International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 3 Issue 7, July - 2014

An Efficient Approach for Frequent Pattern Mining Using Parallel Computing.

Prachi S. Bhokare
Computer Engineering Dept
Thakur College of Engg
Mumbai, India

Dr. Rekha Sharma
Computer Engineering Dept
Thakur College of Engg
Mumbai, India

Mr. Harshal Dalvi
Information Technology Dept
D. J. Sanghavi College of Engg
Mumbai, India

***Abstract*-The highly researchable filed of data mining is nothing but frequent itemset mining. Apriori and FP Growth algorithms are most traditional algorithms for it. To develop fast and efficient algorithm for frequent pattern mining is the most challenging task. In this paper, we are improving the efficiency of Apriori algorithm using Hadoop concept and techniques to handle big data problem.**

Keywords- Association rules, Apriori, Hadoop, KDD, MapReduce.

I. INTRODUCTION

Data mining is also known as **K**$_{\text{nowledge}}$ **D**$_{\text{iscovery}}$ in **D**$_{\text{atabases}}$, which automatically extract useful hidden data from the dataset which is gigantic and ambiguous. In the age of big data, complex statistical analysis such as market basket analysis and data association analysis has become an urgent need for enterprises with an effective way to analyze the large scale data deeply. So, KDD has drawn attention from industry as well as research communities. Association rule is most essential concept of data mining which can discover the relationship between itemset from the database that often has hundreds of attributes and records, contains complex relationship between data tables, and remains a time-consuming process.

A traditional mining algorithm faces many problems while handling massive data. With rapid growth of internet, internet of things and sensor network, data are increasing exponentially. There are many proficient algorithms are proposed which are based on two main algorithms: Apriori and Frequent pattern growth for association rule mining in the large scale data will cause to "high memory consumption, low computing performance, poor scalability and reliability" and other problems. In this paper, we implemented an efficient approach for frequent pattern mining using parallel Apriori algorithm which is based on MapReduce. The rest of the paper is organized as follows. Section II. Frequent pattern mining. Section III. Introduction to Hadoop Section IV. Implementation for Parallel Apriori algorithm Section V. Performance evaluation Section VI. Summarizes the paper.

II. FREQUENT PATTERN MINING

One of central themes of data mining is association rule mining which was first invented by the Agrawal [1], [2]. This association rule-mining task further classified into two steps:

Step1. Find out the frequent itemset which can fulfill the minimum support criteria of user specified minimum support.

Step2. Generate the rule by applying the user define minimum confidence and these frequent itemsets.

The Support and Confidence are the common factors to measure the strength of the association rule.The rules that have a support and confidence greater than thresholds are called as strong rules. There are many types of association rule. 1. Generalized Association Rule. 2. Quantitative Association Rule. 3. Interval Association Rule. 4. Sequential pattern mining. 5. Maximal Association Rule.

Many algorithms are there to generate association rules. Some of listed below:

- Apriori Algorithm
- FP-Growth Algorithm
- Eclat Algorithm
- LORE etc.

Ariori is one of the traditional algorithms, which is a seminal algorithm proposed by R.Agrawal and R.Srikant in 1994 for mining frequent itemsets for Boolean association

rule [1], [2]. Even though it is most simplex algorithm but it is costly in terms of space and time complexity since requires repeated scan of database. With rapid growth of internet, internet of things and sensor network, data are increasing exponentially which often cause non negligence problems of memory overflow and huge delay in communication.

Researcher proposes many algorithms to overcome above problems but the performance of every algorithm is different on different databases [3], [9].

III. INTRODUCTION TO HADOOP

Now a day data deluge is the most frequent challenge faces by the many customers. In 2010 digital data universe was 1.2 Zetta Byte. According to the scientist, in 2020 it will be around 35 Zetta Byte. Actual problem is not only it's size but the most problematic thing is that 90% of digital universe is unstructured. Hadoop provide the solution for this problem. Using Hadoop we can handle many problems related to the big data mining. It also provides the solution for increasing the efficiency of many sequential algorithms. Hadoop adaption in industry increases day by day. It is one of the scalable fault tolerant distributed systems for data storage and processing [4], [5].
There are two components of Hadoop:
1. HDFS
2. MapReduce.

Map Reduce provides fault tolerant distributed processing. It is a recent programming framework for processing and generating large datasets. The main important feature of Map Reduce is that it can operate on both structured and unstructured data. The two primitive functions that MapReduce provided are: Map and Reduce [12]. The Map function is applied on the input data and produces a list of intermediate <key, value> pairs.

Map :: (Key1) ⟶ list (key2, value2)

The Reduce function is then applied to the list of intermediate values that has the same key. It typically performs some form of merging operation, and produces zero or more output <key, value> pairs.

Reduce :: (Key2, list(value2)) ⟶ list (value2)

A key benefit in MapReduce is that the programmer does not need to deal with the complicated code parallelism, and focuses on the required computation. The MapReduce runtime is responsible for parallelization, distributed computing and concurrency control.

IV. IMPLEMENTATION OF PARALLEL– APRIORI

In this section, we are going to redesign the Apriori algorithm to work parallel using the MapReduce. Map/Reduce paradigm, a clearly parallel system which can take full advantage of machine. This new design improves the efficiency of sequential Apriori in many aspects. Here we are using Hadoop environment, with the machine confirmation is CPU Pentium I-7, Ram 4.0 GB, Hard Disk 1.0 TB.

The input data set is then divided into the parts using classification techniques according to the items. Each of which is then assign to the Map function. Each Map function then process, localize input data to find out candidate 1 item set in form of <Key, Value > pair. Here the Key is individual item which is present in input dataset and Value is nothing but the number of occurrences of it. After computation of this, output of each Map function is passes to the data aggregation layer. Data aggregation layer combines all the data according to the key and generate global <Key, Value> pairs. In this stage, all the intermediate results are stored in the temporary file. Later data stored on temporary file again split and passes to the Reduce function. Reduce function prunes the Key items which don't satisfy the minimum support criteria which are previously mention by the user. As per the property of an Apriori Algorithm, all nonempty subset of a non-frequent itemsetmust also be a non-frequent.The output of this stage is further provided as an input to the next iteration. This algorithm stops when there is no output file is present. The parallel Apriori helps in reducing the size of candidate itemsetas, it removes those itemsets whose subset were absent in the previous iteration's output file. Once the frequent itemsets are generated, association rules are developed.

Following figure shows the single iteration data flow of MapReduce.

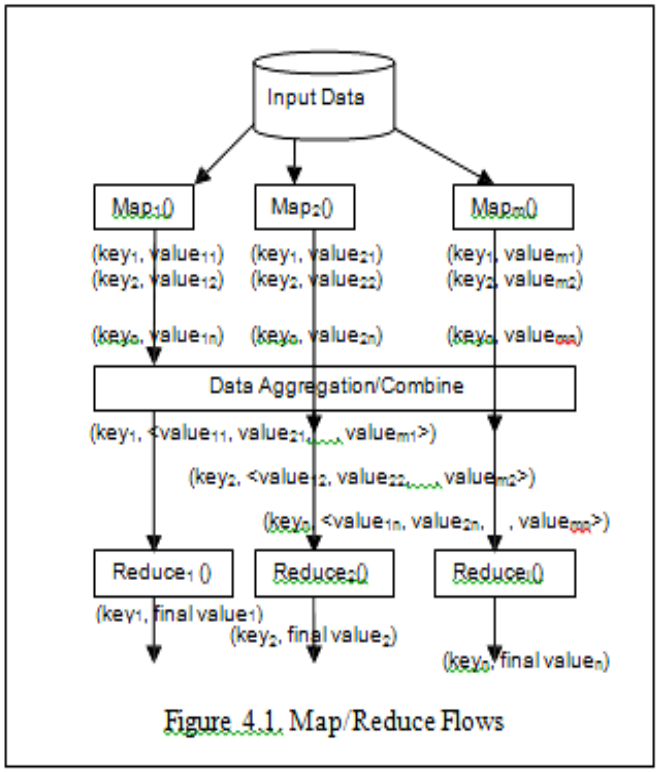


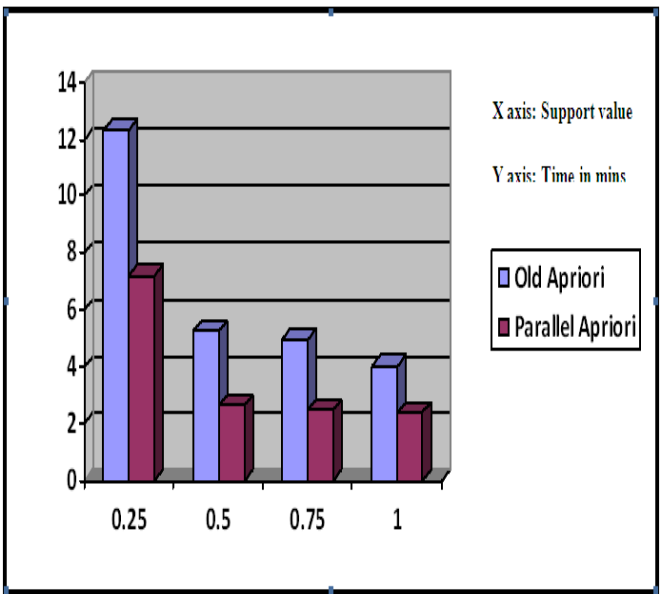
Figure 4.1 Map/Reduce Flows

V. PERFORMANCE EVALUATION

To evaluate the performance of the algorithms over a large range of data, the result of varying minimum support and number of transactions are shown in table given bellow.

Table 5.1 Variation of execution time with minimum support

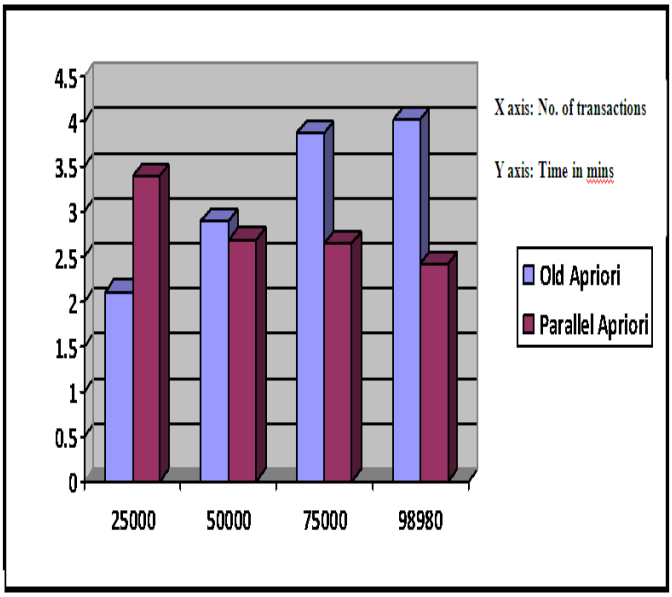
support	Old Apriori	Parallel Apriori
0.25	12.33	7.23
0.50	5.24	2.665
0.75	4.93	2.484
1	4.01	2.41



Graph 5.1 Comparison of Old Apriori with Parallel Apriori

In above graph, we are comparing old AprioriAlgorithm with Parallel Apriori Algorithm. Result shows that, Parallel Apriori performs well as compare to old Apriori algorithm. It will show nearby 50% improvement in performance.

Table 5.2 Comparison of old Apriori and Parallel Apriori with different number of transactions.



Graph 5.2 Comparison of old Apriori and parallel Apriori with different number of transactions.

In this, Comparison of old Apriori and parallel Apriori with different number of transactions i.e. 25000, 50000, 75000, 98980 is shown. As we all knows that old Apriori algorithm performs well for small number of transactions. As soon as transactions get increased its performance gets decreased.

No. of Transactions	Old Apriori	Parallel Apriori
25000	2.10	3.39
50000	2.89	2.67
75000	3.86	2.64
98980	4.01	2.41

From above graph, we gets cleared idea about behavior of both the algorithms. When we execute 25000 transactions on old Apriori and Parallel Apriori, performance of the old Apriori is good as compare to Parallel Apriori. But as soon as the number of transactions gets increased the performance of Parallel Apriori gets increase and performance of old Apriori gets decreased.

We have performed all these experiments on single high performance, I7 Processor having 4 GB ram. From the above results, it is seen that Parallel Apriori algorithm perform well as compare to the old Apriori.

VI. SUMMARY

In this paper, we mainly try to resolve the problems which are faced by the Sequential Apriori algorithm using parallel computing. For it, here we used improved MapReduce function of Hadoop. Map/Reduce paradigm is a clearly parallel system which takes full advantage of each machine with some parallel computing environment. It also have inbuilt ability to handle many difficult problems, including parallelization, concurrency control, network communication and fault tolerance [13].

VII. REFERENCES

[1] Agrawal R, Imielinski T, Swami A, "Mining association rules between sets of items in large databases". In: Proc. of the l993ACM on Management of Data, Washington, D.C, May 1993. 207-216

[2] R. Agrawal and R. Srikant, Fast algorithms for mining association rules, in Proc. 20th Int. Conf. Very Large Data Bases, VLDB , edited by J.B. Bocca, M. Jarke, and C. Zaniolo, Morgan Kaufmann 12(1994) 487–499.

[3] W. Fang, K. K. Lau, M. Lu, X. Xiao, C. K. Lam, Y. Yang, B.He, Q.Luo, P. V. Sander, and K. Yang.Parallel data mining on graphics processors. Technical Report 07, The Hong Kong University of Science & Technology, 2008

[4] http://docs.amazonwebservices.com/ElasticMapReduce/latest/DeveloperGuide/Introduct ion_EMRArch.html.

[5] Mr.Kiran C. Kulkarni1, Mr.R.S.Jagale2, Prof.S.M.Rokade3, A Survey on Apriori algorithm using MapReduce Technique, International Journal of Emerging Technology and Advanced Engineering Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008Certified Journal,, Volume 3, Special Issue 4, March 2013)

[6] Data Mining Using Clouds: An Experimental Implementation of Apriori over MapReduce, Juan Li1, Pallavi Roy1, Samee U. Khan1, Lizhe Wang2, Yan Bai3

[7] Available from: http://hadoop.apache.org/hdfs/ [Last cited on 2011 Oct 15].

[8] Karim M, Hossain M, Rashid M, Jeong BS, Choi HJ. A MapReduce Framework for Mining Maximal Contiguous Frequent Patterns in Large DNA Sequence Datasets. IETE Tech Rev 2012;29:162-8

[9] Han Jiawei, KamberMiceline. Fan Ming, MengXiaofeng translation, "Data mining concepts and technologies". Beijing: Machinery Industry Press. 2001

[10] R. Agrawal and J.C. Shafer , "Parallel Mining of Association Rules,"IEEE Tran. Knowledge and Data Eng. , vol. 8, no. 6, 1996,pp.962-96

[11] Zhuobo Rong Sch. of Comput. & Inf. Sci., Southwest Univ., Chongqing, China Dawen Xia ; Zili Zhang "Complex statistical analysis of big data: Implementation and application of Apriori and FP-Growth algorithm based on MapReduce".

[12] MapReduce Tutorial http://pages.cs.wisc.edu/~gibson/mapReduceTutorial.html

[13] Map Reduce Programming Tutorial - Manjrasoft *www.manjrasoft.com/download/MapReduceModel.pdf*