Advances in Computer Science and Information Technology (ACSIT)
Print ISSN: 2393-9907; Online ISSN: 2393-9915; Volume 1, Number 3; November, 2014 pp. 107-111

http://www.krishisanskriti.org/acsit.html

A Comparative Study of Various Database Scan Techniques for Frequent Pattern Generation

Jagrati Malviya[1] and Mrs. Anju Singh[2]

[1]Department of Computer Science. & Engineering BUIT, BU Bhopal
[2]Department of Computer Science & Information Technology UTD, BU Bhopal
E-mail: [1]jagratimalviya89@gmail.com, [2]asingh0123@rediffmail.com

Abstract: *There are lots of data mining tasks such as association rule, clustering, classification, regression and others. Among these tasks association rule mining is most prominent. One of the most popular approaches to find frequent item set in a given transactional dataset is Association rule mining. Frequent pattern mining is one of the most important tasks for discovering useful meaningful patterns from large collection of data. This paper explores the various database scan techniques for frequent pattern generation. These preprocessing techniques are very useful and important for reducing database scan time and space. There are lots of preprocessing techniques are available some of them discussed in this paper such as FP Tree, CP Tree, CFP Tree, K Map, Hash Tree, FP Growth Tree, COFI Tree, CT-PRO Tree. This paper also focuses on the comparative analysis of various compact database scan generation techniques on the basis of some parameters.*

Keyword: *Data mining, association Rule mining, Frequent pattern Tree, CP Tree, CFP Tree, K Map, Hash Tree, FP Growth Tree, COFI Tree, CT-PRO Tree.*

1. INTRODUCTION

In recent years amount of data in the database has increased rapidly. The increasing size of the database has led to growing interest in extraction of useful information from the bulk of data. Data mining is a technique useful for attaining useful information from vast databases. [2].Association rule mining is one of the most important data mining problems. The purpose of association rule mining is the discovery of association relationship among a set of items. The mining of association rule include two sub problems (1) finding all frequent item sets that appear more often than a minimum support threshold, and(2)generate association rules using these frequent itemsets. The first subproblem plays an important role in association rules mining [1].Frequent item set mining is one of the most important and common topic of research for association rule mining in data mining research area. A frequent item set is an item set that occurs frequently .In frequent pattern mining to check whether a item set occurs frequently or not we have a parameter called support of an item set. An item set is termed frequent if its support count is greater than the minimum support count set up initially [3].

2. LITERATURE REVIEW

FP Tree, CP-Tree, K Map, CFP Tree, Hash Tree, FP Growth Tree, COFT Tree, CT-PRO Tree

2.1 FP Tree (Frequent Pattern tree)

First scan the database and manage the items appearing in the transaction. Then all the items whose support is less than the minimum support which is user defined are considered as infrequent are deleted from consideration. All other remaining items are considered as frequent items and arrange in the sorted order of their frequency. This list is known as header table when store in table. All the respective support of the items is stored using pointers in the frequent pattern tree. Then construct the frequent pattern tree which is also known as compact tree. The sorted items according to frequency in header table are used to build the FP-tree. This needs a complete database scan. When the item insert in the tree checks if it exist earlier in tree as in same order then increment the counter of support by one which is mentioned along with each item in the tree separated by comma, otherwise add new node with 1 as a support counter. A link is maintained using pointers which same item and its entry in header table. In header table, pointer points to the first occurrence of each item [5, 6, 7] and [16].

2.2 CP Tree (Compact Pattern Tree)

First scan the database and manage the items appearing in the transaction. Then all the items whose support is less than the minimum support which is user defined are considered as infrequent are deleted from Consideration. All other remaining items are considered as frequent items and arrange in the sorted order of their frequency. This list is known as header table when store in table. All the respective support of the items is stored using pointers in the frequent pattern tree. Then construct the frequent pattern tree which is also known as compact tree [12]. The sorted items according to frequency in header table are used to build the FP-tree. This needs complete database scan. when the item insert in the tree checks if it exist

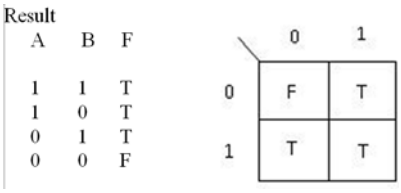
earlier in tree as in same order then increment the counter of support by one which is mentioned along with each item in the tree separated by comma, otherwise add new node with 1 as a support counter. A link is maintained using pointers which same item and its entry in header table. In header table, pointer points to the first occurrence of each item [11].

2.3. K Map

A Karnaugh map provides a pictorial method of grouping together expressions, sharing common factors thus eliminating unrelated variables. A karnaugh map reduces the need for extensive calculation by taking advantage of the humans´ pattern recognition capability. This also permits the rapid identification and elimination of potential race conditions. A Karnaugh map is composed of many grid boxes. Each grid box in a k-map corresponds to a min term or max term. Using the defined min terms, the truth table can be created as a two variables in Table 1 and Fig. 1.

Table 1: Truth table for two variables

Variables in k-map, Fig. 1: General case of a two



Result		
A	B	F
1	1	T
1	0	T
0	1	T
0	0	F

	0	1
0	F	T
1	T	T

If the number of terms n is even then matrix of size 2n=2 2n=2 is created and if the number of terms n is odd then a matrix of size2 (n 1) =2 2(n 1) =2 created. In this research study, the k-map approach on uncertain textual data to find a frequent term set which reduces the database scans and improves the efficiency and accuracy of algorithm [9, 10].

2.4. CFP Tree (Condensed FP Tree)

A new tree-based data structure, named compressed FP-Tree (CFP-Tree) is introduced. It is a variant of CT-Tree data structure that we introduced in with the following major differences: items are sorted in descending order of their frequency (instead of ascending order, as in CT-Tree) and there is a link to the next node with the same item node (while links are not present in CT-Tree) [13]. A CFP-tree constructed with minimum support thresh- old min sup can efficiently support two types of important queries related to frequent pattern mining: (1) query with minimum support constraint, for example, “find all the pat- terns with support higher than s%”, where s ≥ min sup; and (2) query with item constraint, e.g. “find all the frequent patterns containing items in I0”, where I0 is the set of items a user is interested in. These two types of queries are very common in practice, and are also essential for efficiently evaluating more complex queries [8].

2.5. Hash Tree

The hash tree is an important data structure that used to mining frequent item sets by homo Apriori algorithm. Because the hash tree can’t be built successful one time. Hash tree construction method determines the hash table size, hash functions and the no of itemsets in the leaf nodes.

Except cycle 1, in each cycle, Apriori algorithm store a hash-tree in a candidate itemset, which can help us accelerate the search speed and calculation frequency of occurrence. A hash-tree is composed by the root node, internal nodes and leaf nodes. The depth of the root node is 1. All the candidate itemsets save in leaf nodes, and each leaf node can contain several candidate itemsets. Each internal node in the tree contains a hash table, and internal node in depth of d belongs to the hash table, each point to an internal node or leaf node in depth of d+1. In the initial of building a hash tree, all the nodes are the leaf nodes. When insert a candidate k-itemset X={x1, x2, …, xk} into the hash-tree, from the root node down until find out a leaf node [19].

Hash tree is a very quick way to search an item. When there are many item sets, hash tree could be used to find out if a given item set has got required support count. If we are at a leaf–find all item sets contained in transaction. If we are at an interior node–hash on each remaining element in transaction. Root node–hash on all elements in transaction. [14]

2.6. FP Growth Tree

FP-Growth algorithm is an efficient algorithm for producing the frequent item sets without generation of candidate item sets. It based upon the divide and conquers strategy. It needs a 2 database scan for finding all frequent item sets. This approach compresses the database of frequent item sets into frequent pattern tree recursively in the same order of magnitude as the numbers of frequent patterns, then in next step divide the compressed database into set of conditional databases. [15]

FP-Growth algorithm is currently one of the fastest algorithms to mine association rules. This algorithm generates frequent itemsets and it does not create huge amount of candidate itemsets like Apriori algorithm. Before applying FP-growth algorithm on source database, the source database must be pre-processed. During the pre- processing of the database, the frequency of all items are determined by scanning the whole database, then all infrequent items are discarded from each transaction and finally the items of each transaction are sorted according to their frequency. Then from the pre-processed database a FP- tree is constructed. FP-tree is a highly compact representation of the original database, which is assumed to fit into the main memory. Algorithm 1 shows the algorithm of the construction of FP-tree [17, 18].

2.7. CT-PRO Tree

CT-PRO is also the variation of classic FP-tree algorithm. It is based upon the compact tree structure. It traverses the tree in bottom up fashion. It is based upon the non-recursive based technique. Compress tree structure is also the prefix tree in which all the items are stored in the descending order of the frequency with the field index, frequency, pointer, and item-id. In this all the items if the databases after finding the frequency of items and items whose

frequency is greater than minimum support are mapped into the index forms according to the occurrence of items in the transaction. Root of the tree is always at index '0' with maximum frequency elements. The CT-PRO uses the compact data structure known as CFP-tree i e. compact frequent pattern tree so that all the items of the transactions can be represented in the main memory.

The CT-PRO algorithm consists following basic steps:

1. In the first step all the elements from the transaction are found whose frequency is greater than the minimum user defined support.
2. Mapping the elements according to the index value.
3. Construct the CFP-tree which is known as globally CFP-tree.

Mine the Global CFP-tree by making local CFP-tree for each particular index. In this way by following the above steps can easily find the frequent item sets. The frequency of item sets greater then minimum support [14, 16].

2.8. COFI- Tree

COFI tree generation is depends upon the FP-tree however the only difference is that in COFI tree the links in FP-tree is bidirectional that allow bottom up scanning as well . The relatively small tree for each frequent item in the header table of FP-tree is built known as COFI trees. Then after pruning mine the each small tree independently which minimise the candidacy generation and no need to build he conditional sub-trees recursively. At any time only one COFI tree is present in the main memory thus in this way it overcome the limitations of classic FP-tree which can not fit into main memory and has memory problem. [14].

COFI tree is based upon the new anti-monotone property called global frequent/local non frequent property [8]. The COFI trees of all frequent items are mined independently one by one, first tree is discarded before the next COFI tree is come into picture for mining. Based on the support count and participation count frequent patterns are identified and non frequent items are discarded in the end of processing [16].

COFI tree mine the frequent item sets very easily then the FP-growth algorithm with the help of FP-tree. It saves to memory space as well as time in comparison to the FP-growth algorithm. It mines the large transactional database with minimal usage of memory. It does not produce any conditional pattern base. Only simple traversal is needed in the mining process to find all the frequent item sets. It is based upon the locally and globally frequent item sets thus easily remove the frequent item sets in the early stages and don't allow any locally non frequent elements to takes part in next stage [16].

3. COMPARISON ANALYSIS

Table 1: Comparative Analysis of Various Compact Database Generation Techniques For mining Frequent Pattern

		Parameters				
S. No	**Algorithms**	**Structure**	**Approach**	**Techniques**	**Memory Utilization**	**Databases**
1	FP Tree	Compact tree structure	Recursive	Each transaction is read and then mapped onto a path in the FP-tree. This is done until all transactions have been read.	In large database, FP tree cannot fit into the main memory.	Good for short databases
2	CP Tree	Compact prefix tree structure	Non-Recursive	It construct CP Tree according to current sort order of item list and update frequency count, then rearrange the item list according to descending order	The prefix tree based approach may suffer from the limitation of memory size, when it tries to hold whole database information.	Good for dense database
3	K Map	It provides a pictorial method of grouping together expressions.	Non Recursive	K-Map can be described as a special arrangement of a truth table.	Easily fit into main memory that's why less memory is required.	Good for small databases

4	Hash Tree	Complex Tree Base Structure	Recursive	It based on minimum Support and Confidence	Easily fit into main memory due to low memory utilization	Good for large database
5	CFP Tree	Compressed FP Tree structure	Non recursive	It constructs CFP Tree and remove redundant pattern then find frequent item on the basis of minimum support.	For large database the items can also easily fit into main memory. .	Good for dense databases.
6	FP Growth Tree	Simple tree based structure	Recursive	It constructs conditional frequent pattern tree and conditional pattern base from database which satisfy the minimum support	Low as for large database complete tree structure cannot fit into main memory	Good for dense databases
7	COFI Tree	Uses Bidirectional FP Tree structure	Non-Recursive	It constructs bidirectional FP Tree and builds the COFI-Trees for each item then mines the COFI-Tree locally for each item	Better, Fit into main memory due to mining locally in parts for the complete tree, Thus every part present in main memory	Good for dense as well as sparse databases
8	CT-PRO Tree	Uses compressed FP-Tree data structure	Non-Recursive	It constructs the compact FP-Tree through mapping into index and then mine frequent item sets according to projections index separately	Best, as Compress FP-tree structure used and mine according to projections separately thus easily fit into main memory	Good for Sparse Databases

4. CONCLUSION

Data mining has become an important field of research and has found a wide range of applications across to various areas. Mining frequent itemsets in a transaction database is critical for mining association rules. An efficient method for discovering complete frequent itemsets is very useful in solving many mining problems. Many of the techniques have been developed for database scan. Each techniques work with different conditions. The performance and efficiency of the techniques vary according to parameters. This paper described the different database techniques like as FP Tree, CP Tree, CFP Tree, K Map, Hash Tree, FP Growth Tree, COFI Tree, CT-PRO Tree. A comparative analysis of different techniques for database scans with reference to the parameters discussed in this paper. On the basis of analysis of different database scan techniques we can say that CT-PRO Tree is good for sparse and dense database and easily fit in to main memory and it works in non recursively. In CT-PRO projection concept is also apply. FP-Growth is the first successful tree base algorithm for mining the frequent item sets. We want to improve efficiency in FP tree so apply Parallel and partition technique but both techniques are based on projection.

REFERENCES

[1] Aggarwal C C, "An Introduction to uncertain data algorithm and applications", *Advances in Database Systems* 2009; 35; 1–8.

[2] Han I., Kamber M, "Data Mining concepts and Techniques", *M. K. Publishers* 2000; 335–389.

[3] Thakur R S, Jain R C, Pardasani K R, "Graph Theoretic Based Algorithm for Mining Frequent Patterns", *IEEE World Congress on Computational Intelligence Hong Kong* 2008; 629–633.

[4] Leung C K S., Hao B, "Efficient algorithms for mining constrained frequent patterns from uncertain data", *Proceedings of the 1st ACM SIGKDD Workshop on Knowledge Discovery from Uncertain Data* 2009; 9-18.

[5] Aggarwal C C., Philip S Yu, "A Framework for Clustering Uncertain Data Streams", *Data Engineering, IEEE 24th International Conference on ICDE'08*, 2008; 150-159.

[6] Chui C K., Kao B, Hung E, "Mining Frequent Item sets from Uncertain Data", *Springer-Verlag Berlin Heidelberg PAKDD'07*, 2007; 4426; 47-58.

[7] Aggarwal C C, Yan L, Wang Jianyong, Wang Jing, "Frequent pattern mining with uncertain data", *In Proc. KDD* 2009; 29-37.

[8] Leung C K S., Carmichael C L., Hao B., Efficient mining of frequent patterns from uncertain data, In Proc. IEEE ICDM Workshops'07. 2007; 489-494.

[9] Khare N, Adlakha N, Pardasani K R, "Karnaugh Map Model for Mining Association Rules in Large Databases", *International Journal of Computer and Network Security* 2009; 1(2); 16–21.

[10] Lin Y C, Hung C M, Huang Y M, "Mining Ensemble Association Rules by Karnaugh Map", *World Congress on Computer Science and Information Engineering* 2009; 320–324.

[11] Syed Khairuzzaman Tanbeer , Carson Kai-Sang Leung , on "Web Technologies and Applications" (2013).

[12] Guimei Liu Hongjun Lu Wenwu Lou, JeffreyXuYu , On "Computing, Storing and Querying Frequent Patterns".

[13] Yudho Giri Sucahyo , Raj P. Gopalan , "CT-PRO: A Bottom-Up Non Recursive Frequent Itemset Mining Algorithm Using Compressed FP-Tree Data Structure".

[14] Jong soo Park , Ming Syan chen and Phillip s. yu , “An effective hash based algorithm for mining association”.

[15] J.Han, J.Pei and Y.Yin., “Mining frequent patterns without candidate Generation” *Proceeding of ACM SIGMOD International Conference Management of Data*, 2000, pp.1-12 .

[16] Bharat Gupta, Dr. Deepak Garg , “FP Tree based algorithm analysis: FP Growth, COFI Tree and CT-PRO” , *International Journal on Computer Science and Engineering (IJCSE).*

[17] liawei Han and Micheline Kamber, "Data Mining: Concepts and Techniques", *China MachinePress*, 2001.8, pp. 158-161.

[18] liawei Han, lian Pei and Yiwen Yin, "Mining Frequent Patterns without Candidate Generation", *ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'OO)*, pp 1-12, Dallas,TX, May 2000.

[19] Chen Feng, Jia Yuanhua,Niu Zhonghai and Yi Huixin , “Research on construction method of association rule mining based on the hash tree”,*3rd International Conference on Computer and Electrical Engineering* (ICCEE 2010).