Optimum Frequent Pattern Approach for Efficient Incremental Mining on Large Databases using Map Reduce

Arpan H. Shah
Research Scholar
Department of Computer Science
Parul Institute of Technology
Vadodara, India

Pratik A. Patel
Assistant Professor
Department of Computer Science
Parul Institute of Technology Vadodara, India

ABSTRACT
Data mining defines hidden pattern in data sets and association between the patterns. In data mining, association rule mining is key technique for discovering useful patterns from large collection of data. Frequent itemset mining is a famous step of association rule mining. Frequent itemset mining is used to gather item sets after discovering association rules. Some limitations exist with the traditional association rule mining algorithms for large-scale data. As for FP-Growth algorithm, the success is limited by internal memory size because mining process is on the base of large tree-form data structure. A new traditional approach, FP-growth technique is very efficient in large amount of data. FP-Growth algorithm constructs conditional frequent pattern tree and conditional pattern based from database which satisfies the minimum support. However, FP growth algorithm requires a tree storage structure, which results in high computation time. The proposed algorithm realizes to construct Optimum pattern Tree with the node as the data item of the transaction. This rare algorithm is implemented on Hadoop to reduce the computation cost. The Hadoop environment supports for handling the large data and process them in parallel manner for better performance. The optimal frequent pattern is obtained that satisfies the minimum support and confidence value.

Keywords
Association rules, Data mining, Frequent Item set Mining, FP growth, Large database, Optimum pattern Tree

1. INTRODUCTION
Frequent item set mining is a significant part of association rule mining in data mining area. In frequent pattern mining to check whether an itemset occurs frequently or not we have a parameter called support of an itemset [1]. An itemset is called frequent if its support count is greater than the minimum support count set up initially. Association rule is encountered by X→Y where X and Y are item sets and their intersection is null i.e. X∩Y= {}. The support of an association rule is the support of the union of X and Y, i.e. XUY. X is called the head or antecedent and Y is called the tail or consequent of the rule. A number of algorithms were proposed by many researchers for making of frequent item sets, Firstly Apriori, like algorithms are proposed but because to their large number of candidate generation, database scan is more, [2] slow processing and for some cases when support threshold is low then frequent patterns generation becomes doubtful because of high memory dependency, huge search space and large I/O required in these type of algorithms [2]. After years of study, association rule mining algorithms are well established and effective in general cases [1]. However, when it comes to big data, related algorithms are not mature and need further research. In a practical situation, database is updated periodically and threshold value often changes with needs of mining. It is clearly inefficient that the whole mining process has to be restarted from the beginning every time new data is inserted into database or mining parameter is reset. Furthermore, to deal with the issues resulted from large-scale data; algorithm parallelization has become inevitable. Now, we are focusing on FP growth Algorithm on incremental database with same threshold value and database value.

2. LITERATURE ANALYSIS REGARDING FP-GROWTH STRATEGIES
Let, I = {i1, i2… in} be a set of the items, and D is regarded as a transaction database. A transaction T containing one or more items is a subset of I, where Ti ⊆ I (1≤ i ≤ n) [1]. Assume that X is an itemset, transaction T contains X if and only if X ⊆ T. Support number, denoted by count(X) , refers to the appearing times of X in the database D. Correspondingly, support degree is described as the percentage of X included in D and symbolized as support(X). If support number or support degree of X is no less than a given minimum support number , or minimum support degree s , namely threshold value, X is called frequent itemset, or large itemset, otherwise non-frequent itemset. The relationship between count(X) and support(X) is:

$$\text{Count}(X) = \text{support}(X) \times |D|$$

Where, D represents the number of the transactions in the database D. Association rules can be classified as X →Y, where X ⊂ I, Y⊂ I and X∩Y =∅ [11]. Supposing that s % of the transactions in D include X ∪Y, the support degree equals to s %. And similarly, the confidence degree turns out to be a conditional probability, that is:

$$\text{Support}(X \rightarrow Y) = p(X \cup Y)$$

$$\text{Confidence}(X \rightarrow Y) = p(Y \mid X)$$

Most researches about association rule mining focus on the first step, because of its crucial role in the algorithm efficiency. FP-Growth algorithm implements divide-and-conquer strategy. During the second scanning period, transactions are compressed into an FP-tree [12]. Meanwhile, the relationships in transactions among these items remain unchanged in the tree. After that, FP-tree would be

decomposed into conditional trees, which are used to mine frequent item sets.

- A large database is compressed into a condensed, smaller data structure, FP-tree which escapes valuable, scanning database repeatedly.
- Our FP-tree-based mining implements a pattern-fragment growth system to avoid large number of candidate sets.
- A partitioning-based, divide-and-conquer method is used to destroy the mining task into a set of less important tasks for mining limited patterns in conditional databases, which significantly decreases the search space.

3. MODIFIED FP-GROWTH STRATEGIES

There are many things which are considering FP growth approach which is weakness of frequent item sets.

The existing approaches for mining the incremental database are inefficient. The computation cost increases with re-initialization of algorithm updated set of data.

With large databases a single processor does not have enough memory to store the entire data and to process.

The basic FP growth tree requires memory space for storing the tree structure.

FP-tree is also expensive to build

The FP-Growth search for the frequent itemsets in FP-tree, which is a compact data structure built in memory and only needs a 2-time scanning. Thus, it is obvious that FP-Growth could do much better with a relatively high-powered computer or in a distributed environment. In addition to the algorithms mentioned above, incremental mining is another hotspot [1]. There are raised FUP and FUP2. The association rules would be updated when datasets were added or deleted. The UWEP came based on apriori algorithm. The early pruning techniques of UWEP greatly reduced the number of candidate itemsets, thus enhancing the efficiency of FUP [2]. There were an effective FP-tree based incremental algorithm, named FIUA and FIUA2. FIUA and FIUA2 could only be applied on the condition of dynamic threshold value and updated database respectively. Both of them failed to take into consideration the fact that several mining conditions were likely to change simultaneously in reality.

It is an Incremental Mining Binary Tree which is used to record the itemset in an efficient way. Furthermore, most tree-based incremental algorithms still had fatal shortcomings like memory dependency. To be specific, the step of producing FP-tree and other temporary trees might cause insufficient memory when processing mass data [5]. Besides, the recursive nature of FP-Growth algorithm also brought lower CPU utilization. Moreover, Map Reduce becomes more widely used to realize parallel algorithm and is a key model of processing large-scale data within a distributed environment. Another proposed technique is called PARMA, which could generate approximate association rules in MapReduce. It raised a parallelized strategy for the Apriori Algorithm. The famous algorithm proposed by Google is PFP (Parallel FP-Growth) algorithm under MapReduce framework. The only problem was that PFP could not solve the incremental problem. In view of absolute majority of MapReduce model in big data area, PIFP-Growth makes further improvements on existing PFP algorithm [3].This algorithm was implemented on Hadoop. The original Apache Hadoop was chosen as the distributed framework to implement the Apriori algorithm.

4. PROPOSED WORK METHODOLOGY

As a past year study, we have apply many technique or approach in FP growth algorithm. Now, we are proposing new or fresher technique which would handle on a large dataset called Optimum Frequent Pattern (OPT). By using this scheme, we will get optimum frequent pattern.

4.1 Optimum Pattern Tree (OPT) Approach*:*

The proposed method works on the incremental database. The new data items are added in the existing database. The mining process is done by the construction of the Optimum Frequent pattern with the nodes as the data items of the transactions. The process of adding the item set is also done in an incremental manner as the property of the optimum frequent pattern. The sub trees are created from the optimum frequent pattern for the mining procedure. The mining is the bottom-up approach where each sub tree obtained from the optimum frequent pattern is observed for the frequent pattern based on the support count of the transaction. The optimal frequent pattern is obtained that satisfies the minimum support and confidence value. The algorithm is implemented in MapReduce environment to reduce the computation cost. The MapReduce environment supports for handling the large data and process them in parallel manner for better performance. Fig. 1 illustrates the 6 steps of optimum frequent pattern Approach.

Step 1: First we will consider a transactional dataset as an input and find a each item with their support count. To give an example, if minimum support count is 3, then it will consider only those items whose give support count is greater than 3. Like, {a, b, c, d, e, and f} in Table 1.

Step2: If support count of each item is greater than minimum support count so, the item is frequent. Infrequent item will not put into this dataset

ID	Transaction
1	{a, b, c, d, f}
2	{a, c, d, f}
3	{b, c, e}
4	{b, e}
5	{a, c, d, e}

. Table 1: Transaction Data

Step 3: Sort the transactions in descending order of items support count and construct FP tree with their support count.

Step 4: After we will calculate support count of each item. Among them, select largest support count as starting vertex and join the path according to their support count.

In example, C has a largest support count. So, it will come first and join the path according to decreasing order. So, c=4, a=3, b=3, and e=3.

F is not considering because his support count is less than original support count.

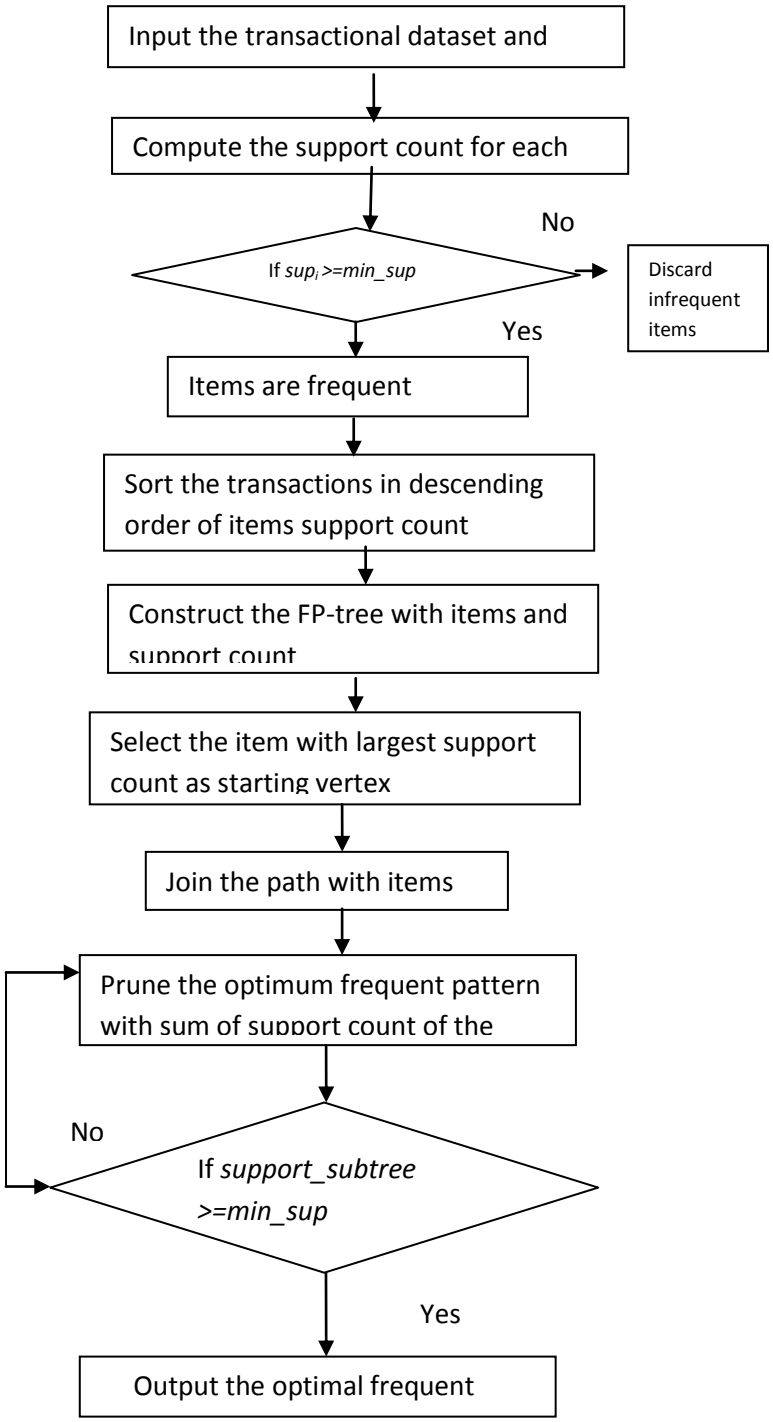

Figure 1: Workflow of the system

Step 5: We will differentiate one itemset with remain itemsets. This process consider for all itemset. In this process, we will consider largest support count among their itemset. Then it will construct optimum pattern tree.

In this example, f is not considered because, Item set f's support count is smallest. So, it is not come to the picture in OPT.

Table 2: Sorted Item according to descending order

ID	Transaction
1	{c, a, b, d, f}
2	{c, a, d, f}
3	{c, b, e}
4	{b, e}
5	{c, a, d, e}

Table 3: Largest Support Count of items

Item Set	Frequency
c	4
b	3
a	3
d	3
e	3

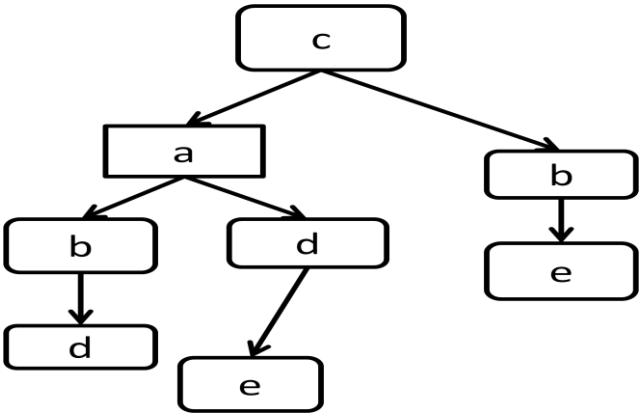

Figure 3: Optimum pattern tree from FP tree

Step 6: If sum of support count of each item set is greater than original minimum support count then output will be optimal frequent itemsets.

4.2 Optimum Pattern Tree Algorithm:

Input: The transactional dataset D, minimum support min_sup

Output: Optimal Frequent Pattern

For each transaction Ti

Compute the support of each item supi

If supi >=min_sup

Items are frequent

Else

Discard the Items that are infrequent

End For

Sort the transactions according to the item support in

Decreasing order

For each transaction Ti

Construct the FP-tree with items with their support count

End For

Select the item with largest support count as starting vertex for

Optimum Frequent Pattern

Join the path with corresponding items

For each sub-tree

Prune the Optimum Frequent Pattern with the sum of

Support count of the items

Compute the support value of the transaction

If the support_subtree >=min_sup

Output the frequent pattern

End For

5. EXPRIMENTAL RESULTS

In this section, there is a new approach called Optimum Frequent Pattern on incremental dataset with map reduce frame work including association rule mining algorithm like Apriori, FP-Growth, were compared and analyzed through experiments. All the experiments were performed on a 2.7GHz Intel i5-4210 PC with 4GB main memory. The programs are written and implemented in JAVA. The new minimum support degrees are showed below x axis, which are less than the original threshold value.

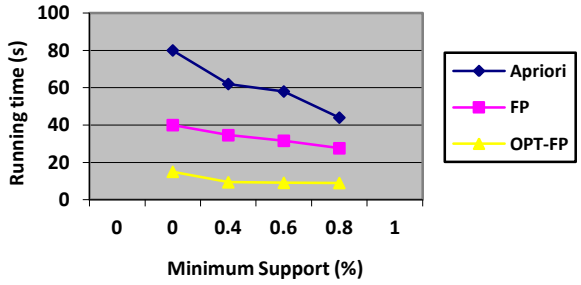

Figure 2: Experiment of Connect-4

Experiment of connect-4 dataset that database contains all legal 8-ply positions in the game of connect-4 in which neither player has won yet and in which the next move is not forced. Database contains 67557 numbers of Instances and 42 numbers of attributes and running time of Apriori, FP-Growth and Optimum Frequent Pattern-FP tree approach. We have to consider minimum support and minimum confidence as conditional parameter to get optimal frequent pattern. We have also down computation time and running time. We have to analysis and implemented in map reduce framework which will get less computation time.

6. CONCLUSION

Frequent itemset algorithms like, FP Growth constructs the conditional frequent pattern tree which satisfies the minimum support and minimum confidence. We considered the following factors for creating our new scheme called optimal pattern tree, which are the time and the memory consumption, these factors, are affected by the approach for finding the optimum frequent itemsets. Optimum pattern tree growth algorithm concludes less memory consumptions, huge amount of communication, optimal association rule. This kind of approach is finding best frequent pattern in large dataset.

7. REFERENCES

[1] G. Xiaoting Wei, Yunlong Ma, Feng Zhang, Min Liu, Weiming Shen, "Incremental FP-Growth Mining Strategy for Dynamic Threshold Value and Database Based on MapReduce", Proceedings of the 2014 IEEE 18th International Conference on Computer Supported Cooperative Work in Design, pp. 271-275

[2] Zahra Farzanyar, Nick Cercone "Efficient Mining of Frequent itemsets in Social Network Data based on MapReduce Framework" IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, 2013

[3] Arpan Shah, Pratik A. Patel, A Collaborative Approach of Frequent Item Set Mining: A Survey, IJCA, 2014, pp. 34-36

[4] Daniele Apiletti, Elena Baralis, Tania Cerquitelli, Silvia Chiusano, Luigi Grimaudo "SEARUM: a cloud-based Service for Association Rule Mining" 12th IEEE International Conference on Security and Privacy in Computing and Communications (Trust Com), 2013, pp. 1-8

[5] Le Wang, Lin Feng, Jing Zhang, Pengyu Liao, "An Efficient Algorithm of Frequent Itemsets Mining Based on Map Reduce", Journal of Information & Computational Science, May 20, 2014, pp. 2809–2816

[6] Zhuobo Rong ,DawenXia , Zili Zhang "Complex Statistical Analysis of Big Data: Implementation and Application of Apriori and FP- Growth Algorithm Based on MapReduce", IEEE 2013,pp. 968 972

[7] Palak Patel, Prof.Purnima Gandhi. "Association Rule Mining Using Improved FP-Growth Algorithm" International Journal for Technological Research in Engineering Vol-1, Issue 10, June-2014 pp. 1173-1176

[8] M SUMAN, T ANURADHA, K GOWTHAM, A RAMAKRISHNA, "A Frequent Pattern Mining Algorithm Based on Fp-Tree Structure and Apriori algorithm", International Journal of Engineering Research and Applications (IJERA) Vol-2, Issue 1, Jan-Feb 2012, pp.114-116 .

[9] Shravanth Oruganti, Qin Ding, Nasseh Tabrizi, "Exploring HADOOP as a Platform for Distributed Association Rule Mining" FUTURE COMPUTING The Fifth International Conference on Future Computational Technologies and Applications, 2013, pp. 63-67

[10] Anitha Modi, Radhika Krishnan, "An Improved Method for Frequent Itemset Mining", International Journal of Emerging Technology and Advanced Engineering ISSN 2250-2459, Volume 3, Issue 5, May 2013, pp.143-146

[11] R. Prabamanieswari, "A Combined Approach for Mining Fuzzy Frequent Itemset" International Journal of Computer Applications (0975 – 8887) International Seminar on Computer Vision (ISCV-2013), pp.1-5

[12] Jyoti Jadhav, Lata Ragha, Vijay Katkar "Incremental Frequent Pattern Mining" International Journal of Engineering and Advanced Technology (IJEAT) ISSN: 2249 – 8958, Volume-1, Issue-6, August 2012, pp. 223-228

[13] SADHANA KODALI, KAMALAKAR M, CH. GAYATRI, K.PRAVALLIKA, "An Efficient Approach for the Implementation of FP tree", International Journal of Innovative Research in Computer and Communication Engineering, Vol. 1, Issue 7, September 2013, pp. 1446-1452

[14] Jyotsana Dixit, Abha Choubey, "A Survey of Various Association Rule Mining Approaches", International Journal of Advanced Research in Computer Science and Software Engineering, Volume 4, Issue 3, March 2014

[15] Pradeep Rupayla, Kamlesh Patidar, "A Comprehensive Survey of Frequent Item Set mining Methods", IJETAE, ISSN 2250-2459, Volume 4, Issue 4, April 2014, pp. 351-353