Mining Frequent Itemsets by using Binary Search Tree Approach

CH.M.H.Saibaba
Assistant Professor,
Department of CSE & IT,
Aryabhata Institute of Technology & Science,
Mohabatnagar, Hyderabad,

Dr. Rekha Redamalla
Professor of Computer Science & Principal,
Bhoj Reddy College of Engineering for Women
Vinay Nagar Colony, Saidabad,
Hyderabad,

ABSTRACT
Data Mining is the process of extracting hidden patterns from data. Finding frequent itemsets is computationally the most expensive step in association rule discovery. The Efficient Hashing Tree (EHT) algorithm is even faster than Apriori and FP- growth algorithms. Its drawback is however, that the time needed to build a compact tree and the memory requirement depends upon the number of frequent 2 – itemsets. [1]
The above drawbacks are rectified by using Binary Search Tree (BST) algorithm. By using this approach we can construct a binary search tree very quickly by considering the frequent itemsets. This algorithms works well for 1–itemset, 2–itemsets, 3–itemsets and more than 3–itemsets. By using this approach it requires very less memory requirement for mining frequent itemsets.

1. INTRODUCTION

Data mining commonly involves four classes of task:

Classification - Arranges the data into predefined groups. For example an email program might attempt to classify an email as legitimate or spam. Common algorithms include nearest neighbor, Naive Bayes classifier and neural network.

Clustering - Is like classification but the groups are not predefined, so the algorithm will try to group similar items together.
Regression - Attempts to find a function which models the data with the least error. A common method is to use Genetic Programming.
Association rule learning - Searches for relationships between variables. For example a supermarket might gather data of what each customer buys. Using association rule learning, the supermarket can work out what products are frequently bought together, which is useful for marketing purposes. This is sometimes referred to as "market basket analysis".

2. TECHNICAL DETAILS

Decision trees: Tree-shaped structures that represent sets of decisions. These decisions generate rules for the classification of a dataset. Specific decision tree methods include Classification and Regression Trees (CART) and Chi Square Automatic Interaction Detection (CHAID).
Genetic algorithms: Optimization techniques that use process such as genetic combination, mutation, and natural selection in a design based on the concepts of evolution.
Rule induction: The extraction of useful if-then rules from data based on statistical significance. These capabilities are now evolving to integrate directly with industry-standard data warehouse and OLAP platforms.

2.1 Architecture of Data Mining

To best apply the advanced techniques, they must be fully integrated with a data warehouse as well as flexible interactive business analysis tools. Many data mining tools currently operate outside of the warehouse, requiring extra steps for extracting, importing, and analyzing the data. The resulting analytic data warehouse can be applied to improve business processes throughout the organization, in areas such as promotional campaign management, fraud detection, and new product rollout. Figure 1 illustrates architecture for advanced analysis in a large data warehouse.

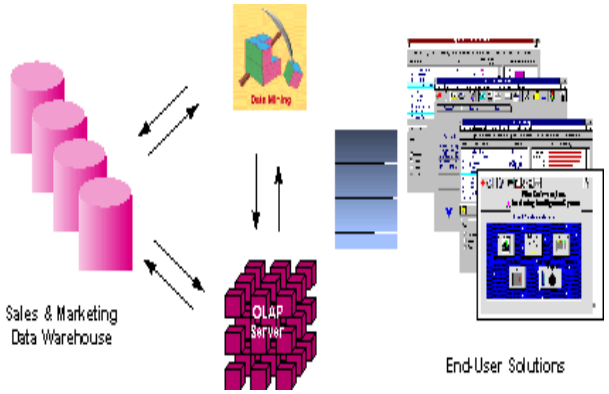

Figure 1 - Integrated Data Mining Architecture

The ideal starting point is a data warehouse containing a combination of internal data tracking all customer contact coupled with external market data about competitor activity. Background information on potential customers also provides an excellent basis for prospecting. This warehouse can be implemented in a variety of relational database systems: Sybase, Oracle, Redbrick, and so on, and should be optimized for flexible and fast data access.
A wide range of companies have deployed successful applications of data mining. While early adopters of this technology have tended to be in information-intensive industries such as financial services and direct mail marketing, the technology is applicable to any company looking to leverage a large data warehouse to better manage their customer relationships.

3. RELATED WORK

Discovery of interesting association relationships among huge amounts of data will help marketing, decision making and business management. Association rule mining is a widely used technique for large – scale data mining, which can be simply defined as a process of finding interesting patterns and trends from a given data. [1]

There are two main strategies for mining frequent itemsets: the candidate generation and test approach and the pattern growth approach. Apriori and its several variations belong to the first approach, while FP – growth and H – mine are examples of the second. Apriori algorithms suffer from the problem spending much of their time to discard the infrequent candidates on each level. Another problem can be the high I/O cost which is inseparable from the level – wise approach. In case of Apriori algorithm the database is accessed as many times as the size of the maximal frequent itemset is. This problem is partly overcome by algorithms based on pattern growth.

The FP–growth (Frequent Pattern – growth) algorithm differs basically from the level – wise algorithms, that use a "candidate generate and test" approach. It does not use candidates at all, but it compresses the database into the memory in a form of a so – called FP – tree using a pruning technique. The patterns are discovered using a recursive pattern growth method by creating and processing conditional FP – trees. The drawback of the algorithm is its huge memory requirement which is dependent on the minimum support threshold and on the number and length of the transactions. [1]

A new algorithm EHT (Efficient Hashing Tree – based) for mining complete frequent itemsets directly from the database. Mining of FP – tree structure is done recursively by building conditional trees that are of the same order of magnitude in number as the frequent patterns, but mining the EHT structure is done recursively by building conditional trees that are of less order of magnitude in number as the frequent patterns. [2]

Association rule mining is a very popular data mining technique and it finds relationships among the different entities of records (for example transaction records). Since the introduction of frequent itemsets in 1993 by Agrawal et al. It has received a great deal of attention in the field of knowledge discovery and data mining. [3] The problem of association rules mining was introduced in as well. This algorithm was improved later to obtain the Apriori algorithm.

Many variants of the Apriori algorithm have been developed, such as AprioriTid, ArioriHybrid, Direct Hashing and Pruning (DHP), Dynamic Itemset Counting (DIC), Partition algorithm, etc. A variant of FP-growth is the H-mine algorithm. It uses array-based and trie-based data structures to deal with sparse and dense datasets respectively. PatriciaMine employs a compressed Patricia trie to store the datasets. [4] FP-growth uses an array technique to reduce the FP-tree traversal time. Eclat is the first algorithm to find frequent patterns by a depth-first search and it has been shown to perform well. It uses a vertical database representation and counts the itemset supports using the intersection of tids. However, because of the depth-first search, pruning used in the Apriori algorithm is not applicable during the candidate itemsets generation. [5] VIPER and Mafia also use the vertical database layout and the intersection to achieve a good performance. However, their compression scheme has limitations especially when tids are uniformly distributed. Zaki and Gouda developed a new approach called Eclat using the vertical database representation. [6]

4. METHODOLOGY

In Binary Search Tree (BST) approach, the frequent itemsets are arranged depending upon the occurrences of itemsets. In this approach the frequent itemsets are arranged in form of the nodes with the technique of binary search tree. The binary search tree is built by considering the 1 – itemset, 2 – itemsets, 3 – itemsets and more than 3 – itemsets. The starting node is constructed and later on if the next node occurrence is more than the root node then it will be arranged on the right side of the root node. If the next node is less than the root node then it will be arranged on the left side of the root node. The different binary search trees are constructed depending upon the itemsets.

The different binary search trees are constructed for 1 – itemsets, 2 – itemsets, 3 – itemsets and more than 3 – itemsets. By using this binary search tree approach for mining the frequent itemsets in data mining, it will take very less memory space for storing the information of nodes in the binary search tree. Another advantage of binary search tree approach for mining the frequent itemsets is that we can identify an itemset which has occurred for many times very quickly without any loss of time. Another important advantage of this binary search tree approach we can search an itemset very quickly.

This approach will take very less time complexity for constructing the binary search tree by considering the itemsets. By considering the itemsets and itemsets are arranged in the form of binary search tree.

Let us consider the 1 - itemsets I1 as 2, I2 as 3, I3 as 4, I4 as 1, I5 as 6 and I6 as 2 occurrences. Then Binary Search Tree approach can be implemented in the following manner:

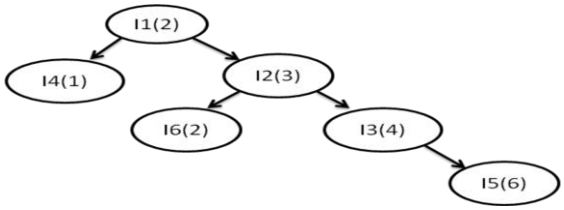

Figure – 2 Binary Search Tree for 1 – itemsets

Let us consider the 2 – itemsets as I1,I2 as 3, I2,I3 as 4, I3,I4 as 5, I4,I5 as 6, I5,I6 as 7, I3, I5 as 2 occurrences, then the approach can be implemented in the following manner:

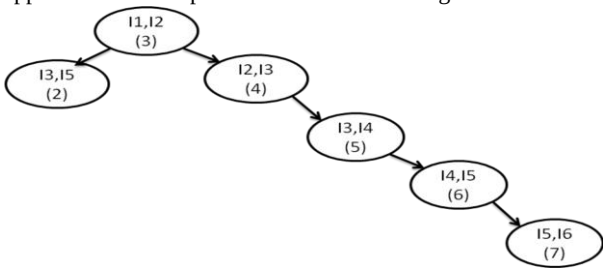

Figure – 3 Binary Search Tree for 2 - itemsets

The following algorithm is used for constructing the binary search tree for mining the frequent itemsets in data mining:

Procedure BST (Tree, Node)
Add first node of the binary tree to H;
For each 2-itemset entry (top down order) in binary tree do
If BST (I) >= Root, then
Create a link to the second tree of binary tree2
End procedure

Procedure buildsubstree (N)
Add first node of Binary Tree to N;
Create a new node for this Binary Tree;
Create a link to the second node of Binary Tree;
End procedure

5. PERFORMANCE AND ANALYSIS

The proposed Binary Search Tree (BST) approach for mining the frequent itemsets is so efficient when we compare with other approaches like Apriori, Apriori TID, Frequent Pattern – Growth (FP–Growth); Efficient Hashing Tree (EHT) based algorithms. By using Binary Search Tree approach for mining frequent itemsets is very quicker and occupies very less memory space for storing the itemsets values. The execution time of the Binary Search Tree method is always smaller than that of the Apriori, Apriori TID, FP-growth and Efficient Hashing Tree based algorithms.

It can be easily concluded that the execution time dependency of the Apriori algorithm on the number of transactions is linear. FP-growth algorithm reads the database twice and stores the database in the form of a tree in the main memory. The memory requirement of the Binary Search Tree algorithm depends only on the number of frequent number of itemsets in the given transactions. Since Binary Search Tree algorithm stores only the items needed for finding frequent 2-itemsets which are then used to form a tree in the main memory, the memory requirement of the Binary search tree algorithm does not depend on the number of transactions.

We can compare the different concepts of algorithms for mining the frequent itemsets in following manner.

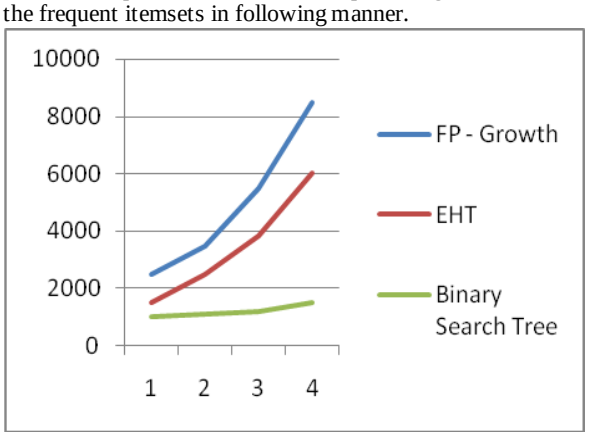

Figure -4 Comparisons of different algorithms in view of time of mining frequent itemsets

In the above graph, the comparison of the Frequent – Pattern Growth, Efficient Hashing Tree based approach and Binary Search Tree approaches are compared. It can be noticed that the Binary Search Tree approach will take very less time for mining the frequent itemsets. For mining the frequent itemsets by using FP – Growth and EHT it takes maximum time (in Seconds) where as by using Binary Search Tree Approach it will takes very less time for mining the frequent itemsets. For mining 1 – itemsets it takes very less time, in the same way for FP Growth and EHT it will average time for the mining the frequent itemsets. For mining the 2 – itemsets, FP – Growth and EHT is taking a few seconds but whereas the same 2 – itemsets can be mined very quickly by using the Binary Search Tree approach. Similarly for mining the 3 – itemsets and more 3 – itemsets, it takes very large time for mining the frequent itemsets by using FP – Growth and EHT algorithm.

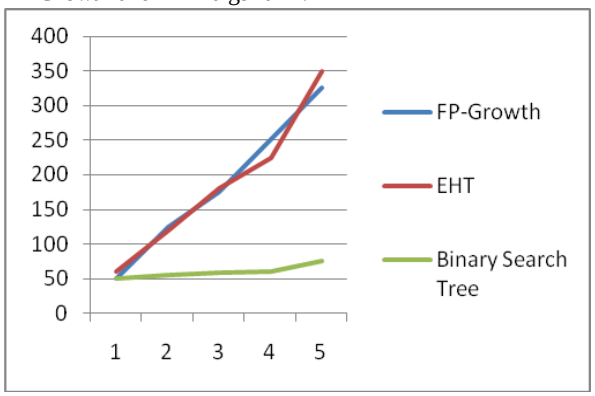

Figure -5 Comparisons of different algorithms in view of space for mining frequent itemsets

The above graph represents mining the frequent itemsets by using Frequent Pattern Growth (FP – Growth), Efficient Hashing Tree (EHT) and Binary Search Tree algorithms. Binary Search Tree approach will takes occupies very less space when we compare with Frequent Pattern Growth and Efficient Hashing Tree based algorithms. For mining the frequent itemsets, FP–Growth approach, EHT algorithm is occupying a very large space in memory for mining the frequent itemsets. The Binary Search Tree Approach is taking very less space for mining the frequent itemsets.

6. CONCLUSION

Binary Search Tree (BST) algorithms is implemented for Mining the Frequent itemsets. The Binary Search Tree algorithm is implemented for mining the 1 – itemsets, 2 – itemsets, 3 – itemsets and more than 3 – itemsets. The Binary Search Tree algorithm is useful for mining the frequent itemsets. The proposed system is implemented and tested on large data sets and it is useful for the applications wherever the itemsets are regularly utilised.

7. FUTURE WORK

The algorithm and implementations are further extended to much wider domain of problems such as e – commerce, network intrusion detection, anti – spam, E mail etc.,

For solving the duplication of itemsets, the analysis of algorithm is extended to Separate Chaining Technique in order to overcome the duplication of itemsets.

8. REFERENCES

[1] A.V. Senthil Kumar and R.S.D. Wahidabanu, "Mining Frequent Itemsets: Efficient Hashing and Tree Based Approach", International Journal of Computer Science and Software Technology (IJCSST), Vol 1, No.1, January – June 2008, pp 1 – 5.

[2] Mingjun Song and Sanguthevar Rajasekaran, Member, IEEE, "A Transaction Mapping Algorithm for Frequent Itemsets Mining", IEEE Transactions on Knowledge and Data Engineering, pp 1 – 4.

[3] Sara Ansari and Mohammad Hadi Sadreddini, "An Efficient Approach to Mining Frequent Itemsets on Data Streams", Proceedings of World Academy of Science, Engineering and Technology, Volume 37, January 2009, pp 489 – 492.

[4] Karam Gouda and Mohammed J Zaki, "Efficiently Mining Maximal Frequent Itemsets", pp 1 – 4.

[5] Ruoming Jin and Gagan Agarwal, "An Algorithm for In – Core Frequent Itemset Mining on Streaming Data", pp 1 – 4.

[6] Azzam Sleit, Wesam AlMobaideen, Aladdin H. Baarah and Adel H, Abusitta, "An Efficient Pattern Matching Algorithm", Journal of Applied Sciences 7 (18), 2691 – 2695, 2007, pp 2691 – 2693.