ISSN: 2321-7782 (Online)
e-ISJN: A4372-3114
Impact Factor: 6.047

Volume 5, Issue 7, July 2017

International Journal of Advance Research in Computer Science and Management Studies

Research Article / Survey Paper / Case Study

Available online at: www.ijarcsms.com

Parallel Mining of Frequent Itemsets in Hadoop Cluster Having Heterogeneous Nodes

Dhanashree Shirke[1]
Department of Computer Engineering, ME,
Shree Ramchandra College of Engineering
Pune – India

Prof. Deepti Varshney[2]
Professor, Department of Computer Engineering
Shree Ramchandra College of Engineering
Pune – India

Abstract: Mining of big data is a difficult process. Many different approaches are used for obtaining the frequent itemset. Numerous existing data mining techniques are developed & presented to derive association rules and frequently occurring itemsets, but with the rapid arrival of era of big data traditional data mining algorithm have been unable to meet large datasets analysis requirements. Also these algorithms lack mechanisms like load balancing, data distribution I/O overhead, and fault tolerance. To overcome these problems various parallelized approaches using Hadoop MapReduce model are developed to perform frequent itemsets mining from big data. The current Hadoop implementation assumes that computing nodes in a cluster are homogeneous in nature. Hadoop lacks performance in heterogeneous clusters where the nodes have different computing capacity. So there is a need to develop an approach to improve performance of heterogeneous Hadoop clusters. This paper presents a literature analysis on different techniques for parallel mining of frequent itemset.

Keywords: Big Data, Data mining, FIM, Frequent Itemset Mining, Hadoop MapReduce, Parallel Mining.

I. INTRODUCTION

Today huge amount of data is being gathered together in many important areas like e-commerce, social network, finance, health care, education, banking and ticket reservation. Due to growth of IT industries, services, technologies and data, the huge amount of complex data is generated from the various sources that can be in various form. Such complex and massive data is difficult to handle and process that contain the billion records of million user & product information that includes the online selling data, audios, images, videos of social media, news feeds , product price and specification etc. The necessity of big data arrives from the worldwide famous companies like Google, Yahoo, Facebook, Microsoft, and Twitter for the reasons of analysis of huge data which can be in unstructured form. For example, Google contains the huge amount massive data. To handle and process this massive data big data analytics is needed. Big data analytics analyse the huge amount of information and reveal the association rules, hidden patterns, trends and the other meaningful information.

In 2012, Gartner has given the definition of Big Data as follows: "Big data is high volume, high velocity, and/or high variety information assets that require new forms of processing to enable enhanced decision making, insight discovery and process optimization."

Data mining is a process of discovering previously unknown and useful information from large databases. The most widely used data mining technologies include association rules discovery, clustering, classification, and sequential pattern mining. Among them, the most popular technology is association rules discovery, which is mining the possibility of simultaneous occurrence of items, and then building relationships among them in databases.

Frequent itemsets mining (FIM) is a core problem in association rule mining (ARM). Speeding up the process of FIM is critical and indispensable, because FIM consumption accounts for a significant portion of mining time due to its high computation and input/output (I/O) intensity. When datasets in modern data mining applications become excessively large, sequential FIM algorithms running on a single machine suffer from performance deterioration. To address this issue, various parallel mining algorithms were proposed and implemented. In this paper we summarize various parallel mining algorithms.

II. HADOOP MAPREDUCE MODEL

MapReduce is a model that allows developer for processing and generating massive amount of unstructured data in parallel across a distributed cluster of processors of standalone computer. As the name implies map reduce model is divided into two parts Map() procedure and Reduce() procedure. Map procedure sorting and filtering the data and Reduce procedure is used for summarize the data. Libraries of map reduce is written in many programming language. Map reduce model is based on distributed computing in parallel computing and it is proposed by Google. In MapReduce data is stored in the Hadoop Distributed File System (HDFS) and these data may be structured (file system) or unstructured (database) [1].

The Map and reduce function of MapReduce model are defined with respect of (key, value) pair. From HDFS Map takes one pair of data and returns a list of pair in different domain.

Map (K1, V1) → List (K2, V2)

To each group Reduce function is applied and from this domain summarization of values produces.

Reduce (K2, list (V2)) →List (V3)

In the Map Reduce Model there are mainly three steps:

Map step →Data is split into input splits and map() is applies on every input split and output is stores in temporary storage in this redundant data is manage.

Shuffle Step→ Data is redistribute on the base of map () output. In this data which is belonging to same key is places on one node.

Reduce Step→ In this each group of data is processed per key, in parallel [5] [14].

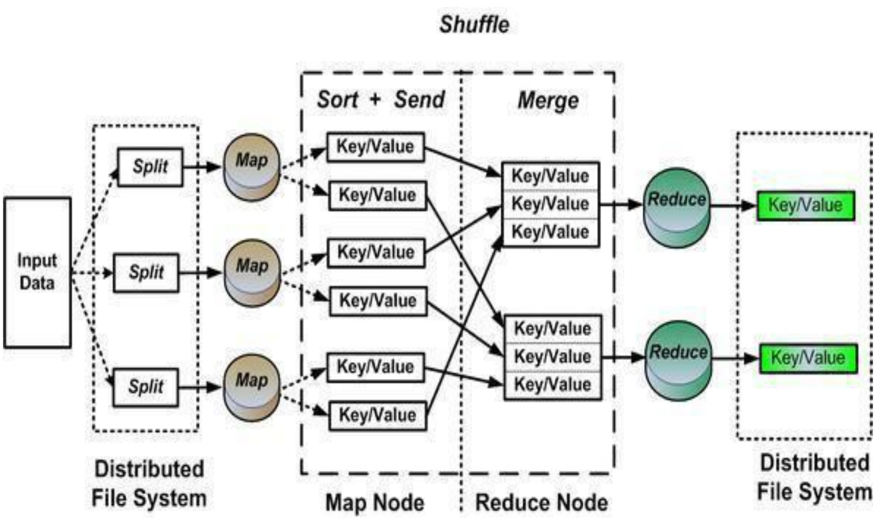

Fig 1: MapReduce Model

III. SURVEY ON DIFFERENT FIM TECHNIQUES

Basically, there are three classic frequent itemset mining algorithms that run in single node. Loop is the main logic behind success of Apriori [12] algorithms. In Apriori algorithm loop k produces frequent itemsets with length k. By using the property and o/p of k loop, loop k+1 calculate candidate itemsets. Property is: any subset in one frequent itemset must also be frequent. FP-Growth [3] algorithm creates an FP-Tree by two scan of the whole dataset and then frequent itemsets are mined from frequent pattern tree. Eclat [2] algorithm transposes the whole dataset into a new table. In this new table, every row contains list of sorted transaction ID of respective item. In last frequent itemsets are extracted by intersecting two transaction lists of that item.

A. *Modified Apriori Algorithm*

Othman et al. [16], presented two different ideas for conversion Apriori algorithm into MapReduce task. In first way, all possible itemsets are extracted in Mapping phase, and then in Reduce phase itemsets those does not satisfy minimum support threshold are taken out. In second way, direct conversion from Apriori algorithm is carried out. Every loop from Apriori algorithm is converted into MapReduce task. These presented approaches are used by [8], [11]. In this approaches large data is shuffled between Map and Reduce tasks [12]. To solve these problems, they presented MRApriori algorithm. MRApriori is nothing but MapReduce based improved Apriori algorithm which uses two-phase structure

B. *MREclat Algorithm*

Title Zang et al. [15], presented improved Eclat algorithm to increase the efficiency of FIM from large datasets. Parallel algorithm MREclat based on MapReduce framework is called as MREclat algorithm. MREclat also solves the problems of storage and capability of computation not enough when mining frequent itemsets from large complex datasets. MREclat algorithm has very high scalability and better speedup in comparison with other algorithm. Algorithm MREclat consists of three steps: in the initial step, all frequent 2-itemsets and their tid-lists from transaction database is got; the second is the balanced group step, partition frequent 1-itemsets into groups; the third is the parallel mining step, the data got in the first step redistributed to different computing nodes according to the group their prefix belong to. Each node runs an improved Eclat to mine frequent itemsets. Finally, MREclat collects all the output from each computing node and formats the final result.

C. *Dist-Eclat and BigFIM Algorithm*

Moens et al.[9], proposed two methods for frequent itemset mining for Big Data on MapReduce, First method DistEclat is distributed version of pure Eclat method which optimizes speed by distributing the search space evenly among mappers, second method BigFIM uses both Apriori based method and Eclat with projected databases that fit in memory for extracting frequent itemsets. Advantage of Dist-Eclat and BigFIM is that it provides speed and Scalability Respectively. Dist-Eclat does not provide scalability and speed of BigFIM is less.

D. *PARMA Algorithm*

Riondato et al. [6], has been presented Parallel Randomized Algorithm (PARMA algorithm) which finds set of frequent itemsets in less time using sampling method. PARMA mines frequent patterns and association rules from precise data. As a result mined frequent itemsets are approximate those are close to the original results. It finds the sampling list using k-means clustering algorithm. The sample list is nothing but clusters. The main advantage of PARMA is that it reduces data replication and algorithm execution is faster.

E. *MRPrePost Algorithm*

Liao et al. [4], presented MRPrePost algorithm based on MapReduce framework. MRPrePost is an improved version of PrePost. Performance of PrePost algorithm is improved by including a prefix pattern. On this basis, MRPrePost algorithm is well suitable for mining large data's association rules. In case of performance MRPrePost algorithm is more superior to PrePost and PFP. The stability and scalability of MRPrePost algorithm is better than PrePost and PFP. The mining result of MRPrePost is approximate which is closer to original result.

F. *ClustBigFIM Algorithm*

Big FIM [9] overcomes the problems of Dist-Eclat such as, mining of sub-trees requires entire database into main memory and entire dataset needs to be communicated to most of the mappers. BigFIM is a hybrid approach which uses Apriori algorithm for generating k-FIs, and then Eclat algorithm is applied to find frequent item sets. Candidate itemsets do not fit into memory for greater depth is the limitation of using Apriori for generating k-FIs in BigFIM algorithm and speed is slow for BigFIM. To address above limitation Gole et al. [13], proposed a method ClustBigFIM. ClustBigFIM provides hybrid approach for frequent itemset mining for large data sets using combination of parallel k-means, Apriori algorithm and Eclat algorithm. ClustBigFIM overcomes limitation of Big FIM by increasing scalability and performance. Resulting output of ClustBigFIM gives the approximate results that are closer to the original results but with faster speed. ClustBigFIM work with four steps which need to be applied on large datasets, steps are Find Clusters, Finding k-FIs, Generate Single Global TID list, Mining of Subtree.

IV. PERFORMANCE ISSUE OF HETEROGENEOUS HADOOP CLUSTER

In existing parallel FIM algorithms, each processor has to scan a database multiple times and to exchange an excessive number of candidate itemsets with other processors. These algorithms suffer from potential problems of high I/O and synchronization overhead. So they are not feasible for massive databases. Also, we observe that data locality is a determining factor for the MapReduce performance. Data placement strategy is very practical and efficient for a homogeneous Hadoop cluster setup where all nodes are identical in terms of both computing and disk capacity. In homogeneous computing environments, all the nodes have identical workload, indicating that no data needs to be moved from one node into another. In a heterogeneous cluster, however, high-performance nodes can complete processing local data faster than a low-performance node. After the fast node finished processing data residing in its local disk, the node has to handle unprocessed data in a remote slow node. The overhead of transferring unprocessed data from slow nodes to fast peers is high if the amount of moved data is huge. An approach to improve MapReduce performance in heterogeneous computing environments is to balance data load in a Hadoop cluster having heterogeneous nodes [8].

V. SYSTEM OVERVIEW

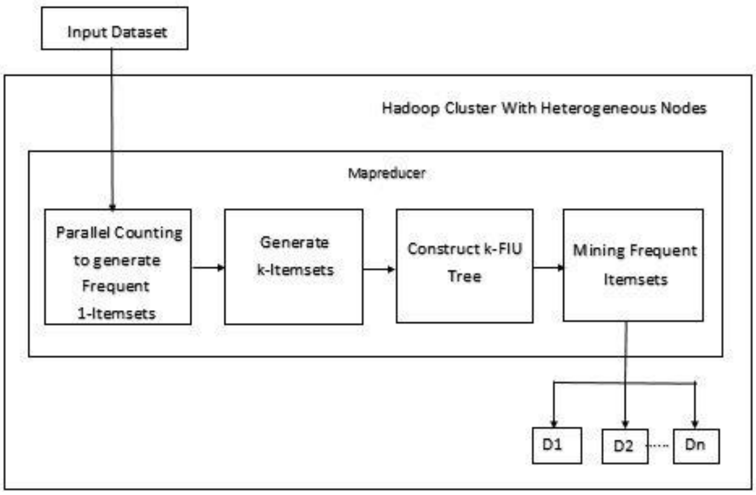

Fig2: System Architecture

In our proposed system, Hadoop cluster setup with heterogeneous nodes is used. Generally a high-speed node can finish processing data stored in a local disk of the node faster than low-speed node. After a fast node completes the processing of its local input data, the node must support load sharing by handling unprocessed data located in one or more remote slow nodes. When the amount of transferred data due to load sharing is very large, the overhead of moving unprocessed data from slow nodes to fast nodes becomes a critical issue affecting Hadoops performance. To boost the performance of Hadoop in heterogeneous clusters, we need to minimize this data movement between slow and fast nodes. Data movement can be reduced if the number of file fragments placed on the disk of each node is proportional to the node's data processing speed. This is achieved by a data placement scheme that distributes and store data across multiple heterogeneous nodes based on their computing capacities.

Then MapReduce jobs are used for performing parallel mining of frequent itemsets on Hadoop cluster. The first MapReduce job is responsible for the first round of scanning to create frequent one-itemsets. The second MapReduce job scans the database again to generate k-itemsets by removing infrequent items in each transaction. Third MapReduce job constructs k-FIU-tree (frequent Ultrametric tree) and mines all frequent k-itemsets. So, the mappers independently and concurrently decompose itemsets; the reducers perform combination operations by constructing small ultrametric trees as well as mining these trees in parallel.

Our proposed approach significantly improves performance of Hadoop heterogeneous clusters by minimizing the data movement between slow and fast nodes. Use of Frequent Itemsets Ultrametric Tree (FIUT) significantly reduces the computing time and storage space by eliminating the overhead of recursively searching and traversing conditional Frequent Pattern trees.

VI. SYSTEM ANALYSIS

To balance the data load in a heterogeneous Hadoop cluster, we develop a data placement mechanism in the Hadoop to distribute data set to multiple computing nodes in accordance with the computing capacity of each node. Then MapReduce jobs are used for performing parallel mining of frequent itemsets in Hadoop cluster.

We introduce a metric called "Threshold" to measure heterogeneity of each node in a heterogeneous cluster. Threshold is calculated as below:

T=C/W

Where,

T= Threshold factor

C= Capacity of each node in the cluster

W= Total size of all files in data grid

First large input file is divided into a number of even-sized fragments. Then, the fragments are assigned to each node in a cluster in accordance to the threshold value.

We use MapReduce job where each mapper sequentially reads each transaction from its local input split. Then, mappers compute the frequencies of items and generate local one-itemsets. Next, these one-itemsets with the same key emitted by different mappers are sorted and merged in a specific reducer, which further produces global one itemsets. Finally, infrequent items are pruned by applying the minimum support; and consequently, global frequent oneitemsets are generated.

Given frequent one-itemsets generated by the first MapReduce job, the second MapReduce job applies a second round of scanning on the database Prune infrequent items from each transaction record. The second job marks an itemset as a k-itemset if it contains k frequent items (2 <= k <=M, where M is the maximal value of k in the pruned transactions).

Each mapper in the third MapReduce job constructs local k-FIU tree independent of other mapper where leaf node contains the count of the itemset. Reducer mines all frequent itemsets only by checking the count value of each leaf in the tree.

VII. Mathematically Model of the System

Let **S = {I, F, O}** Where,

I is the set of Inputs

F is the set of functions

O is the set of Output

I= {I1}

- I1= Input Dataset

- F= {F1, F2, F3, F4,F5}
  - F1= Data Placement on Heterogeneous Nodes
  - F2=Parallel Counting to generate Frequent 1-Itemsets
  - F3=Generate k-Itemsets by pruning Original Dataset
  - F4=Construct Frequent Items ultrametric (k-FIU) Tree
  - F5=Mining Frequent Itemsets from FIU Tree
- O= {O1}
  - O1= Set of Frequent Itemsets

VIII. Results and Analysis

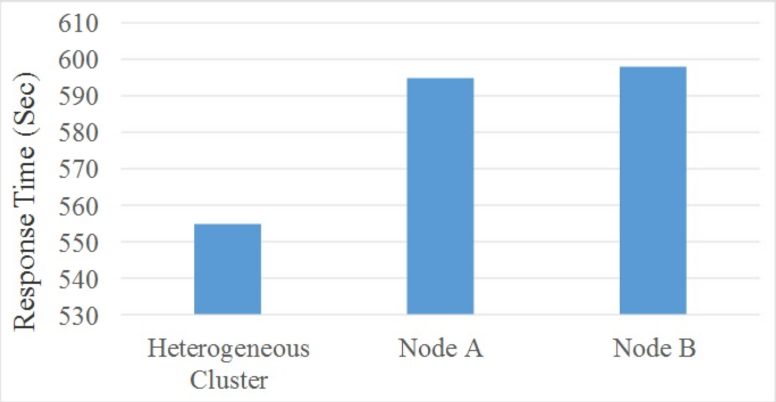

Fig 3: Impact of data placement in Hadoop cluster on performance of frequent itemsets mining

We have used input dataset in text format. Hadoop cluster having two heterogeneous nodes is used to perform parallel mining of frequent itemsets. The same amount of data is applied to each node in cluster separately and system's response time is measured in terms of seconds. Then the same amount of data is applied to Hadoop cluster having heterogeneous nodes. We observed that the response time of the system is much better. This proves that the proposed system gives better performance even when heterogeneous nodes are used to perform parallel mining of frequent itemsets.

IX. Conclusion

To solve the scalability and load balancing challenges in the existing parallel mining algorithms for frequent itemsets, MapReduce programming model is used to perform parallel frequent itemsets mining. Various FIM techniques are proposed and developed from last couple of year which overcomes the problems of memory and computational capability insufficient when mining frequent itemsets from massive datasets. Also, the current Hadoop implementation assumes that computing nodes in a cluster are homogeneous in nature. Hadoop lacks performance in Hadoop clusters having heterogeneous nodes where the nodes have different computing capacity. In this paper many FIM algorithms based on MapReduce framework are studied and a system is proposed to perform parallel FIM using Hadoop MapReduce framework on Heterogeneous Hadoop cluster.

References

1. A.Pradeepa, Dr.Antony selvadoss Thanamani, "Parallelized Comprising for Apriori algorithm using Mapreduce framework", International Journal of Advanced Research in Computer and Communication Engineering, Vol. 2, November 2013.
2. D. Laney, "3-D Data Management: Controlling Data Volume, Velocity and Variety", META Group Research Note, February 6, 2001.
3. J. Han, J. Pei, and Y. Yin, "Mining Frequent Patterns Without Candidate Generation", in Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data, ser. SIGMOD '00. New York, NY, USA: ACM, 2000, pp. 1–12.
4. Jinggui Liao, Yuelong Zhao, and Saiqin Long, "MRPrePost- A Parallel algorithm adapted for mining big data", IEEE Workshop on Electronics, Computer and Applications, 2014.
5. LeWang, Lin Feng, Jing Zhang, Pengyu Liao, "An efficient algorithm of frequent itemset mining based on MapReduce" Journal of International and computation Science, May 2014.
6. M. Riondato, J. A. DeBrabant, R. Fonseca, and E. Upfal. "PARMA: a parallel randomized algorithm for approximate association rules mining in MapReduce", In Proc. CIKM, pages 85–94. ACM, 2012.
7. 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, ser. ICUIMC '12. New York, NY, USA: ACM, 2012, pp. 76:1–76:8.
8. M.Zaharia, A.Konwinski, A.Joseph, Y.zatz, and I.Stoica. Improving mapreduce performance in heterogeneous environments. In OSDI'08: 8th USENIX Symposium on Operating Systems Design and Implementation, October 2008
9. Moens S , Aksehirli E , Goethals B , "Frequent Itemset Mining for Big Data", Big Data, 2013 IEEE International Conference on , vol., no., pp.111,118, 6-9 Oct. 2013 DOI: 10.1109/BigData.2013.6691742.
10. N. Li, L. Zeng, Q. He, and Z. Shi, "Parallel Implementation of Apriori Algorithm Based on MapReduce", in Proceedings of the 2012 13th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, ser. SNPD '12. Washington, DC, USA: IEEE Computer Society, 2012, pp. 236– 241
11. O. Yahya, O. Hegazy, and E. Ezat, "An efficient implementation of Apriori algorithm based on Hadoop MapReduce model", International Journal of Reviews in computing, vol. 12, pp. 59–67, 12 2012.
12. Rakesh Agrawal, Tomasz Imieliński, and Arun Swami, "Mining association rules between sets of items in large databases", SIGMOD Rec. 22, 2 (June 1993), 207-216. DOI=10.1145/170036.170072.
13. Sheela Gole, and Bharat Tidke, "Frequent Itemset Mining for Big Data in social media using ClustBigFIM algorithm", International Conference on Pervasive Computing (ICPC), 2015.
14. Zhang Danping, Yu Haoran and Zheng Linyu "Apriori Algorithm Research Based On Map Reduce in Cloud Computing environment" The Open Automation and Control System Journal, 2014.
15. Zhigang Zhang, GenlinJi, and Mengmeng Tang, "MREclat: an Algorithm for Parallel Mining Frequent Itemsets", 2013 International Conference on Advanced Cloud and Big Data, DOI 10.1109/CBD.2013.22.