A Modern Search Technique for Frequent Itemset using FP Tree

Megha Garg
Research Scholar, Department of Computer Science & Engineering
J.C.D.I.T.M, Sirsa, Haryana, India

Krishan Kumar
Department of Computer Science & Engineering
J.C.D.I.T.M, Sirsa, Haryana, India

Rajan Garg
Department of Mechanical Engineering
SDITM, Israna, Panipat, Haryana, India

Abstract- In data mining, association rule mining becomes the major task of descriptive technique which means discovering meaningful patterns from large collections of data sets. Mining frequent itemset is very fundamental part of association rule mining. Mining for frequent patterns in transactional databases has been studied for more than a decade. Therefore, a number of ways have been proposed recently to discover approximate frequent item sets including horizontal layout based techniques, vertical layout based techniques, and projected layout based techniques. There are a few incremental algorithms are present, FUP2 and SWF that allow both addition and deletion of transactions. However, they are not efficient because they have to rescan the whole dataset at least once and memory consumption problem. They are not suitable in real time situations where transactions are added or deleted constantly and frequent patterns mining could be required at any time.

Keywords - KDD, TID, SQL, Pattem, FP-tree, Pruned etc.

I. Introduction

Data mining is to find valid, novel, potentially useful, and ultimately understandable patterns in data. With the increase in Information Technology, the size of the databases created by the organizations due to the availability of low-cost storage and the evolution in the data capturing technologies is also increasing. These organization sectors include retail, petroleum, telecommunications, utilities, manufacturing, transportation, credit cards, insurance, banking and many others, extracting the valuable data, it necessary to explore the databases completely and Decisions. KDD applications deliver measurable benefits, including reduced cost of doing business, enhanced profitability, and improved quality of service. Therefore Knowledge Discovery in Databases has become one of the most active and exciting research areas in the database community.

II. DATA MINING CONCEPT

Data mining generally involves four classes of task; **classification, clustering, regression, and association rule learning.** Data mining refers to discover knowledge in huge amounts of data. It is a scientific discipline that is concerned with analyzing observational data sets with the objective of finding unsuspected relationships and produces a summary of the data in user friendly ways that one can understand and use. Data mining as a field of study involves the merging of ideas from many domains rather than a pure discipline the four main disciplines, which are contributing to data mining include:

- Statistics: it can provide tools for measuring significance of the given data, estimating probabilities and many other tasks.
- Machine learning: it provides algorithms for inducing knowledge from given data.

- Data management and databases: since data mining deals with huge size of data, an efficient way of accessing and maintaining data is necessary.
- Artificial intelligence: it contributes to tasks involving knowledge encoding or search techniques.

Define the problem of finding the association rules from database. This introduces the basic concepts of frequent pattern mining for discovery of interesting associations and correlations between itemsets in transactional and relational database. Association rule mining can be defined formally as follows:

I= {i1, i2, i3, …, in} is a set of items
like (computer, CD, printer, papers etc.)

Let DB be a set of database transactions.
Where each transaction T is a set of items, such that $T \subseteq I$. Each transaction is associated with unique identifier, transaction identifier (TID).

Let X, Y be a set of items, an association rule has the form $X \Rightarrow Y$ where $X \cap Y = \emptyset$. X *i*s called the antecedent and Y is called the consequent of the rule where, X, Y is a set of items is called as an **itemset** or a **pattern.**

Let *freq(X)* be the number of rows containing X itemset in given database. The **support** of an itemset X is defined as fraction of all rows containing the itemset, i.e. *freq(X)/D*.
The **support** of an association rule is the support of the union of *X* and *Y, i.e.*

$$\text{Support}(X \Rightarrow Y) = (X \cup Y)/D$$

The **confidence** of an association rule is defined as the percentage of rows in *D* containing itemset X that also contain itemset *Y, i.e.*

$$\text{Confidence } (X \Rightarrow Y) = P(X/Y) = \text{support}(X \cup Y)/\text{support}(X)$$

II. PROBLEM DEFINITION

Problem of mining frequent itemsets arises in the large transactional databases when there is need to find the association rules among the transactional data for the growth of business. Many different algorithms has been proposed and developed to increase the efficiency of mining frequent itemsets. However a large number of these rules will be pruned after applying the support and confidence thresholds. Therefore, previous computations will be wasted. To avoid this problem and to improve the performance of the rule discovery algorithm, mining association rules may be decomposed into two phases:

A. Discover the large itemsets –

The sets of items that have transaction support above a predetermined minimum threshold known as frequent Itemsets.

B. Use the large itemsets –

Generate the association rules for the database that have confidence above a predetermined minimum threshold. After the large itemsets are identified, the corresponding association rules can be derived in straightforward manner. The main consideration is of First step i.e. to find the extraction of frequent itemsets.

III. PROBLEM STATEMENT

Let I = $\{i_1, i_2, i_{3\ldots\ldots\ldots,}i_n\}$ be a set of items and "n" is considered the dimensionality of the problem.

Let D be the task relevant database which consists of transactions where each transaction T is set of items such that *T € I*. A transaction T is said to contain itemset X, which is called a pattern, i.e. X *€ T € I*. A transaction is a pair which contains unique identifier TID and set of items.

A transaction T is said to be maximal frequent if its pattern length is greater than or equal to all other existing transactional patterns and also count of occurrence (support) in database is greater than or equal to specified minimum support threshold.

An itemset X is said to be frequent if its support is greater than or equal to given minimum support threshold i.e. count(X) > minimum support.

Transactional database D and minimum support threshold is given, therefore the problem is to find the complete set of frequent itemsets from Transactional type of databases to increase the business, so that relation between customers behavior can be found between various items.

IV. PROBLEM SOLUTION

New algorithm based upon the improved Apriori and the FP-tree structure is implemented in Microsoft Dot Net Framework 3.5 and SQL Server 2008 because of fast computation time and huge data storage capacity. We have developed tool as desktop and web based user friendly application.
This proposed algorithm is based upon the Apriori property i.e. all non empty subsets of the frequent itemsets are frequent. Algorithm has two procedures.

1) *Find all those maximal transactions*: which are repeating in the database equal to or greater than min user defined support also known as maximal frequent itemset.
2) *Get all nonempty subsets:* of those maximal frequent itemset as frequent according to Apriori property. Scan the database to find 1-itemset frequent elements. There may be many items found which are 1-itemset frequent but not include in maximal frequent transactions.

A. Procedure 1-

Input: Database D, minimum support
Step 1: Take a 2- dimensional array; Put the transaction into 2-dimmension array with their count of repetition.
Step 2: Arrange them in increasing order on the basis of the pattern length of each transaction.
Step 3: Find maximal transactions (k-itemset) from the array whose count is greater than or equal to the minimum support known as maximal frequent itemsets or transactions. If k-itemsets count is less than minimum support then look for k-itemsets and (k-1)-itemsets jointly for next (k-1) maximal itemsets and so on until no itemsets count found greater than minimum support. If no such transaction found then go to Procedure 2.
Step 4: Once the maximal frequent transactions found, than according to Apriori property consider all its non empty subsets are frequent.
Step 5: There are itemsets remaining which are not included in maximal frequent itemset but they are frequent. Therefore find all frequent 1-itemset and prune the database just considering only those transactions which contain frequent 1-itemset element but not include in maximal frequent transaction.
Output: some or all frequent itemsets, Pruned database D1.

B. Procedure 2-

Input: Pruned database D1, minimum support
Step 1: Find frequent 1-itemset from pruned database; delete all those items which are not 1-itemset frequent.
Step 2: Construct FP-tree for mine remaining frequent itemset by following the procedure of FP-tree algorithm as discussed above in section 2B.
Output: Remaining frequent itemsets.

V. Solution Examples

A. Procedure 1-

Input: Database D, Minimum support = 2

Step1: After scanning a database put items in 2-dimensional array with the count of repetition.

Step2: Find maximal itemset (4-itemset). Check weather its count is greater or equal to specified support, its count is 2 in our case which is equal to given support therefore this transaction is considered as maximal frequent. (If its count is less than support value then we scan k-1 and k-itemset in array for k-1 maximal itemset jointly and so on until finding all maximal frequent itemset from a array. i.e. 3-itemset and 4-itemset for checking 3- itemset maximal and so on).

2- itemset	count	3-itemset	count	4-itemset	count
{I2,I3}	2	{I1,I2, I4}	1	{I1,I2,I3, I5}	2
{I1,I3}	2	{I1,I2, I3}	1		
{I2,I3}	2	{I1,I2, I4}	1		

Table1. Itemsets in array table 1-1

Step 3*:* According to Apriori property subset of maximal frequent itemset is also considered as frequent i.e., Maximal frequent itemset: {I1, I2, I3, I5}.

All subsets are frequent (Apriori Property) i.e.

{I1, I2, and I3},{I1, I2, I5}, {I2, I3, I5}, {I2, I3}, {I2, I5}, {I1, I2}, {I1, I3}, {I1, I5}, {I3, I5}, {I1}, {I2}, {I3}.

Step 4*:* Scan the database for finding the above mined support.

Step 5*:* Find 1-itemset frequent from database, it is found that I4 which is frequent but not include in maximal frequent itemset. (There may be many items remain which are not include in maximal frequent itemsets, in our case only 1 item is there).

Prune the database by considering only transaction which contains I4 itemset.

Result*:* Some frequent itemsets ({I1, I2, I5}, {I2, I3, I5}, {I2, I3}, {I2, I5}, {I1, I2}, {I1, I3}, {I1, I5}, {I3, I5}, {I1}, {I2}, {I3}), Pruned database

TID	List of items
T2	I2,I4
T4	I1,I2,I4

Table2. Pruned database of table 1-1

B. Procedure 2-

Input: Pruned database, minimum support = 2

Step 1: Find frequent 1-itemset from pruned database with support = 2, It is found I1 is not frequent therefore delete it.

Itemset	Frequency

I2	2
I4	2
I1	1

Table3. Pruned database

Transactions become: T2: I2, I4 and T4: I2, I4.

Step 2: Construct FP-tree for remaining transaction in pruned database.

- In this case I2 and I4 have same frequency, therefore no need to arrange in L order (descending order of their frequencies).
- A new branch is created for each transaction. In this case single branch is created because of same set of transactions.
- Construct Conditional pattern base and FP-tree only for item I4.

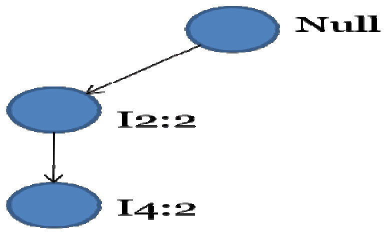

Figure1. Conditional pattern base FP tree

Item	**Pattern base**	**Conditional FP-tree**	**Frequent Item**
I4	{I2: 2}	{I2:2}	{I4,I2:2}

Table 4. Mining the FP-tree by creating conditional pattern base of table 4-4

By this procedure we can easily find unmined frequent itemset i.e. {I4, I2} which some of previous algorithm [4] are not able to find. Now we get all the frequent itemset in a particular database.

Output*:* remaining itemsets ({I4, I2})

The remaining frequent itemsets which are not mined by only maximal frequent itemsets are mined by the FP-growth procedure without generation of candidate itemsets and also in efficiently usage of memory because after pruning whole database is easily fit into main memory.

VI. CONCLUSION

In the presented approach following consideration factors for creating the new scheme are taken, which are the time and the memory consumption, these factors, are affected by the approach for finding the frequent itemsets. Work has been done to develop an algorithm which is an improvement over Apriori and FP-tree with using an approach of improved Apriori and FP-Tree algorithm for a transactional database. According to observations, the performances of the algorithms are strongly depends on the support levels and the features of the data sets (the nature and the size of the data sets). Therefore one can employed in it the scheme to guarantee the time saving and the memory in the case of sparse and dense data sets. It is found that for a

transactional database where many transaction items are repeated many times as a super set in that type of database maximal Apriori (improvement over classical Apriori) is best suited for mining frequent itemsets. The itemsets which are not included in maximal super set is treated by FP-tree for finding the remaining frequent itemsets. Thus this algorithm produces frequent itemsets completely. This approach does not produce candidate itemsets and building FP-tree only for pruned database that fit into main memory easily. Thus it saves much time and space and considered as an efficient method as proved from the results.
For both data sets the running time and memory consumption of the new scheme outperformed Apriori. Whereas the running time of the scheme performed well over the FP-growth on the collected data set at the lower support level where probability of finding maximal frequent itemsets is large and at higher lever running time is approximately same as the FP-Tree. The memory consumption is also approximately same as the FP-Tree at higher support and performed well at lower support.

REFERENCES

[1] A. Savasere, E. Omiecinski, and S. Navathe. "An efficient algorithm for mining association rules in large databases". *In Proc. International Conference. Very Large Data Bases (VLDB),* Sept. 1995, pages 432–443.

[2] Aggrawal.R, Imielinskit, Swami.A. "Mining Association Rules between Sets of Items in Large Databases". *In Proc. International Conference of the 1993 ACM SIGMOD Conference Washington DC, USA.*

[3] Agrawal.R and Srikant.R. "Fast algorithms for mining association rules". *In Proc. International Conference Very Large Data Bases (VLDB),* Sept. 1994, pages 487–499.

[4] Brin. S, Motwani. R, Ullman. J. D and S. Tsur. "Dynamic itemset counting and implication rules for market basket analysis". *In Proc. ACM-SIGMOD International Conf. Management of Data (SIGMOD),* May 1997, pages 255–264.

[5] C. Borgelt. "An Implementation of the FP- growth Algorithm". *Proc. Workshop Open Software for Data Mining,* 1–5.ACMPress, New York, NY, USA 2005.

[6] Han.J, Pei.J, and Yin. Y. "Mining frequent patterns without candidate generation". *In Proc. ACM-SIGMOD International Conf. Management of Data (SIGMOD),* 2000.

[7] Park. J. S, M.S. Chen, P.S. Yu. "An effective hash-based algorithm for mining association rules". *In Proc. ACM-SIGMOD International Conf. Management of Data (SIGMOD),* San Jose, CA, May 1995, pages 175–186.

[8] Pei.J, Han.J, Lu.H, Nishio.S. Tang. S. and Yang. D. "H-mine: Hyper-structure mining of frequent patterns in large databases". *In Proc. International Conf. Data Mining (ICDM),* November 2001.

[9] Yiwu Xie, Yutong Li, Chunli Wang, Mingyu Lu. "Data mining Concepts and Techniques" *By Jiawei Han, Micheline Kamber, Morgan Kaufmann Publishers,* 2006.

[10] S.P Latha, DR. N.Ramaraj. "Algorithm for Efficient Data Mining". *In Proc. International Conf. on IEEE International Computational Intelligence and Multimedia Applications*, 2007, pp. 66-70.

[11] Dongme Sun, Shaohua Teng, Wei Zhang, Haibin Zhu. "An Algorithm to Improve the Effectiveness of Apriori". *In Proc. International Conf. on 6th IEEE Int. Conf. on Cognitive Informatics (ICCI'07),* 2007.

[12] Q. Lan, D. Zhang, B.Wu. "A New Algorithm For Frequent Itemsets Mining Based On Apriori And FP-Tree". *In Proc. International Conf. on Global Congress on Intelligent System,* 2009, pp.360-364.

[13] W.LIU, J.CHEn, S.Qu, W.Wan. "An Improved Apriori Algorithm. *In Proc. IEEE International Conference,* 2008, pp.221-224.

[14] S.P Latha, DR. N.Ramaraj. "Agorithm for Efficient Data Mining". *In Proc. International Conf. IEEE International Computational Intelligence and Multimedia Applications*, 2007, pp. 66-70.

[15] M. El-Hajj and O. R. Zaiane. "Inverted matrix: Efficient discovery of frequent items in large datasets in the context of interactive mining". *In Proc. International Conf. on Data Mining and Knowledge Discovery (ACM SIGKDD),* August 2003.

[16] M. El-Hajj and O. R. Zaiane. "COFI-tree Mining:A New Approach to Pattern Growth with Reduced Candidacy Generation". *Proceedings of the ICDM 2003Workshop on Frequent Itemset Mining Implementations, Melbourne, Florida, USA, CEUR Workshop Proceedings,* vol. 90, pp. 112-119, 2003.

[17] Y. G. Sucahyo and R. P. Gopalan. "CT-ITL: Efficient Frequent Item Set Mining Using a Compressed Prefix Tree with Pattern Growth". *Proceedings of the 14th Australasian Database Conference, Adelaide, Australia,* 2003.