International Journal of Applied Information Systems (IJAIS) – ISSN : 2249-0868
Foundation of Computer Science FCS, New York, USA
International Conference and Workshop on Communication, Computing and Virtualization (ICWCCV 2015) – www.ijais.org

An Efficient Approach for Parallel and Incremental Mining of Frequent Pattern in Transactional Database

Pamli Basak
Computer Engineering Department
TCET, Kandivali (E)
Mumbai

Rashmi Thakur
A.P., Computer Engineering Department
TCET, Kandivali (E)
Mumbai

ABSTRACT
In this paper, we provide an overview of parallel incremental association rule mining, which is one of the imminent ideas in the new and rapidly emerging research area of data mining. A useful tool for discovering frequently co-occurrent items is frequent itemset mining (FIM). Since its commencement, a number of significant FIM algorithms have been build up to increase mining performance. But when thedataset size is huge, both the computational cost and memory use can be toocostly. In this paper,we put frontward parallelizing the FP-Growth algorithm.We use MapReduce to execute the parallelization of FP-Growth algorithm. Henceforth, it splits the mining task into number of sub-tasks, implements these sub-tasks in parallel on nodes and then combines the results back for the final result.Experiments show that the result increases the computational speed as compared to apriori and fp-growth.

General Terms
Data Mining, Association Rule Mining, Incremental Data Mining.

Keywords
FIM, AIUA, Parallel FP-growth, Parallelized Incremental Mining, Mapreduce, Hadoop.

1. INTRODUCTION
A collection of huge and complicated datasets which is difficult to process by using traditional methods and available technologies of data mining is refer to as big data. To route big data using some analytical approach can even barely finish the work, it takes long time and the result might not be satisfactory. To solve this problem, data mining is tackle with new opportunities and challenges.

To find a frequent itemsets in large set of transaction is one of the most investigated fields of data mining. An Association rules mining and frequent itemset mining became a widely researched area, and hence in rapid speed algorithms have been presented. Association rules are if/then statements that help uncover relationships between seemingly unrelated data in a relational database or other information repository. In market basket analysis, association rules like "the customers who buy bread are most likely to buy milk" might be generated according to the processing results.

Association rule mining algorithms, after years of study are well effective and established in majority of cases. However, the traditional algorithms are not well work when it comes to big data. In a real life situation, databases are continuously updating on daily basis and support value also often changes with needs of mining. It is obviously inefficient to restart the whole mining process again whenever new data is inserted into the original database or resetting the mining parameters. This issues leads to incremental mining concept. Furthermore, algorithm parallelization has become predictable to deal with the difficulties arising from large scale data.

This paper discusses about a parallel FP-growth (PFP-growth) and parallelized incremental FP-growth (PIFP-Growth) mining approaches. The algorithm solves the incremental problem brought by the dynamic support value and database at the same time, which avoids repeated computation. MapReduce framework, implemented on Apache Hadoop is applied to achieve parallel mining.

2. BACKGROUND WORK
An elementary necessity for mining for mining association rules is mining frequent itemsets. Numerous algorithms exist for frequent itemset mining. Apriori and FP-Growth are the traditional method.

2.1 Apriori
Apriori [2] is an algorithm for frequent item set mining and association rule learning over transactional databases. It proceeds by recognizing the frequent individual items in the database and widening them to larger item sets providing those item sets appear adequately often in the database. It works with Candidate Generation and Test Approach. Candidate generation and Candidate counting and selection are thetwo steps to be perform in each iteration.

The bottleneck of Apriori: candidate generation are: it generates huge candidate sets; 10^4 frequent 1-itemset will generate 10^7 candidate 2-itemsets and to discover a frequent pattern of size 100. It also does not support incremental mining.

2.2 Fp-Growth
To overcome the problem of candidate generation FP-growth [3] is given. FP-growth is a program to find frequent item sets with the FP-growth algorithm, which corresponds to the transaction database as a prefix tree which is enhanced with links that organize the nodes into lists referring to the same item. The search iscarried out by prognostic the prefix tree, working recursively on the result, and trimming the original tree. The implementation also supports sifting for closed and maximal item sets with conditional item set repositories, although the approach used in the program differs in as far as it used top-down prefix trees rather than FP-trees. FP-growth condense a large database into a compact, Frequent-Pattern tree (FP-tree) structure with highly reduced, but complete for frequent pattern mining and avoid costly database scans. It develops an efficient, FP-tree-based frequent pattern mining method with a divide-and-conquer methodology which decomposes mining tasks into smaller ones and avoids candidate generation.

The disadvantage of this algorithm consists in the TID_set being too long, taking considerable memory space as well as computation time for intersecting the long sets. Incremental data mining is not hold by this algorithm.

3. PARALLELIZATION IN DATA MINING

An integrating association rule mining can generate more competent and correct classifiers than conventional techniques. The newly introduces MapReduce based association rule mining for pull out strong rules from large datasets. This mining is used afterward to build up a new large scale classifier. MapReduce simulator was developed to evaluate the scalability of traditional algorithms on MapReduce. The developed associative rule mining inherits the MapReduce scalability to huge datasets and to thousands of processing nodes. For the purpose of data mining to large data, parallel comprising this algorithm utilizes different approaches in rule discovery, rule from frequent itemsets, and rule clipping methods in these research fields. Parallel data mining algorithms works on distributed environment and it amplify the effectiveness of algorithm as compare to traditional algorithms.

Zhang et. al.proposed parallel FP-growth algorithm [4] on distributed machines. PFP panels computation in such a way that each machine performs an independent group of mining tasks. Pradeepa et. at. Presented Parallelized Apriori algorithm [5] to evaluate an precise and efficientclassification technique, greatly competitive and scalable compared with other conventional and associative classification methods. Procedures of FP-tree building and mining are similar to traditional FP-Growth, which run on a single computer node. This division eliminates computational reliance between machines, and thus communication between them.

PFP uses three MapReduce [6] phases to parallelize FP-growth for any transaction database DB.

- Step1: Shredding: Partitioning DB into successive small chunks.
- Step2: Parallel Counting: To count the support values of all items that are present in database using MapReduce pass.
- Step3: Grouping the items: Group list is the list of groups, where each group has unique id.
- Step4: Parallel FP-growth: One MapReduce pass is requiring. Mapper reads the group list and Reducer builds their own local FP-tree.

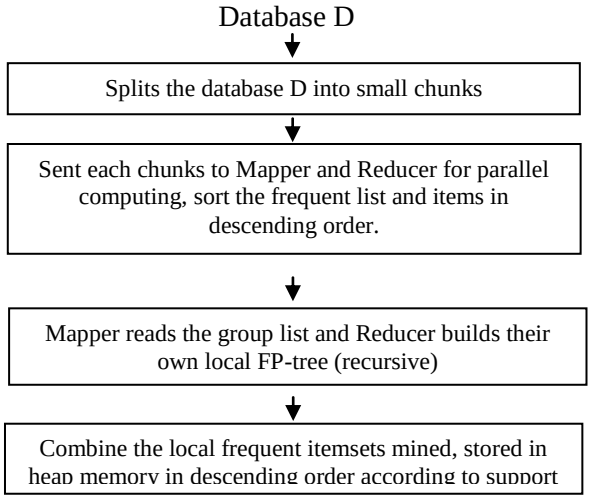

Figure1: Flowchart of Parallel FP-Growth

- Step5: Integration: Integrating the result from step4 to final result.

3.1 Mapreduce Framework

Hadoop MapReduce [6] is a software framework for easily writing applications which process vast amounts of data in-parallel on large clusters of commodity hardware in a dependable, fault-tolerant way.A MapReduce job usually splits the input data-set into independent chunks which are processed by the map tasks in a completely parallel way. The structure sorts the outputs of the mapper, which are then given as input to the reducer.Usually both the input and the output of the job are stored in a file-system. The structure takes care of scheduling tasks, supervising them and re-executes the failed tasks.The MapReduce framework consists of a one master JobTracker and one slave TaskTracker per cluster-node. The master is accountable for scheduling the jobs' component tasks on the slaves, observing them and re-executing the failed tasks. The slaves perform the tasks as instructed by the master.The programmer writes two functions a map function and areduce function each of which defines a mapping from oneset of key-value pairs to another.

4. INCREMENTAL DATA MINING

In real world, databases are updating endlessly where exactly predictable algorithms like Apriori, FP-growth perform ineffectually. If we could use the preceding analysis to incrementally mine the frequent itemset from the updated database, the mining process would become more proficient and cost of mining process would be decrease. The process of updating database continuously is Incremental Data Mining.

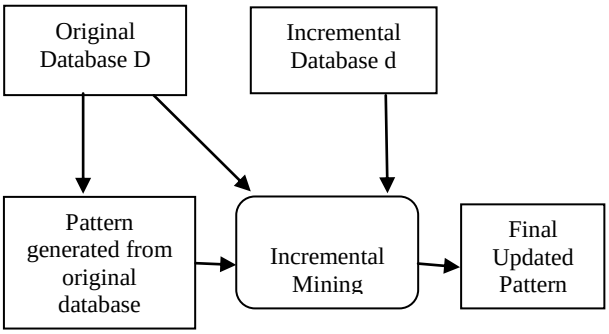

Figure 2: Overview of Incremental Data Mining

Incremental mining reduces cost of mining process by reusing the earlier mined results. Two main factors to monitor performance of incremental mining algorithms are proper memory utilization and speed of overall mining process. Incremental data mining makes the mining process more efficient in terms of time and space requirement and cost of mining process would be lessen.

Cheung et al. adopted the idea of Apriori algorithm and presented in FUP algorithm [7] to update association rules with incremental transactions, which is reusing frequent itemsets of an incremental db. But it still needs to scan the original DB multiple times and the original DB is always very large.To improve the efficiency of FUP algorithm, T.Garib et. Al. presented FIM algorithm[8], which require only one scan for the original DB and reduce the generation of candidates.

To get better efficiency of IUA, Chen et. Al. proposed a more improved algorithm AIUA [9]. He presented new function, which joins the corresponding frequent item sets and evades the iteration to generate many useless candidates.FIM_AIUA [10] algorithm proposed by Yuchen et. Al., which expands FIM and

AIUA algorithm to update association rules with incremental transactions and with least support changes simultaneously. Incremental Updating Algorithm also faces the problem of several scan of original DB and it also requires numerous iterations to generate many futile candidates. Thus it is time-consuming and incompetent. FIM_AIUA algorithm also perk up the effectiveness and accurate the mistake of My_IUA algorithm.

In realistic application, user always adjusts the minimum support when new transactions are inserted into database. FIM_AIUA is suitable for updating association rules with incremental transactions and minimum threshold value changes concurrently.

5. PARALLEL INCREMENTAL FP-GROWTH

Parallel Incremental data mining combines the features of both parallelism and incremental mining to improves the efficiency. After long study, association rule mining algorithms are well recognized and effectual in wide-ranging cases. However, when it comes to large data, allied algorithms are not mature and need advance research. In actual situation, database is updated periodically and threshold value often changes with needs of mining. It is clearly incompetent that the whole mining process has to be revived from the beginning every time new data is addedinto database or mining parameter is reset. Furthermore, to deal with the issues resulted from large-scale data, algorithm parallelization has become certain. Wei et. Al. proposed parallelized incremental FP-Growth mining strategy [11] successfully solves the incremental issue brought by the dynamic threshold value and database at the same time, which shuns repeated computation. This parallel mining strategy based on MapReduce framework is implemented on Apache Hadoop [12].

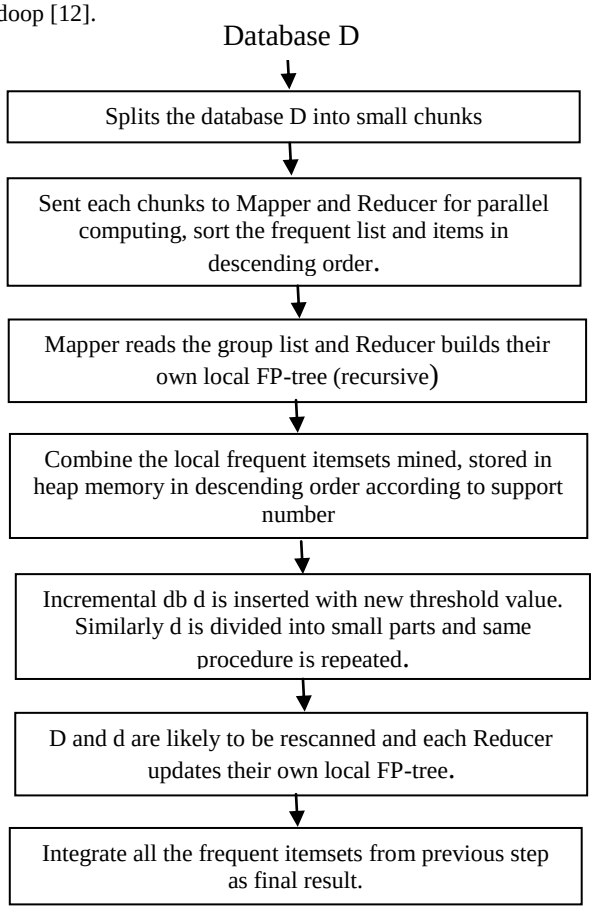

Figure3: Flowchart of Parallel Incremental FP-Growth

On the basis of the structure of PIFP-Growth algorithm [11] and MapReduce model, four MapReduce phases are used in the whole process.

- Stage 1: InputSplit, a technique defined by Google, divide the database into small parts; these parts are sent to Mapper and Reducer to finish the parallel counting of support number; the counting outcomes are integrated into a frequent list and items are sorted in descending order; all the items on the frequent list are divided into groups.
- Stage 2: It takes one Map Reduce pass to obtain the local frequent item sets. Mapper reads the group list and allocate the related transactions to different computer node according to the group nodes; then each Reducer constructs their own local FP-tree; during the recursive progress, the frequent item sets are pull out.
- Stage 3: The integration stage is to combine the local frequent item sets mined.
- Stage 4: During this stage, d is added into database and the support value is reset as s′. If there exist new items on the frequent list equal or greater than s′, these items have to be clustered as well.
- Stage 5: In this stage, new added datasets d is taken into consideration. The datasets go through a Map Reduce pass like stage 1.
- Stage 6: According to the new frequent list, database D and d are likely to be rescanned. Mapper allocates transactions related to new frequent items to corresponding cluster; each Reducer updates their own local FP-tree; new frequent items are mined in updated FP-trees.
- Stage 7: The last step is to amalgamate all the frequent item sets from stage 6 as the final result.

6. EXPERIMENTAL RESULTS

In this segment, PFP-Growth and two main association rule mining algorithm, including Apriori, FP-Growth, were compared and examined through experiments.

All the experiments were performed on 3.10GHz Intel i5 processor with 4GB RAM. The program is written in JAVA and executed on Hadoop.

Table 1 shows 3 datasets used in test for association rule mining.

Table 1. Datasets for Experiment

Dataset	Size(MB)	Transaction	Items	Database
T10I4D 100K	3.93	100,000	870	8000
Retail	4.07	105,068	964	9200
Kosarak	30.50	990,002	41,270	90000

The new minimum support degrees are showed below x-axis. Figure 4. Experiment of T10I4D100K, Figure 3.Experiment of Retail and Figure 4. Experiment of kosarak give the total running time of Apriori, FP-Growth and PFP-Growth.

From the results we can say that PFP-Growth cost the smallest amount of time, comparing with other two algorithms. When data size is small, divergence is not obvious.

On the other hand, as the amount of data boost, PFP-Growth shows great advantage in total running time over other two algorithms, particularly when threshold value is low.

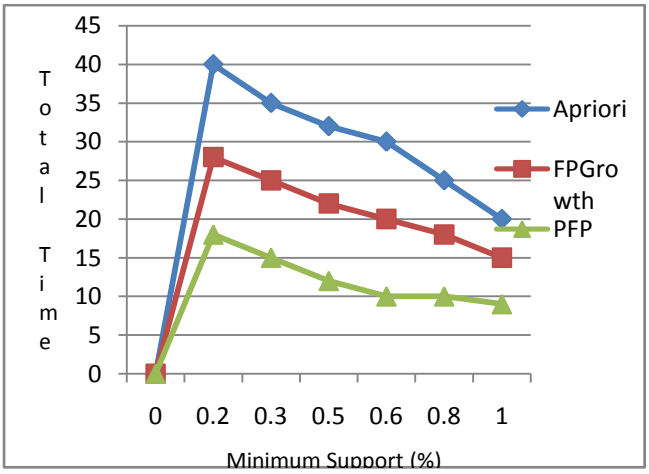

Figure4: Experiment of T10I4D100K

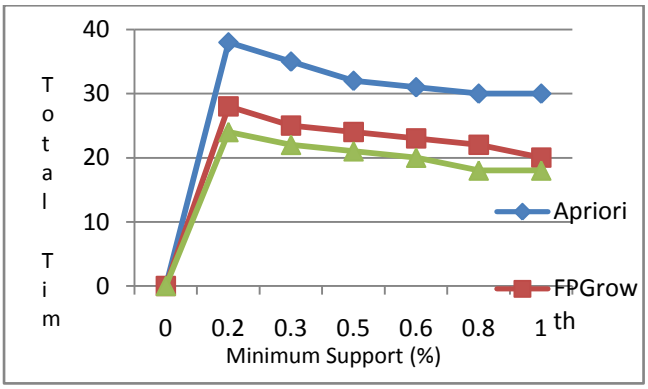

Figure5: Experiment of Retail

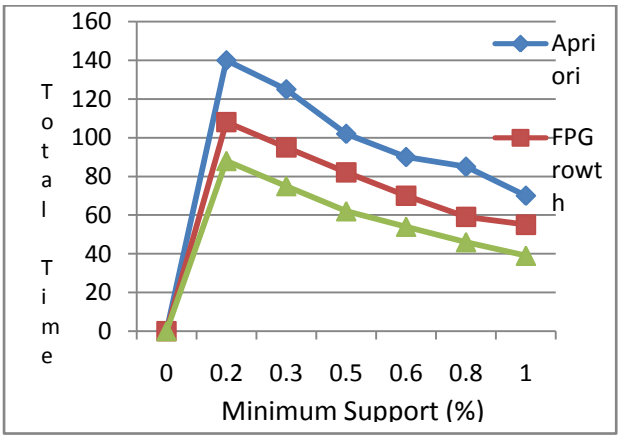

Figure6: Experiment of Kosarak

7. CONCLUSION

Apriori and FP-growth; traditional algorithms and data mining methods have faced limitations when dealing with large sized data. For occasion, Apriori algorithm needs to scan the data from external storage frequently so as to get the frequent itemsets, which takesheavy I/O load with decreased performance. InFP-growth algorithm, TID_set being too long, taking considerable memory space as well as computation time for intersecting the long sets.This paper discusses Parallel FP-Growth mining tactic and Parallelized incremental FP-growth mining tactic, which are parallelized under the MapReduce framework.

Experimental results of Parallel FP-growthspecify that this algorithm is effective in reducing time byeliminating duplicated work and spurious items. Also, it minimize the response time to a query for the set of frequent.

8. ACKNOWLEDGEMENT

I would like to express profound gratitude to**Dr. R. R. Sedamkar** for his invaluable support,cooperation, constant encouragement, supervision and useful suggestions throughout the work. His moral support continuous guidance enables me to complete my work successfully.

9. REFERENCES

[1] Jaiwei Han and Micheline Kamber, *Data Mining, Concepts and Techniques*: An imprint of Elsevier, Second Edition, 2006.

[2] R.Agrawal and R.Srikant, "Fast algorithms for mining association rules," *in Int.Conf.VLDB*, pages 487-499, September 1994.

[3] Jaiwei Han, Jian Pei and Yiwen Yin- "Mining Frequent Patterns without Candidate Generation," *in Int.Conf.ACM-SIGMOID*. pages 1-12, June 2000.

[4] H. Li, Y. Wang, D. Zhang, M. Zhang and E. Chang, PFP: Parallel FP-Growth for Query Recommendation, *Proceedings of the 2008 ACM Conference on Recommender Systems*, 2008, pages 107-114.

[5] A. Pradeepa, and A. S. Thanamani, PARALLELIZED COMPRISING FOR APRIORI ALGORITHM USING MAPREDUCE FRAMEWORK, *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 2(11), 2013, pp. 4365-4368.

[6] J. Dean and S. Ghemawat, -MapReduce: simplified data processing on large clusters, *Communications of the ACM*, vol. 51, Jan. 2008, pp. 107–113.

[7] D.W.Cheung, J.Han, V.T.Ng, and C.Y.Wong, "Maintence of discovered association rules in large databases:an incremental updating technique. " *in Int.Conf.on Data Engineering*, pages 106-114, February 1996.

[8] T.F.Garib, M.Taha, and H.Nassar, "An efficient technique for incremental updating of association rules." *International Journal of hybrid Intelligent Systems*, pages 45-53, May 2008.

[9] An Hongmei, Chen Ping- "Study of Incremental Updating Algorithm for Association Rules"- Atlantis Press, Paris, France,2012.

[10] Li Sun, Yuchen Cai, Jiyun Li, Juntao Lv- "An Efficient Algorithm for updating Association Rules with Incremental Transactions and Minimum support Changes Simultaneously"- *IEEE Third Global Congress on Intelligent Systems*, 2012.

[11] X. Wei, Y. Ma, F. Zhang, M. Liu, W. 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*, May 2014, pages 271-276.

[12] S. Kurazumi, T. Tsumura, S. Saito, and H. Matsuo, Dynamic processing slots scheduling for I/O intensive jobs of Hadoop MapReduce, *Proceedings of the 2012 3rd International Conference on Networking and Computing*, 2012, pp. 288-292.