMerge procedure (See, procedure 4) to merge the tree. Finally,call InfrequentChildNodePruning procedure (See, procedure 5) to prune the infrequent child nodes in the MIS-tree. The resultant MIS-tree is the compact MIS-tree.

3.2.2 Mining frequent patterns from MIS-tree

Mining the frequent patterns from the compact MIS- tree is shown in Algorithm 6. The process of mining the MIS-tree in ICFP-growth is almost same as mining the 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 Procedure 6). If an suffix item is not a frequent 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. 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 the patterns in which it represents the item having lowest MIS value will also be infrequent. So, for an infrequent suffix item, it is not necessary to construct conditional pattern base.

3.2.3 Infrequent child node pruning

The ICFP-growth approach skips the construction of conditional pattern bases for the infrequent suffix items. So in the compact MIS-tree, the child nodes belong to infrequent items have no significance because its prefix paths (conditional pattern bases) are not used. So, if we can prune the child nodes belonging to infrequent items, the resultant MIS-tree will still preserve the transaction details pertaining to frequent patterns. So, in the MIS-tree, “infrequent child node pruning” is performed such that every branch ends with the node of a frequent item. pruning should be performed only on the child nodes belonging to infrequent items. We illustrate the “infrequent child node pruning”in Example 3.

Procedure1:MIS-tree(Tran:transaction dataset, I: item set containing n items, MIS: minimum item support values for n items)

1: Let L represent the set of items sorted in non-decreasing order of their MIS values.
2: create the root of a MIS-tree, T, and label it as "null".
3: **for** each transaction t # 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 $ 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 MinFrequentItem-HeaderTable.
17: Call MisMerge(Tree).
18: Call InfrequentChildNodePruning(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 item name then **then**
3: merge these nodes and set the count as the summation of these nodes counts.
4: **end if**
5: **end for**

Procedure5 InfrequentChildNodePruning (Tree)

1: choose the last but one item i*j* in MinFrequentItemHeaderTable. That is, item having second lowest MIS value.
2: **repeat**
3: **if** ij item is infrequent item **then**
4: using node-links parse the branches of the Tree.
5: **repeat**
6: **if** ij node is the child 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 child 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

Procedure 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 %=i*j* **!** # with support = ij.support;
4: construct %'s conditional pattern base and %'s conditional MIS-tree Tree %.
5: **if** Tree % $0 **then**
6: call CpGrowth(Tree %, %, MIS(i*j*)).
7: **end if**
8: **end if**
9: **end for**

Procedure7 CpGrowth(Tree, #, MIS(#))

1: **for** each i*j* in the header of Tree **do**
2: generate pattern %=i*j* **!** # with support = i*j* .support;
3: construct %'s conditional pattern base and %'s conditional MIS-tree Tree%
4: **if** Tree% $0 **then**
5: call CpGrowth(Tree%,%,MIS(#)).
6: **end if**
7: **end for**

Example3: Continuing with example 1,2, let the MIS-tree derived after performing a single scan on the dataset contain three branches, say !i,j", !j,k" ,!k,l",!l,m",!m,n". Among the set of items{i,j,k,l,m,n}, we know that items j,k,l are infrequent items. First, let us consider the item l, having relatively lowest MIS value for pruning..This process is repeated until all infrequent items are pruned. Thus the final resulted MIS-tree contains only two branches !i",!l,m" and !m,n".

4. Conclusion

In this paper, we have proposed an efficient algorithm by using the notions" least minimum support" and "infrequent child node pruning" so that the size of the resultant MIS- tree may be less than or equivalent to the MIS-tree constructed by CFP-growth approach. ICFP-growth approach overcoming the performance problems in IMSApriori.The memory requirements of ICFP-growth approach will never exceed those of CFP-growth approach. Mostly, it requires relatively less memory.

References

1.Agrawal, R. and Srikanth, R. (1994). *Fast algorithms for mining association rules.* In VLDB.

2.B. Liu, W. H. and Ma, Y. (1999*).Mining association rules with multiple minimum supports*. In SIGKDD Explorations.

3.ErayOzkural, C. A. (2004). *A space optimization for fp-growth*. InFIMI04.

4.G. Melli, R. Z. O. and Kitts,B.(2006).*Introduction to the special issues on successful real-world data miningapplications.*InSIGKDDExplorations, volume8

5.H. Jiawei, P. Jian, Y. Y. andRunying, M. (2004).*Mining frequent patterns without candidate generation: A frequent-pattern tree approach*.InDMKD.

6.Kiran,R.U.andReddy,P.K.(2009).*An improved multiple minimum support based approach to mine rare association rules*.

7.In IEEE Symposium on Computational Intelligence and DataMining. Mannila,H.(1997).

8.R. Agrawal, T. I. andSwami, A. (1993). *Mining association rules between sets of items in large databases*. In SIGMOD.

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

10.Kiran, KrishnaReddy,(2010), *Rare association rule mining using FP-growth algorithm.*

T Ravi Kumar received M.Sc.(Comp.Sc.) from Andhra University, and M.Tech(CSE) from JNTU Hyderabad. He has got 15 years experience in teaching for UG and PG students in several colleges. His area of research interests are Data Mining, Wireless sensor networks and Web Technologies. Currently he is doing research in Web data mining.

K.Raghava Rao,Professor in CSE, having 12 years experience in teaching for UG and PG students and 3 years of Software Development experience in Singapore. He received B.E(CSE) from MU Univeisty, M.Tech(CSE) from RVP Univeristy,Udaipur and Ph.D(CSE) from MahatamaGandhiUniveristy(Kasividyapeeth), Varanasi, in the years 1995, 2005 and 2009 respectively. He published serveral papers in national & international conferences and journals. His research inetests are Data Mining, Wireless sensor networks and Web Technologies.