Volume 3, No. 1, Jan-Feb 2012
ISSN No. 0976-5697
International Journal of Advanced Research in Computer Science
RESEARCH PAPER
Available Online at www.ijarcs.info

A Bitmap Approach for Closed and Maximal Frequent Itemset Mining

V. Umarani
Assistant Professor, Department of Computer Science,
Sri Ramakrishna College of Arts & Science for Women,
Coimbatore, India
v_umarani@yahoo.com

Shanmugha Priya. K*
Research Scholar, Department of Computer Science,
Sri Ramakrishna College of Arts & Science for Women,
Coimbatore, India
priya_cbe86@yahoo.com

Abstract: Association Rule Mining (ARM) plays a fundamental role in many data mining tasks that attempt to find interesting patterns from databases, such as correlations, sequences, episodes, classifiers, clusters, etc. Frequent Itemset Mining (FIM) is one of the essential parts of ARM which is an active research area and a large number of algorithms have been developed. FIM algorithms may be depth-first or breadth-first approach. Most depth-first based approaches do not effectively address the cost of database projections. Consequently, their performance gets degraded severely as the total number of frequent itemsets in a database increases significantly. To solve this problem, a three - strategy adaptive algorithm, Bitmap Itemset Support Counting (BISC) with the closed and maximal frequent itemset is presented in this paper. The core strategy of the proposed algorithm is the usage of the closed and maximal frequent itemset detection which could be able to find least number of association rules. The proposed approach is compared with the conventional approach to evaluate the performance and accuracy. The experimental results show that the proposed approach outperforms the existing approach.

Keywords: Data Mining Algorithms; Bitmap; Frequent Itemset Mining; Association Rule Mining.

I. INTRODUCTION

Association Rule Mining (ARM) is a technique used to discover relationships among a large set of variables in a dataset [1]. It has been applied to a variety of industry settings and disciplines. A typical and widely-used example of association rule mining is *Market Basket Analysis*. For example, the information that customers who buy coffee and sugar also tend to buy milk could be represented by the following association rule: buy (*X, coffee*) and buy (*X, sugar*) => buy (*X, milk*). In general, an association rule is an implication of the following form: *A* => *B*, where *A* and *B* is sets of items. Several items constitute a transaction. A transaction is defined as a set of items bought together. The intended meaning of this association rule is that consumers who buy all items in itemset A also tend to buy all items in itemset B.

Frequent Itemset Mining (FIM) is an important data mining problem that detects frequent itemsets in a transaction database. It is one of the major processes for association mining and plays a fundamental role in many data mining tasks that attempt to find interesting patterns from databases such as correlations, sequences, episodes, classifers, clusters, etc. Many algorithms such as Apriori, FP- growth and Eclat have been proposed to solve the problem.

A *frequent itemset* is one that occurs in at least a user-specific percentage of the database. That percentage is called support. An itemset is *closed* if none of its immediate supersets has the same support as the itemset. An itemset is *maximal* frequent if none of its immediate supersets is frequent.

Bitmap is used for constant number of items [2]. Many algorithms are generally recursive and reduce the database recursively. Thus, the reduced databases usually include a constant number of items in the bottom levels of recursion. For such small databases, the efficiency of the bitmap is used for frequency counting. Bitmap stores the transaction database by a *01* matrix; such that the *ij* element of the matrix is 1 if and only if item is included in j^{th} transaction. Each cell can be represented by *1* bit, thus memory can be saved especially in the case that the database is dense. The advantage of using bitmap is fast database reduction and the processing speed.

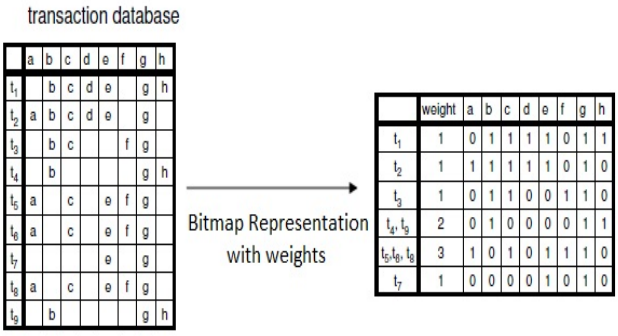

Figure 1. Bitmap Representation of Transaction Database.

Bitmap notation is commonly used in the association rule mining context. In this representation, each transaction *t* in the database *D* is a triple: *t =<TID; values of items; weight>* where, *TID* is the identifier of the transaction *t* and *values of items* is a list of values with one value for each item in the list of items *I* and *weight* is the weight of the transactions, that is, the number of items in the transaction t. An item is supported by a transaction *t* if its value in the *values of items* is 1 and it is not supported by *t* if its value in *values of items* is 0. *Weight* is the number of values which appear in the *values of items* (e.g., the number of items supported by transaction *t*).

The performance [3] of a depth-first frequent itemset (FI) miming algorithm is closely related to the total number of recursions. In previous approaches this is mainly decided by the total number of FIs, which results in poor performance when a large number of FIs are involved. To solve this problem, a three-strategy adaptive algorithm, bitmap itemset support counting (BISC), is presented. The core strategy, BISC1, is used in the innermost steps of the recursion. For a database *D* with only *s* frequent items, a

depth-first approach need up to *s* levels of recursions to detect all the FIs (up to 2^s). BISC1 completely replaces these recursions with a special summation that directly calculates the supports of all the possible 2^s candidate itemsets. With BISC1 the run-time is entirely independent of the database after one database scan, and the per-candidate cost is only *s*. To offset the exponential growth of cost (both time and space) with BISC1 as *s* increases, a second strategy, BISC2, is introduced to effectively double the acceptable range of *s*. BISC2 divides an itemset into prefix and suffix and improves the performance by pruning all the itemsets with infrequent prefixes. If the total number of frequent items in *D* is high, the classic database projection strategy is used. In this case for the first *s* items a single run of BISC (1 or 2) is applied. For each of the remaining items, a projected database is created and the mining process proceeds recursively. To achieve optimal performance, BISC adaptively decides which strategy to use based on the dataset and minimum support.

II. RELATED WORKS

Algorithms for mining frequent itemset and the bitmap related issues are discussed in this section. Apriori algorithm is an iterative; bottom-up, breadth-first search algorithm. It employs the downward closure property for candidate pruning. The algorithm has two main parts, candidate generation and validation. Apriori is effective at candidate pruning but inefficient at support counting. Many algorithms such as AprioriTid [4], DHP, DIC [5], DCP, DCI, kDCI by [6] etc have been proposed to improve the efficiency of the support counting procedure.

The FP-Growth algorithm [7] uses a special data structure FP-Tree which eliminates the need of explicit candidate generation. FP-Tree combines the vertical and horizontal layout for a compact representation of databases. FP-Tree has an associated item list which maintains a linked list for each item to record all the transactions that contain it. FP-growth is generally more effective in dense databases than in sparse ones. Its major cost is the recursive construction of the FP-Trees.

The group bitmap index [8] is a new index type which significantly reduces time of subset searching in large databases. This kind of searching has many applications in the field of data mining and association rules discovery. However, the new index may be also applied in traditional database systems to speed-up the execution of queries seeking for a subset of data items. Experiment shows that the group bitmap index significantly outperforms traditional indexing methods such as B+ tree and bitmap indexing.

A bitmap based association rule algorithm [9] using granular computing technique develops a bitmap based algorithm (Bit-AssocRule) to find association rules. Bit-AssocRule avoids the time-consuming table scan to find and prune the itemsets; all the operations of finding large itemsets from the datasets are the fast bit operations. The experimental result of Bit-AssocRule algorithm with Apriori, AprioriTid and AprioirHybrid algorithms shows Bit-AssocRule is 2 to 3 orders of magnitude faster. This research indicates that bitmap and granular computing techniques can greatly enhance the performance for finding association rule, and bitmap techniques are very promising for the decision support query optimization and data mining applications.

MAFIA [10] is an algorithm for mining maximal frequent itemsets from a transactional database (it has however the option to mine the closed sets as well). It is especially efficient when the itemsets in the database are very long. The search strategy integrates a depth-first traversal of the itemset lattice.

CHARM algorithm [11] is used for mining all frequent closed itemsets. CHARM simultaneously explores the itemset space and transaction space, rather than only the itermset search space. It also uses a highly efficient hybrid search method that skips many levels of the IT-tree (itemset-tidset tree) to quickly identify the frequent closed itemsets, instead of having to enumerate many possible subsets.

III. PROPOSED METHODOLOGY

INPUT: Dataset (Chess/Mushroom/Retail)
OUTPUT: Frequent itemsets, Discovering association rules, Accuracy

A. Proposed Algorithm – CFIM:

The steps of the proposed algorithm (CFIM) are as follows:

a. Load the transaction database D and the prefix itemset of D.
b. Detects all the frequent items and their frequencies in *D*.
c. Then decides the optimal value of *s*.
d. If *s* is larger than the total number of frequent items, it will detect *FIS(D)* using BISC. Otherwise, it only detects all the *s*-frequent itemsets with BISC.
e. Detects the closed frequent items from the generated frequent items using the following condition.
   (a) Check whether it has the same support of frequent item and if so eliminate it from the closed frequent itemsets.
   (b) Otherwise add that item to the closed frequent itemsets.
f. Then generates projected database s for items in between the $(s+1)^{th}$ and n^{th} frequent items (n is the total number of frequent items).
g. Recursively detect *FIS* in each projected database using condition [as in step 5(a) & (b)].
h. The outcome of the process will be the complete set of closed frequent itemsets in D.

B. Proposed Algorithm – MFIM:

The steps of the proposed algorithm (MFIM) are as follows:

a. Load the transaction database D and the prefix itemset of D.
b. Detects all the frequent items and their frequencies in *D*.
c. Then decides the optimal value of *s*.
d. If *s* is larger than the total number of frequent items, it will detect *FIS(D)* using BISC. Otherwise, it only detects all the *s*-frequent itemsets with BISC.
e. Detects the maximal frequent items from the generated frequent items using the following condition.
   (a) Check whether it is a subset of frequent item and if so eliminate it from the maximal frequent itemsets.
   (b) Otherwise add that item to the maximal frequent itemsets.
f. Then generates projected database s for items in between the $(s+1)^{th}$ and n^{th} frequent items (n is the total number of frequent items).
g. Recursively detect *FIS* in each projected database using condition [as in step 5(a) & (b)].

h. The outcome of the process will be the complete set of maximal frequent itemsets in D.

IV. PERFORMANCE COMPARISON

This section gives an overview of the conducted experiments and presents the obtained results to evaluate the performance of the BISC with closed and maximal frequent itemsets with respect to the other FIM algorithms. The performance metrics in the experiments is the total execution time taken, the memory capacity for the process and the accuracy obtained by all algorithms to generate frequent itemsets for different datasets. Clearly, work could see an appreciable reduction in the time and the memory required for ARM using the existing approach and the proposed approach. Also, the proposed approach achieves a high accuracy than the existing approach.

The experimental results show that the proposed BISC based closed and maximal algorithm outperforms the existing algorithm by means of time, memory and accuracy in all the three datasets – chess, mushroom and retail.

The first set of experiment was conducted by the chess dataset to evaluate the performance of the proposed system in terms of time (in seconds), memory (in bytes) and accuracy (in percentage). The proposed system is compared with the existing system and the comparison result is shown as graph and table.

Time is measured by the computational time taken in seconds for the execution process of all the algorithms in the chess dataset. The following figure shows that the execution time for the closed and maximal algorithm is very less when compared to apriori, FP-growth and BISC.

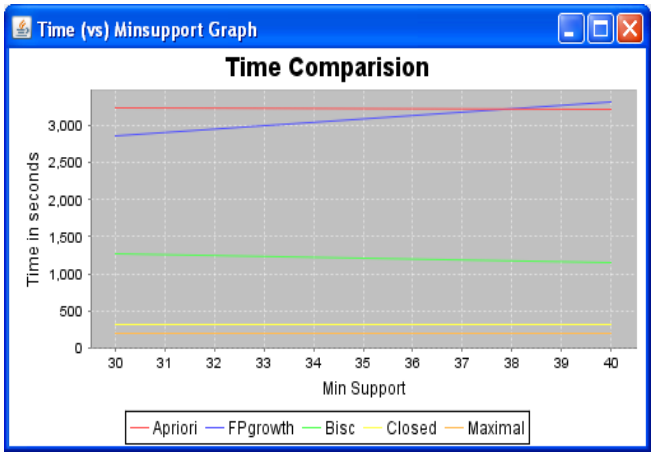

Figure 2 - Graph representing the time of the chess dataset.

Figures 2 shows the processing time versus different minimum support values for the Apriori, FP-growth, BISC, closed and maximal algorithms using the chess dataset. It indicate that as the minimum support increases, the processing time of the proposed algorithm decreases due to the decreased total number of frequent itemsets returned. Also, notice the dramatic increase of processing time of Apriori is about 3223 seconds and FP-growth is about 3312 seconds when minimum support is 40%.

Memory space occupied in bytes for storing the data of the process of all the algorithms of the chess dataset is compared. The following figure shows that the memory consumed by the closed and maximal algorithm is very less when compared to apriori, FP-growth and BISC.

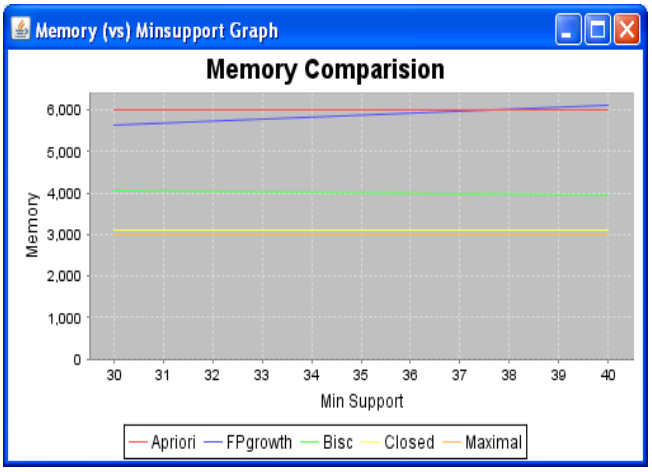

Figure 3 - Graph representing the memory of the chess dataset.

Figures 3 shows the memory consumption versus different minimum support values for the Apriori, FP-growth, BISC, closed and maximal algorithms using the chess dataset. It indicates that as the minimum support increases, the memory of the proposed algorithm decreases due to the decreased total number of frequent itemsets returned. Also, notice the dramatic increase of memory consumed by Apriori is about 6008 bytes and FP-growth is about 6097 bytes when minimum support is 40%.

The rule has been discovered from the training data of the chess dataset by applying association rule approaches such as apriori, FP-growth, BISC, closed and maximal frequent itemsets. Accuracy is calculated based on the number of rules discovered from the training data which is matched with the testing data of the chess dataset. The following figure shows that the accuracy of the closed and maximal algorithm is very high when compared to apriori, FP-growth and BISC.

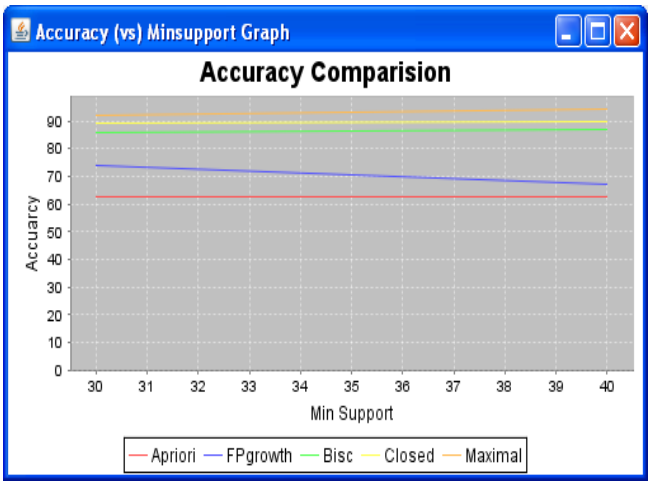

Figure 4 - Graph representing the accuracy of the chess dataset.

Figures 4 shows the accuracy versus different minimum support values for the Apriori, FP-growth, BISC, closed and maximal algorithms using the chess dataset. It indicates that as the minimum support increases, the accuracy of the proposed algorithm increases due to the decreased total number of frequent itemsets returned. Also, notice the dramatic decrease of accuracy of Apriori is about 63% and FP-growth is about 67% when minimum support is 40%.

The table 1 below shows the execution time (in seconds), memory (in bytes) and accuracy (in percentage) for all the

algorithms with minimum support threshold 40% for the chess dataset.

Table 1 - Performance Comparison Table for Chess Dataset

Performance Measures with Minimum Support Threshold 40%				
	Algorithm	**Time (in Sec)**	**Memory (in Bytes)**	**Accuracy (in %)**
Existing Approach	Apriori	3223	6008	63
	FP-Growth	3312	6097	67
	BISC	1156	3941	87
Proposed Approach	Closed FI	312	3097	90
	Maximal FI	203	2988	94

The above table indicates that the performance measures of the proposed approach are better than the existing approach in terms of time, memory and accuracy. The closed and maximal frequent itemset takes less computational time, occupies very less memory and also shows a high level of accuracy when compared to the existing approaches.

V. CONCLUSION AND FUTURE ENHANCEMENT

In this paper, a new algorithm for closed and maximal frequent itemsets with BISC is proposed for efficient frequent itemset mining. Extensive experiments have been performed to evaluate the performance of the proposed algorithm using the datasets such as chess, mushroom and retail. The proposed system outperforms all the existing system of the three datasets namely chess, mushroom and retail that have been tested in terms of time efficiency. The proposed system shows that the space complexity (i.e., memory usage) also very low when compared to the existing system. Experimental results show that the proposed algorithm achieves good accuracy in a reasonable processing time with different minimum support values which achieves till 94%.

The proposed work can further be extended to the other datasets like distributed datasets. In addition, the work can also be extended the concepts and strategies that contribute to the success of BISC to other related areas such as mining frequent itemsets under constraints and sequential pattern mining. The exploration of the possibility of applying suitable form of parallel processing of the BISC techniques, to enhance computational speed is another scope of research.

VI. REFERENCES

[1]. S. Kotsiantis and D. Kanellopoulos," Association rules mining: a recent overview", GESTS International Transactions on Computer Science and Engineering, pp. 71 - 82, 2006.

[2]. T. Uno, M. Kiyomi, H. Arimura, "LCM ver. 3: Collaboration of array, bitmap and prefix tree for frequent itemset mining", Proceedings of the ACM SIGKDD Data Mining Workshop on Frequent Pattern Mining Implementations, pp.77–86, 2005.

[3]. Jinlin Chen, Keli Xiao, "BISC: A bitmap itemset support counting approach for efficient frequent itemset mining", ACM Transactions on Knowledge Discovery from Data (TKDD), October 2010.

[4]. R. Agrawal and R. Srikant, "Fast algorithms for mining association rules", Proceedings of 20th International Conference on Very Large Data Bases, pp. 487-499, 1994.

[5]. Sergey Brin, Rajeev Motwani, Jeffrey D. Ullman, and Shalom Tsur, "Dynamic Itemset Counting and Implication Rules for Market Basket Data", Proceedings of the ACM SIGMOD Conference, pp. 255-264, 1997.

[6]. S. Orlando, P. Palmerini, R. Perego and F. Silvestri, "Adaptive and resource-aware mining of frequent sets", Proceedings of the IEEE International Conference on Data Mining, pp. 338–345, 2002.

[7]. J. Han, J. Pei and Y. Yin, "Mining frequent patterns without candidate generation", Proceedings of ACM SIGMOD International Conference on Management of Data, pp. 1-12, 2000.

[8]. Tadeusz Morzy, Maciej Zakrzewicz, "Group Bitmap Index: A Structure for Association Rules Retrieval", Institute of Computing Science, Poznan University of Technology, 1998.

[9]. Hong-Zhen Zheng, Dian-Hui Chu, De-Chen Zhan, "Association Rule Algorithm Based on Bitmap and Granular Computing", AIML Journal, Volume (5), Issue (3), September, 2005.

[10]. Doug Burdick, Manuel Calimlim, Johannes Gehrke, "MAFIA: A maximal frequent itemset algorithm for transactional databases", Proceedings of the 17th International Conference on Data Engineering, Heidelberg, Germany, 2001.

[11]. Mohammed Javeed Zaki, Ching-Jiu Hsiao, "CHARM: An Efficient Algorithm for Closed Itemset Mining", SDM - SIAM International Conference on Data Mining, Chicago, 2002.