ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 5, Issue 11, November 2017

Revamped Data Partitioning Strategies for Market Basket Analysis using Hadoop Clusters

Omkar Hirve[1], Aditya Pardeshi[2], Omkar Kurhe[3]

U.G. Student, Department of Computer Engineering, Indira College of Engineering, Pune, Maharashtra, India[1]

U.G. Student, Department of Computer Engineering, Indira College of Engineering, Pune, Maharashtra, India[2]

U.G. Student, Department of Computer Engineering, Indira College of Engineering, Pune, Maharashtra, India [3]

ABSTRACT: Current algorithms for frequent item set mining aim at only balancing load equally among the computing nodes. We start the study by discovering that huge performance boost can be obtained by using innovative data partitioning strategies. On a large data set, overhead communication, redundant and irrelevant transaction computing degrade the system. We address this problem by using 'Revamped data partitioning strategies' by using map-reduce programming model. We also find out effective corelation between transactions. We implement the model using multi node hadoop clusters. Highly similar transactions are put into a data partition to improve locality without creating an excessive number of redundant transactions. Experimental results reveal that the system is conducive to reducing network and computing loads by the virtue of eliminating redundant transactions and implementing important values for transactions on Hadoop nodes. It significantly improves the performance of the existing parallel frequent-pattern scheme.

KEYWORDS: Hadoop, Map Reduce, Frequent item set mining.

I. INTRODUCTION

This paper proceeds on improving data partitioning strategies which concern decision making for partitioning. This includes removal of noisy and insignificant data. For every type of use different types of data become important. Hence partitioning the traditional way leads nowhere.

Big data is a term that refers to data sets or combinations of data sets whose size (volume), complexity (variability), and rate of growth (velocity) make them difficult to be captured, managed, processed or analyzed by conventional technologies and tools, such as relational databases and desktop statistics or visualization packages, within the time necessary to make them useful. While the size used to determine whether a particular data set is considered big data is not firmly defined and continues to change over time, most analysts and practitioners currently refer to data sets from tera bytes to petabytes as big data.

The analysis of Big Data involves multiple distinct phases as shown in the figure below, each of which introduces challenges. Many systems focus just on the analysis/modeling phase: while that phase is crucial, it is of little use without the other phases of the data analysis pipeline. Even in the analysis phase, which has received much attention, there are poorly understood complexities in the context of multi clusters where several users programs run concurrently.

II. BACKGROUND AND RELATED WORK

FREQUENT ITEM SET MINING: Frequent itemset mining [3] plays a prominent role as a data mining task. It is a method to find itemsets from the transaction database which occur frequently. The itemset is said to be frequent if it occurs in minimum number of transactions. This minimum number of transaction is indicated by support count. If the item set satisfies the support count it is said to be frequent. Minimum threshold is set for support count which is called

ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 5, Issue 11, November 2017

as minimum support for the item set. Finding frequent item sets help retail industries in making strategic plans about future trends. It can also be applied in various other fields like medicine, web usage mining, bio informatics. Once frequent itemsets are identified, association rules can be generated. A rule is said to be strong if it satisfies minimum confidence which says "If X => Y is a rule then confidence is number of transactions containing X that also contains Y". Strong association rules are required to reveal the elements that occur together in the data sets.

HADOOPMAPREDUCE: HADOOP is an Apache Software Foundation scheme that basically provides two components:
1. A distributed file system called as HDFS
2. A framework and API for map reduce jobs.
HDFS is organized similar to customary UNIX file system with the exception of that data stockpiling is dispersed over few machines. It is not proposed as a substitution to a regular file system, but instead as a file system like layer for substantial distributed frameworks to utilize. Hadoop Map-Reduce is a programming framework which can be utilized for creating applications with requirements to process enormous amount of data. It likewise underpins parallel execution on vast groups. Clients indicate a map function that sorts out a key/value pair to produce an arrangement of intermediate key value sets, and a reduce function that deals with consolidation of all intermediate key value pairs with same key.
The outline is this:
1. Map tasks perform transformation.
2. Reduce performs aggregation.
A Map Reduce work for the most part, parts the data setinto independent units which are handled by the map tasks in a totally parallel way. The structure sorts the yields of the maps, which are then contributed to the reduce tasks. Ordinarily both the input and the output of the task are put in a file-system. The structure deals with scheduling tasks, observing them and re-executes the failed tasks. Ordinarily the compute nodes and the nodes required for storage are same which is MapReduce structure and the Hadoop Distributed File System are running on the same cluster of nodes. This setup permits the framework to adequately plan tasks on the cluster where data is now present; bringing about high total transmission capacity over the cluster.

III. PROBLEMS IN EXISTING SYSTEM

Big data is a term that refers to data sets or combinations of data sets whose size (volume), complexity (variability), and rate of growth (velocity) make them difficult to be captured, managed, processed or analysed by conventional technologies and tools, such as relational databases and desktop statistics or visualization packages, within the time necessary to make them useful. While the size used to determine whether a particular data set is considered big data is not firmly defined and continues to change over time, most analysts and practitioners currently refer to data sets from 30-50 terabytes(10 12 or 1000 gigabytes per terabyte) to multiple petabytes (1015 or 1000 terabytes per petabyte) as big data. To overcome the problems with processing of Big Data, an Open Source Framework, Hadoop is been used. Hadoop has emerged as a popular way to harness the power of large clusters of computers. Map Reduce allows programmers to think in a data-centric fashion: they focus on applying transformations to sets of data records, and allow the details of distributed execution, network communication and fault tolerance to be handled by the Map Reduce framework. Map Reduce is typically applied to large batch-oriented computations that are concerned primarily with time to job completion.

But, Traditional Map Reduce implementations do not consider all the factors while processing. The main motivation is to solve these problems, by allowing pipelining between operators, while preserving fault-tolerance guarantees.

Also in current system, partitioning strategies are too static. Change of scenario demands change in work pattern. Every scenario demands different partitioning; so does market basket analysis. Bigger problem lies in static modes of partitioning large amounts of data. Also priority based partitioning is nowhere to be seen. Another problem with the system are that map reduce programming model in Hadoop which is meant to work with large amounts of data does not differentiate between size of data. If data is less Hadoop proves to be less effective.

ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 5, Issue 11, November 2017

IV. EXISTING SYSTEM ARCHITECTURE

1. Map Reduce:

MapReduce is mainly used for parallel processing of large sets of data stored in Hadoop cluster.A key-value (KV) pair is a mapping element between two linked data items - key and its value.
The key (K) acts as an identifier to the value. An example of a key-value (KV) pair is a pair where the key is the node Id and the value is its properties including neighbor nodes, predecessor node, etc. MR API provides the following features like batch processing, parallel processing of huge amounts of data and high availability.
For processing large sets of data MR comes into the picture. The programmers will write MR applications that could be suitable for their business scenarios. Programmers have to understand the MR working flow and according to the flow, applications will be developed and deployed across Hadoop clusters. Hadoop built on Java APIs and it provides some MR APIs that is going to deal with parallel computing across nodes.
The MR work flow undergoes different phases and the end result will be stored in hdfs with replications. Job tracker is going to take care of all MR jobs that are running on various nodes present in the Hadoop cluster. Job tracker plays vital role in scheduling jobs and it will keep track of the entire map and reduce jobs. Actual map and reduce tasks are performed by Task tracker.
Map reduce architecture consists of mainly two processing stages. First one is the map stage and the second one is reduce stage. The actual MR process happens in task tracker. In between map and reduce stages, Intermediate process will take place. Intermediate process will do operations like shuffle and sorting of the mapper output data. The Intermediate data is going to get stored in local file system.
1.1 Mapper Phase:
In Mapper Phase the input data is going to split into 2 components, Key and Value. The key is writable and comparable in the processing stage. Value is writable only during the processing stage. Suppose, client submits input data to Hadoop system, the Job tracker assigns tasks to task tracker. The input data that is going to get split into several input splits.
Input splits are the logical splits in nature. Record reader converts these input splits in Key-Value (KV) pair. This is the actual input data format for the mapped input for further processing of data inside Task tracker. The input format type varies from one type of application to another. So the programmer has to observe input data and to code according.
A partition module in Hadoop plays a very important role to partition the data received from either different mappers or combiners. Petitioner reduces the pressure that builds on reducer and gives more performance. There is a customized partition which can be performed on any relevant data on different basis or conditions.
Also, it has static and dynamic partitions which play a very important role in hadoop as well as hive. The partitioner would split the data into numbers of folders using reducers at the end of map reduce phase. According to the business requirement developer will design this partition code. This partitioner runs in between Mapper and Reducer. It is very efficient for query purpose.
1.2 Intermediate Process:
The mapper output data undergoes shuffle and sorting in intermediate process. The intermediate data is going to get stored in local file system without having replications in Hadoop nodes. This intermediate data is the data that is generated after some computations based on certain logics. Hadoop uses a Round-Robin algorithm to write the intermediate data to local disk. There are many other sorting factors to reach the conditions to write the data to local disks.
1.3 Reducer Phase:
Shuffled and sorted data is going to pass as input to the reducer. In this phase, all incoming data is going to combine and same actual key value pairs is going to write into hdfs system. Record writer writes data from reducer to hdfs. The reducer is not so mandatory for searching and mapping purpose.
Reducer logic is mainly used to start the operations on mapper data which is sorted and finally it gives the reducer outputs like part-r-0001etc,. Options are provided to set the number of reducers for each job that the user wanted to run. In the configuration file mapred-site.xml, we have to set some properties which will enable to set the number of reducers for the particular task.

ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 5, Issue 11, November 2017

Speculative Execution plays an important role during job processing. If two or more mappers are working on the same data and if one mapper is running slow then Job tracker assigns tasks to the next mapper to run the program fast. The execution will be on FIFO (First In First Out).

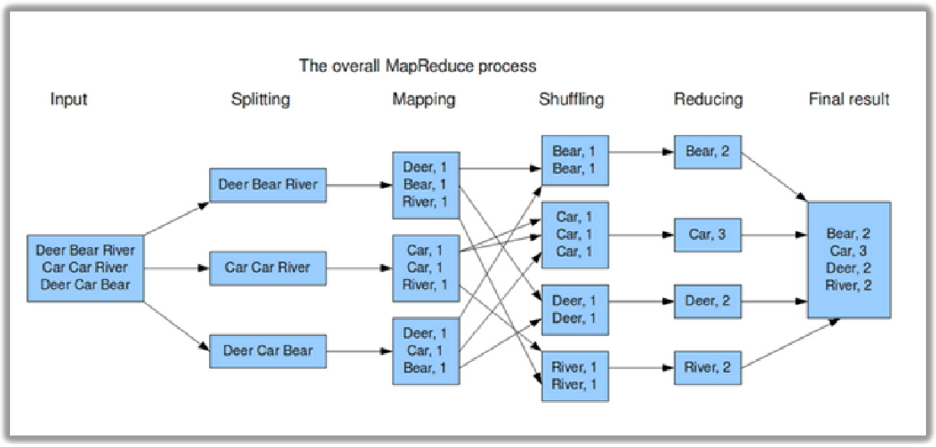

Figure 1: Typical Map Reduce process

2. HDFS : HDFS has a master/slave architecture. An HDFS cluster consists of a single NameNode, a master server that manages the file system namespace and regulates access to files by clients. In addition, there are a number of DataNodes, usually one per node in the cluster, which manage storage attached to the nodes that they run on. HDFS exposes a file system namespace and allows user data to be stored in files.
Internally, a file is split into one or more blocks and these blocks are stored in a set of DataNodes. The NameNode executes file system namespace operations like opening, closing, and renaming files and directories. It also determines the mapping of blocks to DataNodes. The DataNodes are responsible for serving read and write requests from the file system's clients. The DataNodes also perform block creation, deletion, and replication upon instruction from the NameNode.

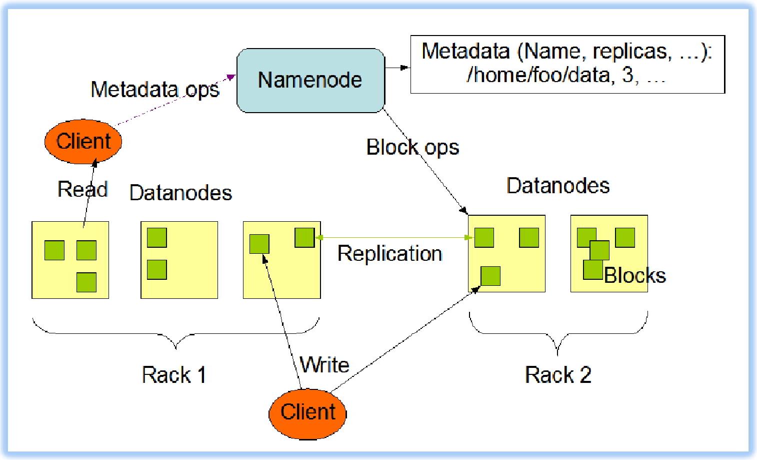

Figure 2: HDFS architecture

V. Proposed System Architecture and Algorithm

The system describes effective market basket analysis using map-reduce programming model. In each mapper sequentially reads each transaction from its local input file. Now a particular item set is transmitted to a node for further

ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 5, Issue 11, November 2017

computation. They are segregated according to frequency of data. Data considered for segregation not only includes name and count of occurrence but also includes cost of that item. Inclusion of cost of product is the major contribution towards the existing system. Cost of the product is included as because items bought frequently may be in relation to a product of relatively high cost which is never purchased due to its cost. Proposed system works on partitioning the relative data so as to reduce time required for map-reduce phase considerably. More the number of effective partitions, less the time required for map-reduce phase.

The proposed system will use Hive software for efficient data handling and its maintenance.

Map reduce programming model generates results according to name and count of occurrence. Market basket value analysis is done to find out effective results.

In market basket analysis, the user logs in and is expected to search for the some items. Based on previous user generated data set, the proposed system suggests some more items to the user. Such items suggested are also processed using our high utility algorithm to provide efficient results.

Following are the advantages of the proposed system.

1. By using this strategy the performance of existing parallel frequent-pattern increases as time decreases.
2. Data relativity for users increases.
3. Data can be effectively processed.

This paper focuses on use of dynamic partitioning. Only some queries built to work on user input or re-definable structure does not make system dynamic. Scripts implementable in various conditions should be built to improve the performance of system instead of traditional methods. Our proposed system deals with such kind of work. We intend to work on large amount of data for further betterment.

Frequent item sets are generated using effective association rules. Frequent item set patterns generated are displayed which improves quality of service provided to the users.

The proposed system can also be improved in number of ways if thought and worked in different manner.

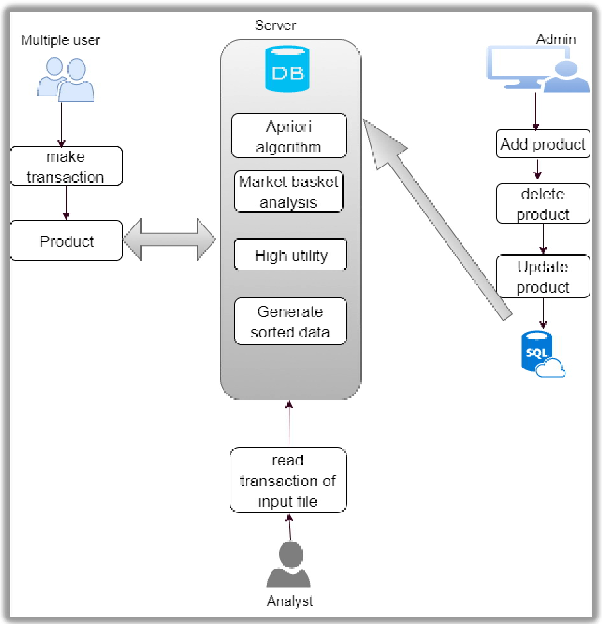

Figure 3: Proposed System architecture

ALGORITHM:

Step 1: Provide user login or user registration.
Step 2: User does the first search.
Step 3: Check for stored reference item sets.
Step 4: Generate the user items based on item sets, system calculated using frequent item set mining.
Step 5: Suggest the end result to user
Step 6: Continue the search until session ends.
Step 7: Repeat the procedure for multiple users.
Step 8: End.

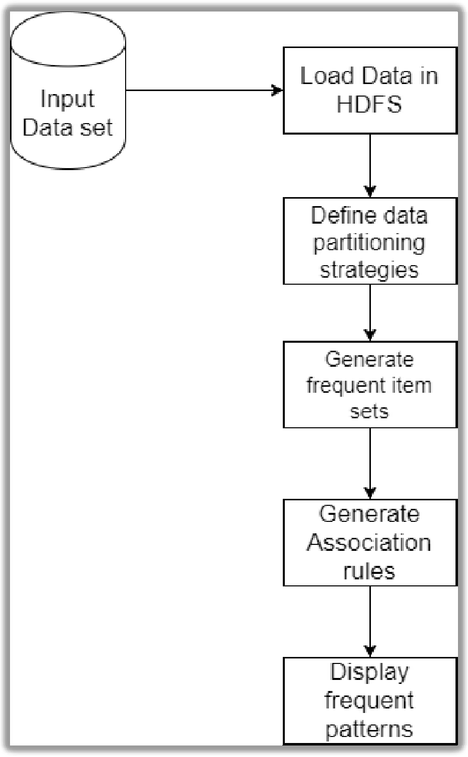
Input Data set
Load Data in HDFS
Define data partitioning strategies
Generate frequent item sets
Generate Association rules
Display frequent patterns

Figure 4: Expected flow of the system

VI. EXPIRIMENTAL RESULTS

We have used some data set to test the way system would work.It is seen that while using hadoop framework, using large sets of data is quite necessary to find out the usefulness of the system. Structuring data in Hadoop/Hive is as important as it is in a RDBMS environment. Indexing and partitioning can significantly improve performance in a Hadoop/Hive environment (like it does in a RDBMS environment) since it reduces the amount of data to be processed. Also we found out that using different number of cluster nodes changes the processing time. System also proves using various new factors for data partitioning also increases performance.

ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 5, Issue 11, November 2017

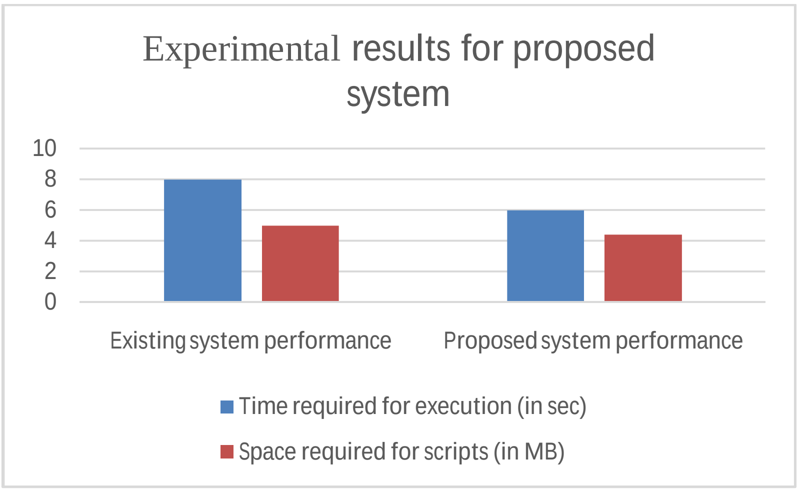

Figure 5: Experimental results for proposed system

VII. CONCLUSION AND FUTURE WORK

The simulation results showed that the proposed algorithm performs better with the total transmission energy metric than the maximum number of hops metric. The proposed algorithm provides energy efficient path for data transmission and maximizes the lifetime of entire network.As the performance of the proposed algorithm is analyzed between two metrics in future with some modifications in design considerations the performance of the proposed algorithm can be compared with other energy efficient algorithm. We have used very small network of 5 nodes, as number of nodes increases the complexity will increase. We can increase the number of nodes and analyze the performance.

REFERENCES

[1] M. J. Zaki, Parallel and distributed association mining: A survey, Concurrency, IEEE, vol. 7, no. 4, pp. 1425, 1999.
[2] I. Pramudiono and M. Kitsuregawa, Fp-tax: Tree structure basedgeneralized association rule mining, in Proceedings of the 9th ACMSIGMOD workshop on Research issues in data mining and knowledge discovery. ACM, 2004, pp. 6063.
[3] J. Dean and S. Ghemawat, Mapreduce: simplified data processing on large clusters, Communications of the ACM, vol. 51, no. 1, pp. 107113, 2008.
[4] S. Sakr, A. Liu, and A. G. Fayoumi, The family of mapreduce and large-scale data processing systems, ACM Computing Surveys (CSUR), vol. 46, no. 1, p. 11, 2013.
[5] M.-Y. Lin, P.-Y. Lee, and S.-C. Hsueh, Apriori-based frequent itemset mining algorithms on mapreduce, in Proceedings of the 6th International Conference on Ubiquitous Information Management and Communication,
ICUIMC 12. New York, NY, USA: ACM, 2012, pp. 76:176:8.
[6] X. Lin, Mr-apriori: Association rules algorithm based on mapreduce, in Software Engineering and Service Science (ICSESS), 2014 5th IEEE International Conference on. IEEE, 2014, pp. 141144.
[7] L. Zhou, Z. Zhong, J. Chang, J. Li, J. Huang, and S. Feng, Balanced parallel fp-growth with mapreduce, in Information Computing and Telecommunications (YC-ICT), 2010 IEEE Youth Conference on. IEEE,
2010, pp. 243246.
[8] S. Hong, Z. Huaxuan, C. Shiping, and H. Chunyan, The study of improved fp-growth algorithm in mapreduce, in 1st International Workshop on Cloud Computing and Information Security. Atlantis Press, 2013.
[9] M. Riondato, J. A. DeBrabant, R. Fonseca, and E. Upfal, Parma: a parallel randomized algorithm for approximate association rules mining in mapreduce, in Proceedings of the 21st ACM international conference on Information and knowledge management. ACM, 2012, pp. 8594.
[10] C. Lam, Hadoop in action. Manning Publications Co., 2010.
[11] H. Li, Y. Wang, D. Zhang, M. Zhang, and E. Y. Chang, Pfp: parallel fp-growth for query recommendation, in Proceedings of the 2008 ACM conference on Recommender systems. ACM, 2008, pp. 107114.
[12] C. Curino, E. Jones, Y. Zhang, and S. Madden, Schism: a workload driven approach to database replication and partitioning, Proceedings of the VLDB Endowment, vol. 3, no. 1-2, pp. 4857, 2010.
[13] P. Uthayopas and N. Benjamas, Impact of i/o and execution scheduling strategies on large scale parallel data mining, Journal of Next Generation Information Technology (JNIT), vol. 5, no. 1, p. 78, 2014.
[14] Yaling Xun, Jifu Zhang, and Xiao Qin, FiDoop: Parallel Mining of Frequent Itemsets Using MapReduce,IEEE Transactions on Systems, Man, and Cybernetics: Systems, Year: 2016, Volume: 46, Issue: 3
[15] Yaling Xun, Jifu Zhang, Xiao Qin, Xujun Zhao, FiDoop-DP: Data Partitioning in Frequent Itemset Mining on Hadoop Clusters, IEEE Transactions on Parallel and Distributed Systems, Year: 2017, Volume: 28, Issue: 1