International Journal of Engineering Research in Computer Science and Engineering (IJERCSE) Vol 2, Issue 8, August 2015

Distributed Market Basket analysis using MapReduce framework on Hadoop

[1]Ms.Shubhangi Gawde, [2] Prof.Vipul Dalal
[1]P.G. Scholar,ComputerEngg. Department.A.R.M.I.E.T.Sapgaon,Thane, India
[2]Guide, Computer Engg.Department.,V.I.T.,Dadar, India.
[1]shubhangi.gawde@mmbgit.org

Abstract- **Distributed Market Basket analysis is immensely important in everyday's business decision making process. Due to its capability of mining customers purchase patterns by discovering what items customer is buying frequently and together. Distributed Market Basket analysis is capable of solving the problem of ever increasing transactional data. MapReduce framework on Hadoop is use to generate the complete set of maximum frequent itemsets .since the frequent itemsets for a particular customer discloses his or her purchase pattern.**

Key words- **MapReduce, frequent itemset, distributed computing, Apriori.**

I. INTRODUCTION

Data mining is the efficient discovery of previously unknown patterns in large datasets. It has attracted a lot of attention from both research and commercial communities for finding interesting information hidden in large datasets. One of the most important areas of data mining is association rule mining; its task is to find all subsets of items which frequently occur and the relationship between them by using two main steps: finding frequent itemsets and generation association rule.

Finding frequent itemsets is one of the most important fields of data mining. Apriori algorithm is the most established algorithm for finding frequent itemsets from a transactional dataset; however it needs to scan the dataset many times and to generate many candidate itemsets. Unfortunately, when the dataset size is huge, both memory use and computational cost can still be very expensive. In addition, single processors memory and CPU resources are very limited, which make the algorithm performance inefficient. Parallel and distributed computing is effective strategies or accelerating algorithms performance.

Furthermore;because of exponential growth of worldwide information, enterprises have to deal with an ever growing amount of data .As these data grow past hundreds of gigabytes towards terabytes or more, it become nearly impossible to process them on a single sequential machine . the solution for the above problem is parallel and distributed computing.

Parallel and distributed computing offer a potential solution for the above problems if the efficient and scalable parallel and distributed algorithm can be implemented. Such easy and Efficientimplementation can be achieved by using Hadoop-MapReduce model which is a programming model for easily and efficiently writing applications that process vast amount of data in parallel on large clusters of commodity hardware in a reliable, fault –tolerant manner.

II. RELATED WORK

To find accurately frequent itemsets from large database is challenging task in data mining. There are many traditional algorithms are available for it such as Apriori, Eclat, and FP - Growth algorithm. here are some technique have implemented to finding out frequent item sets from database using Apriori and each having its own advantage and drawbacks. Google's MapReduce is presented in 2004 but it has also some limitations [12]

A. Finding frequent itemsets using Apriori algorithm:

The most famous is the Apriori algorithm[1][5][6] which has been brought in 1993 by Agrawal which uses association rule mining. Association rule usually required to satisfy user specified minimum support and user specified minimum confidence at the same time. Association rule generation is usually split up into two separate steps:

First, minimum support is applied to find all frequent itemsets in a database. Second, these frequent itemset and minimum confidence constraint are used to form rules.

Advantage of this algorithm is it is easy to find frequent itemset if database is small but it has two deadly bottlenecks. First, it needs great I/O load when frequently scans database and Second, it may produce overfull candidates of frequent item-sets.

B. Finding frequent itemsets using partitioning as well as Apriori algorithm:

Dr.KeranaHanirex and Dr.M.A.Dorai Ragaswamy [2] proposed efficient algorithm for mining frequent itemset using clustering techniques. this algorithm finds the present itemset by partitioning the database transactions into clusters and after clustering it finds the frequent itemsets with the transactions in the cluster directly using improved Apriori algorithm which further reduces the number of scans in the database as well as easy to manage and available easily, hence improve the efficiency as well as new algorithm better than the Apriori in the space complexity but again it uses Apriori algorithm hence efficiency not increases as much as required.

C. Finding frequent itemsets using improved Apriori algorithm based on matrix:

Feng WANG and Yong-hua[3]proposed improved Apriori algorithm based on the matrix to solve the bottleneck of Apriori algorithm, they introduce an improved algorithm based on matrix[8].it uses matrix effectively indicate the affairs in the database and uses "AND operation " to deal with the matrix to produce the largest frequent itemsets and others. The algorithm based on matrix don't scan database frequently, which reduce the spending of I/O .so the new algorithm is better than the Apriori in the time complexity.

D. Finding frequent itemsets using improved Apriori algorithm based on CGAR:

Wael A.AlZoubi Abu Bakar KhairuddinOmar [9] proposed scalable and efficient method for mining association rules using CGAR. which depends on the concepts of clustering and graph data structure ,this new algorithm will be named clustering and graph based rule mining(CGAR).the CGAR method is create a cluster table by scanning the database only once , and then clustering the transactions in to cluster table according to their length 'the frequent 1-itemsets will be extracted directly by scanning the cluster table .to obtain frequent k itemsets and build directed graph for each cluster this approach reduces main memory requirement since it considers only a small cluster at a time but it will not consider large cluster.

E. Finding frequent itemsets using an efficient mining of transactional data using Graph based Technique:

Wael Ahmad AlZoubiKhairuddinOmar, Aurelia Abu Bakar[10] propsed a graph based approach (DGARM) to generate Boolean association rules from a large database of customer transactions. This approach scans the database once to construct an association graph and traverses the graph to generate all large itemsets. Practical evaluations show that the algorithm need to make multiple passes over the database.

F. Finding frequentitemset using Improved Apriori algorithm based on pruning optimization and transaction reduction:

S.cai and C Zhu [11] proposed basic ideas and the shortcomings of Apriori algorithm, studies the current major improvement strategies of it. In order to solve the low performance and efficiency of the algorithm caused by its generating lots of candidate sets and scanning the transaction database repeatedly, it studies the pruning optimization and transaction reduction strategies, and on this basis, the improved Apriori algorithm based on pruning optimization and transaction reduction is put forward.

III. PROBLEM DEFINITION

Google's MapReduce presented in 2004 .In this work it first sorts and converts the transaction in <key,value> pair ,then it store back it to node. It increases time for execution. Also this approach does not remove null transaction due to which unnecessary increase in computations is found. [12]

The most famous is the Apriori algorithm [1] which has been brought in 1993 by Agrawal; et al. has two deadly bottlenecks [2].

1) It needs great I/O load when frequently scans database.
2) It may produce overfull candidates of frequent itemsets.

Our idea is very simple .Our approach to solve these problem is described in IV and V section of this paper.

proposed system

To reduce the overfull candidate created during database scans it is proposed that First, input file is divided in to small blocks of 64MB using HDFS.Then in Second phase improved pruning optimization and transaction reduction method [11] applied on each cluster.

A. Project Requirement

Desktop PC Memory 1-4 GB Storage 160 GB (depending on application).	Name Node
Desktop PC Memory -4 GB Storage 160 GB (depending on application).	Data Node

Table1: Hardware Requirement

Table1 shows minimum requirement to set up Hadoop cluster requiring at least one name node scalable to 'n' data nodes. We can reconfigure it for better reliability.

a. Ubuntu (64-bit):

We have used Ubuntu because it is perfect match with Hadoop and both are open industry standard .Ubuntu is most common platform used to set up Hadoop cluster in cloud environment.

b. Hadoop

Hadoop platform was designed to solve the problem of Big data which has mixture of complex and structured data. Hadoop uses Hadoop Distributed File system is suitable for our project it enables terabytes of storage and retrieval of information ,analysis of data ,distributing the data and task ,creating Mappers for each data node and carrying out the job of collecting the result and final analysis of result set using Reducer

c. Hadoop Distributed file system (HDFS)

HDFS has master slave architecture.HDFS cluster consist of single Name node is a master system that manages the file system name space and regulates the access to file by clients. In addition there are number of Data nodes which manage storage attached to the nodes that they run on. HDFS internally splits the file in 64 MB blocks these blocks are saved as set data nodes.

proposed approach

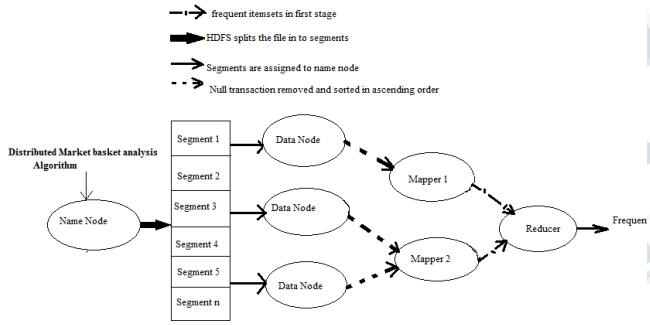

Figure 1: workflow of system

1. Name node splits transactional data into segments when stored on the HDFS.
2. We used MapReduce framework on Hadoop.
3. Name node assigns task to idle data nodes for map operation
4. When segment is assigned to data node first it scans assigned segment(s) as <TID, Itemset> pairs, and it will delete null transactions.
5. Null transaction are those in which no item is buy.
6. Then remaining transactional data is sorted in ascending order.
7. the transaction having largest number of itemsets will be put in the first cluaster CL1.the transaction having the next highest number of items will be put the next cluster CL2.
8. This process is repeated until each cluster has at most one itemset .
9. Next all the transactions in the database are scanned and put the transaction into the cluster that has the highest similarity measures with the existing itemset. The similarity is measured based on that are in common with the existing itemset .Then the number of transactions with in each cluster is counted [2].
10. After forming the cluster apply an improved Apriori algorithm based on the improved pruning optimization and transaction reduction [11]on each cluster.
11. Create a new candidate set (CK) which displays the support count of each items.
12. If its support is not less than the minimum support, the itemsets is K-frequent itemsets or else it is not. Create frequent itemset (LK) of those itemset only who don't have less support than threshold.
13. New CK+1 generated by paining K+1 and again find out the support for each pair .create new frequent itemsets (LK+1) of those items only who don't have less support than threshold.
14. Now CK' is generated by counting the support of the itemsets .if its support is not less than the minimum support ,the item sets is a K-frequent itemsets or else it is not .delete the itemset who have less support than threshold and generate new LK+1.
15. Repeat step 12 and 13 until pair of itemset don't pass the threshold and display the last itemset which pass the threshold.
16. Apply above steps in all cluster and finding itemset.
17. For finding the large itemsets it is enough to go through the transaction within the clusters. There is no need to go through the entire database again. Hence it reduces the redundant database scan and improves the efficiency.

CONCLUSION

In this paper the proposed system uses Hadoop's MapReduce framework, HDFS with Improved pruning and transaction reduce Apriori algorithm .Automatic parallelization and distribution of work among nodes is key features of MapReduce framework .By adding n number of nodes the processing power factor can be increased n times which reduces the computation cost .

REFERENCES

[1] R.Agrawal, and A. Swami-, "Mining association rules between sets of items in large databases". *In: Proc. of the*

l993ACM on Management of Data, Washington, D.C, 207-216, May 1993.

[2]D.K.Hanirex and Dr.M.A.DoraiRangaswamy:" Efficient algorithm for mining frequent item sets using clustering techniques*." In International Journal on Computer Science and Engineering Vol. 3 No. 3, 1028-1032 Mar 2011.*

[3] F. WANG and Y.huaLI:"Improvedapriori based on matrix",*IEEE , 152-155, 2008.*

[4] H.Jiawei and K.Miceline, "Data mining concepts and technologies" *Beijing: Machinery Industry Press. 2001*

[5]M.H. Dunham. Data Mining, Introductory and Advanced Topics: Upper Saddle River, New Jersey: Pearson Education Inc.,2003.

[6] C.Wenwei, "Data warehouse and data mining tutorial*" Beijing: Tsinghua University Press. 2006*

[7]Q.Tong and B.Y.Baoping,"A quantitative association rules mining algorithm"*International Conference on Computer engineering, 33(10):34-35 2007*

[8]H.Shuiyuan and H.Longjun, " A association rules mining algorithm based on matrix and trees*". Computer science. 2006, 33(7):196-198*

[9] Wael A. AlZoubi, Azuraliza Abu Bakar, Khairuddin Omar," Scalable and Efficient Method for Mining Association Rules",*In IEEE International Conference on Electrical Engineering and Informatics 2009.*

[10] W.A.AlZoubi and O.Khairuddin "An Efficient Mining of Transactional Data Using Graph-based Technique",*3rd Conference on Data Mining and Optimization (DMO) 2011, Selangor, Malaysia*

[11] Z. Chen and S. Cai " An Improved Apriori Algorithm Based on Pruning Optimization and Transaction Reduction", *in IEEE International Conference ,978-1-4577-0536-6/11, Dec 2011 .*

[12]JDean and S.Ghemawa"MapReduce:Simplified data processing on clustes".pp 137-150 OSDL 2004.