Comparative Analysis of Single and Multiple Minimum Support Based Association Rule Mining Algorithms

Devashree Rai, Kesari Verma, A. S. Thoke

***Abstract*—Association rule mining techniques discover associations between entities. Some techniques are single minimum support based while some are multiple minimum support based. Single minimum support based approach suffer from rare item problem dilemma while multiple support based approach considers rare items for mining association rules. In this paper we have evaluated performance of Apriori-T and MSApriori-T Algorithms that are single and multiple minimum based approaches respectively. These Algorithms uses an efficient data storage mechanism Total support tree for storing item sets.**

***Index Terms*—Apriori-T, Association rule mining, MSApriori-T, Total support tree.**

I. INTRODUCTION

Data mining is an extensively studied area [1], [2]; approach that has important consideration is association rule mining. It aims to extract interesting correlations, frequent patterns, associations or casual structures among sets of items in the transaction databases or other data repositories.

Problem Definition: Let $I= \{i_1, i_2, .. i_m\}$ be a set of literals called items. Let D be a set of transaction where any transaction $T \in D$ is a set of items such that $T \subseteq I$. Each transaction is associated with a unique identifier TID. A transaction, $T \in D$ contains a set of items or an item set, $X \subseteq I$, if $X \subseteq T$. An association rule is an expression of the form $X \Rightarrow Y$, where $X \subseteq I$, $Y \subseteq I$, and $X \cap Y = \phi$. The rule $X \Rightarrow Y$ holds in D with support, $\sup(X \Rightarrow Y) = P(X \cup Y)$, and confidence , $conf(X \Rightarrow Y) = P(Y \mid X)$. There are various algorithms proposed [3], [4], [5], [6] to mine association rules. First step to generate such rules is to determine the support for sets of items (I) that may be present in the data set, i.e., the frequency with which each combination of items occurs. After eliminating those I for which the support fails to meet a given minimum support threshold, the remaining large I can be used to produce association rules (ARs). The ARs generated are usually pruned according to some notion of confidence in each AR. However this pruning is achieved, it is always necessary to first identify the "large" I contained in the input data. This in turn requires an effective storage structure. Total support tree [7] is an efficient data storage mechanism to store item sets. Therefore in this paper we will consider two well known association rule mining algorithms Apriori-T [7] and MSApriori-T [8] that uses total support tree data structure for item set storage.

Manuscript received March 31, 2012

Devashree Rai, Department of Electrical Engineering, National Institute of Technology Raipur, Raipur, India, 09893872292, (e-mail: rai.devashree@gmail.com).

Kesari Verma, Department of Computer Application, National Institute of Technology Raipur, Raipur, India, 07489355510 (e-mail: keshriverma@gmail.com).

A.S. Thoke, Department of Electrical Engineering, National Institute of Technology Raipur, Raipur, India, 09329750686, (e-mail: asthoke@yahoo.co.in).

II. TOTAL SUPPORT TREE DATA STRUCTURE

The number of possible combinations represented by the items in the input data set scales exponentially with the size of records, which results in increase in storage requirement as well as processing time of mining association rules. A partial solution is to store only those combinations that actually appear in the data set. A further mechanism is to make use of the downward closure property of item sets—"if any given item set I is not large, any superset of I will also not be large." This can be used effectively to avoid the need to generate and compute support for all combinations in the input data. However, the approach requires: 1) a number of passes of the data set and 2) the construction of candidate sets to be counted in the next pass.

The T-tree (Total Support Tree) is a compressed set enumeration tree structure [10]. The T-tree differs from standard set enumeration trees in that the nodes at the same level in any sub-branch are organised into 1-D arrays such that the array indexes represent column numbers. For this purpose it is more convenient to build a "reverse" version of the tree, which permits direct indexing with attribute/column numbers.

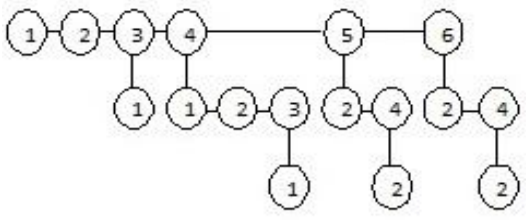

Fig 1. Structure of Total Support Tree

The T-tree uses index mechanism which makes traversal fast. In t-tree item set labels are not required to be explicitly stored and thus no sibling's reference variables (pointers) are required. Total support tree structure for the data set r1:{1,3,4}, r2:{2,4,5}, r3:{2,4,6} is shown in fig 1., Level 1 represent single item sets and store support values for each singleton. Level 2 represents pair of two and soon.

III. APRIORI-T

Apriori-T is an association rule mining algorithm that uses total support tree data structure proposed in [7] which is more efficient than Agrawal and Srikant's Apriori algorithm [3]. The algorithm takes data set as input and produce association rules as output. It works by creating nodes in a T-tree level by level. Each node in the T-tree is an object (T treeNode) comprised of a support value (sup) and a reference (chldRef) to an array of child T-tree nodes. The Apriori T-tree generation algorithm has following steps:

1. Create top level of tree
2. Add support to top level
3. Prune nodes in top level
   - If(node.sup<minsup)
   - node=null
4. Generate second level
5. While(nextlevelexist)
6. Add support to newly generated level
7. Prune the support added level
8. Generate next level
9. End

First step create top level of T-tree where number of nodes will be equal to number of attributes (number of items) in input data set. In second step support is added to each node in first level and in third step nodes are pruned if their support value is less than minimum specified support. The same way other levels are created and pruned. The final tree after pruning is used to generate association rules.

It can be observed that real-world datasets are mostly non-uniform in nature containing both frequently and relatively rarely occurring entities. In literature, it has been reported that there exists useful knowledge pertaining to rare entities [9], [11]. For extracting rare item sets, the single minimum support based approach Apriori suffers from "rare item problem" dilemma [12]. At high minimum support value, rare item sets could not be extracted as rare items fail to satisfy the minimum support criteria. Also, if low minimum support is set to extract rare item sets, the number of item sets explodes as huge number of frequent item sets satisfies the minimum support criteria.

IV. MSAPRIORI-T

MSApriori-T algorithm uses multiple supports instead of single to improve the performance of extracting frequent item sets involving rare items. The MSApriori-T algorithm using total support tree data structure has been proposed in [8]. Each item is assigned with a minsup value known as "Minimum Item Support" (MIS) and frequent item sets are generated if an item set satisfies the lowest MIS value among the respective items. In MSApriori-T each node (TtreeNode) will have support value (sup), multiple support value (MIS) and reference to child nodes array (chdref). Each level is created and corresponding support and MIS value is stored in it. The MSApriori T-tree generation algorithm has following steps:

1. Create top level of tree
2. Add support and MIS value to nodes in top level.
   - For every item $i_j \in I$, $MIS(i_j)$ is calculated as per equation 1.
     - $MIS(i_j) = \beta \times S(i_j)$, if $\beta \times S(i_j) \succ LS$ (1)
       $= LS$ else
3. Prune nodes in top level
   - If(node.sup< node.MIS)
   - Node=null
4. Generate second level
5. While(nextlevelexists)
6. Add support and MIS value equal to lowest MIS of items in item set.
7. Prune the nodes in newly generated level.
8. Generate next level
9. End

The MIS value is assigned to each item equal to a percentage of its support according to equation 1, where β is a user specified proportional value which can be varied between 0 to 1, $S(i_j)$ refers to support of an item equal to $f(i_j)/N$, ($f(i_j)/N$ represents frequency of i_j and N is the number of transaction in a transaction data set) and LS corresponds to user specified least support value.

V. EXPERIMENTAL EVALUATION

The algorithms are implemented in java 1.7.0, i5 processor, windows 7 operating system. Graphs below are obtained after experimenting with some real data sets taken from [13].

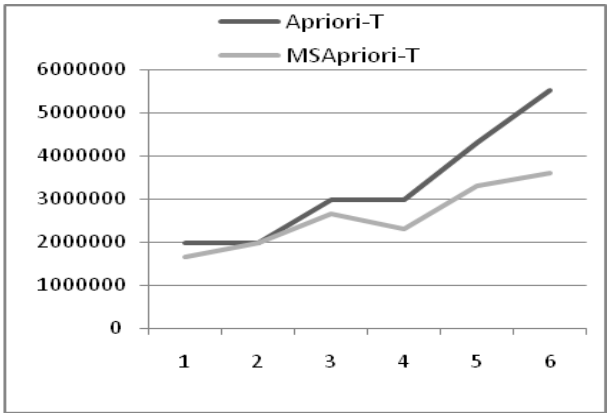

Fig 2. Memory (in bytes) comparison for minsup=20%

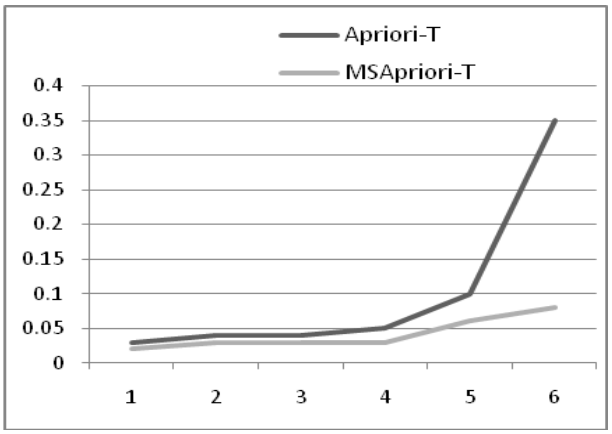

Fig 3. Time (in sec) requirement for minsup=20%

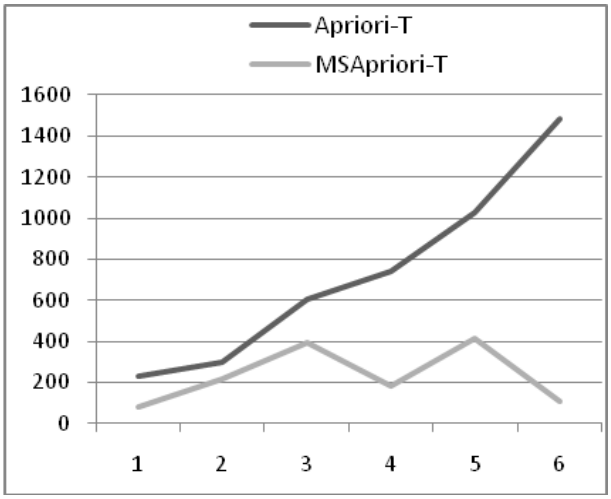

Fig 4. Number of frequent items for minsup=20%

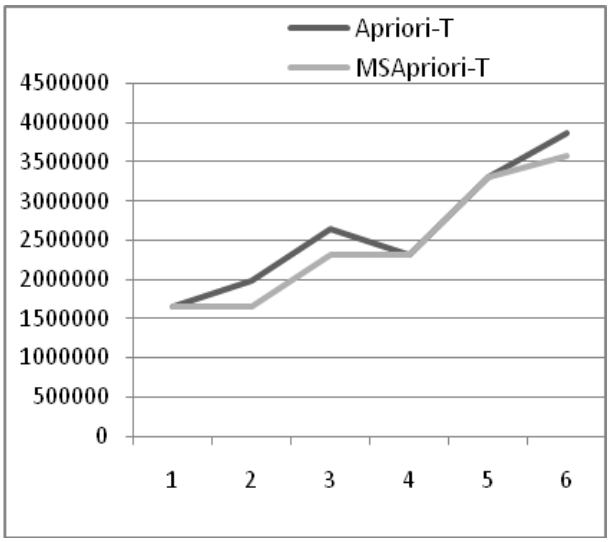

Fig 5. Memory (in bytes) comparison for minsup=40%

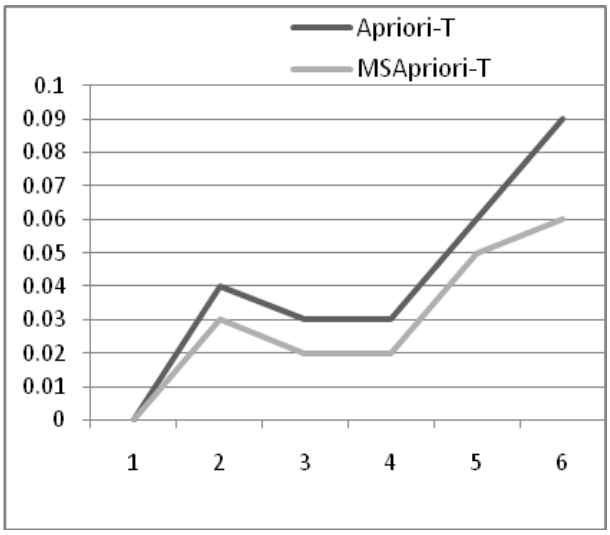

Fig 6. Time (in sec) requirement for minsup=40%

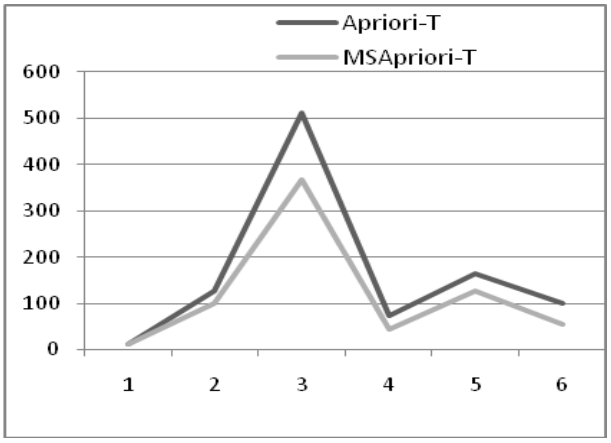

Fig 7. Number of frequent item sets for minsup=40%

A. Analysis

Above graphs compares memory requirement (Fig 2 and Fig 5), time taken (Fig 3 and Fig 6) and number of frequent item set generated (Fig 4 and Fig 7) of the two algorithms for minimum support (Least Support for MSApriori-T) 20% and 40%. It can be observed that for the same minsup value while some data set have low difference in the graph for the two methods, some data set shows explosion of number of frequent item sets in Apriori-T, therefore increases memory and time requirement of algorithm. Though result depends on type of data set, minimum support value but still in most of the cases MSApriori-T proves to be a better approach.

VI. CONCLUSION

Various algorithms are proposed for mining association rules. The memory and time requirement for the algorithms become high due to need to generate large item sets. As memory and time complexity are two important factors for measuring efficiency of an algorithm. This is being optimized by using total support tree data structure. Moreover rare items are ignored by most of the algorithms. MSApriori-T is better algorithm as it considers rare items while generating association rules while maintaining better performance in memory and time requirement.

REFERENCES

1. M.S. Chen, J. Han, P.S. Yu, "Data mining: an overview from a database perspective", *IEEE Transactions on Knowledge and Data Engineering*, 1996, 8, pp. 866-883.
2. J. Han, M. Kamber, *Data Mining: Concepts and Techniques*, Morgan Kaufmann Publisher, San Francisco, CA, USA, 2001.
3. Agrawal, R., Imielinski, T., and Swami, A. "Mining association rules between sets of items in large databases." SIGMOD, 1993, pp. 207-216.
4. Agrawal, R., and Srikanth, R. "Fast algorithms for mining association rules." VLDB, 1994.
5. J.Han, Y. Fu, "Discovery of multiple-level association rules from large database", in *the twenty-first international conference on very large data bases*, Zurich, Switzerland, 1995, pp. 420-431.W.-K. Chen, *Linear Networks and Systems* (Book style). Belmont, CA: Wadsworth, 1993, pp. 123–135.
6. J.Han, J.Pei and Y.Yin., "Mining frequent patterns without candidate Generation", in: *Proceeding of ACM SIGMOD International Conference Management of Data*, 2000, pp. 1-12.

7. Coenen and Leng (2004). Data Structures for Association Rule Mining: T-trees and P-trees to appear in IEEE Transaction in Knowledge and Data Engineering.
8. Devashree Rai, Kesari Verma, A.S. Thoke "MSApriori using total support tree data structure" (accepted for publication) IJCA, to be published.
9. Liu, B., Hsu, W., and Ma, Y. "Mining Association Rules with Multiple Minimum Supports." SIGKDD Explorations, 1999.
10. R. Rymon, "Search through Systematic Set Enumeration," Proc. Third Int'l Conf. Principles of Knowledge and Reasoning, pp. 539-550, 1992.
11. Weiss, G. M. "Mining With Rarity: A Unifying Framework." SIGKDD Explorations, 2004, Vol. 6, Issue 1, pp. 7 - 19.
12. Mannila, H. "Methods and Problems in Data Mining." ICDT, 1997.
13. Uci: Blake, c.l., & Merz, C.J (1998) UCI repository of machine leaning databases from www.ics.uci.edu/~mlearn/MLrepository.html.

AUTHORS PROFILE

Devashree Rai, Completed B.E. (Computer Sc.) in 2009.Currently pursuing MTECH (Computer Technology) from National Institute of Technology Raipur. Presented three papers in national and international conferences.

Kesari Verma Post graduated in 1998 and got doctorate in 2007. She is current Assistant professor in National Institute of Technology, Raipur.

She has 12 year R & D and Teaching experience. She has more than 30 publications in journals and conferences.

A.S. Thoke (Born 1950) graduated in Electrical engg. In 1972, Post graduation in 1978(Power System) and PhD (Electrical eng) in 2005.
He joined technical education of M.P. (India) in July 74 as lecturer Electrical Engg. Currently he is a Professor in Electrical Engg. in National Institute of Technology NIT, Raipur (C.G.) India.
He has a teaching and R&D experience of 38 years. He has about 50 publications in journals and conferences.