DOI: http://dx.doi.org/10.26483/ijarcs.v8i7.4499

ISSN No. 0976-5697

Volume 8, No. 7, July – August 2017

International Journal of Advanced Research in Computer Science

SURVEY PAPER

Available Online at www.ijarcs.info

A COMPARATIVE STUDY OF DATA PARTITIONING IN FREQUENT ITEMSET MINING ON HADOOP MAPREDUCE CLUSTERS

Dr .E.Mary Shyla
Assistant Professor, Department of Computer Science
Sri Ramakrishna College of Arts and Science
Coimbatore, India

Jitha Janardhanan
MPhil Research Scholar, Department of Computer Science
Sri Ramakrishna college of Arts and Science
Coimbatore, India

Abstract: Distributed parallel algorithms for mining frequent balanced itemsets aims to load by equally dividing data among a collection of computing nodes. Over the history, frequent itemsetsbased parallel algorithm methods have been illustrated in the literature. In this comparative study aims to present a study of Frequent pattern mining techniques deviations among in Hadoop MapReduce concepttunder the data mining techniques that are in use in large database transactions broadcasted among computing nodes. Number of comparative studies has been performed to assess the performance of MapReduce cases and the outcome discloses that Spark Framework with advanced load balancing strategy having better performance than other predictive methods like Apriori, Randomized algorithms.

Keywords:Data Mining, Frequent Pattern Mining, Hadoop, Spark.

1. INTRODUCTION

Data mining is the extraction of unknown predictive information from huge databases, is a controlling new technology with great prospective to help corporations as well as research hub on the majority significant information in their data warehouses. Data mining tools forecast future developments and behaviors, allowing businesses to make practical, knowledge-driven judgments. Mining frequent itemset in distributed environment is a distributed problem and must be performed using a distributed algorithm that does not need raw data exchange between participating sites [1].

Distributed data mining is the operation of data mining in distributed data sets. According to [2], two dominant architectures exist in the distributed environments which are listed as distributed and shared memory architectures. In distributed memory each processor has a private DB or memory and has access to it. In this architecture, access to other local DB is possible only via message exchange. This architecture offers a simple programming method, where limited bandwidth may reduce the scalability. In distributed memory each processor has a private DB or memory and has access to it. In this architecture, access to other local DB is possible only via message exchange..

A basic necessity for mining association rules is mining frequent itemsets. Numerous algorithms exist for frequent itemset mining. Apriori and FP-Growth are the traditional method. Apriori 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.FP-Growth is used to overcome the problem of candidate generation. 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 is carried out by prognostic the prefix tree, working recursively on the result, and trimming the original tree. The implementation also supports shifting 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. This algorithm does not hold incremental data mining. FrequentItemsets Mining (FIM) is a center issue in association rule mining (ARM), grouping mining, and so forth.

In this paper, we explore strategies for parallel Frequent Itemset Mining techniques based on distributed Hadoop Clusters. Such representations have been usually used to partition the input domain of the system being tested, which in turn is used to choose and create clusters so as to attain certain strategies for partition coverage. Such models are widely applied for distributed database applications and are therefore a natural and a practical choice in our context.

2. RELATED WORK

S. Sakr, A. Liu, and A. G. Fayoumi [3] discussed theconstant increase of computational power has created a great flow of data which has called for a model shift in the computing architecture and large dimensional data processing systems. MapReduce is easy and controlling programming model that permits simple development of scalable parallel submissions to process vast amounts of data on large clusters of commodity machines. It separates the

submission from the details of running a distributed program such as issues on data distribution, allocation and liability tolerance. However, the innovative implementation of the MapReduce structure had some restrictions that have been attempted by many research efforts in more than a fewfollow up works after its introduction. This article presented a comprehensive survey for a group of advances and mechanisms of large scale data processing mechanisms that have been implemented based on the unique idea of the MapReduce structure and are currently gaining a lot of momentum in both research and manufacturing communities. They also cover a set of introduced schemes that have been implemented to present declarative programming interfaces on peak of the MapReduce framework.

M.-Y. Lin, P.-Y. Lee, and S.-C. Hsueh[4] proposed to improve the presentation of the Apriori-like frequent itemset mining algorithms.It is differentiated by mutually map and reduce functions, MapReduce has appeared and excels in the mining of datasets of terabyte scale or larger in moreover homogeneous or heterogeneous clusters. Reducing the allocation overhead of every map-reduce phase and maximizing the deployment of nodes in every phase are keys to flourishing MapReduce implementations. In this paper, authors presented three kind of algorithms, named SPC, FPC, and DPC to examinesuccessfulexecution of the Apriori algorithm in the MapReduce framework. In the DPC attributes in dynamically merging candidates of different lengths and outperforms together the straight-forward algorithm SPC and the predetermined passes joint counting algorithm FPC.

X. Lin [5] proposed aconventional Association Rules algorithm has calculating power shortage in dealing with huge datasets. In order to conquer these difficulties a distributed association rules algorithm based on MapReduce programming model named MR-Apriori is proposed. In this paper, authors introduced the MapReduce programming framework of Hadoop platform and Apriori algorithm of data mining proposed the detailed process of MR-Apriori algorithm. Theoretical and experimental outcomes demonstrated MR-Apriori algorithm create a sharp enhance in efficiency.

A. Arcuri and L. Briand[6] discussed a frequent itemset mining (FIM) plays an necessary function in mining associations, connections and many other significant data mining tasks. Unfortunately, as the amount of dataset gets bigger day by day, most of the FIM algorithms in literature become unsuccessful suitable to also too huge resource constraints or too much communication cost. The authors proposed a balanced parallel FP-Growth algorithm BPFP, based on the PFP algorithm, which parallelizes FP-Growth in the MapReduce approach. BPFP appends into PFP load balance feature, which advances parallelization and thereby get better performance. Through empirical study, BPFP outperformed the PFP which uses some simple grouping approach.

M. Riondato, J. A. DeBrabant, R. Fonseca, and E. Upfal [7] presented an optimal randomized parallel technique for mining Frequent Itemsets and Association Rules. The authors presentedmining algorithm, PARMA, attains near-linear accelerate while avoiding costly duplication of data. PARMA does this by generating multiple tiny random samples of the transactional dataset and running a mining algorithm on the samples separately and in parallel. The resultant collections of Frequent Itemsets or Association Rules from every sample are combined and filtered to present a single collection in output. Since PARMA mines random subsets of the dataset, the ending result is a rough calculation of the precise solution. The concluding probabilistic analysis showed that PARMA provided fixed guarantees on the excellence of the approximation. The end user identifies accuracy and confidence parameters and PARMA calculates an approximation of the group of interest that assures these parameters. The authors planned and implemented the algorithm in the MapReduce parallel computation framework.

S. Chen, Z. Chen, Z. Zhao, B. Xu, and Y. Feng[8] discussed the era of “Big Data” there is a capable need to enlargement enormous data set using huge cluster structure. Anyway, lacking the right approaches to hold the data, it is demanding to gain a good presentation from the system. In this paper authors discussed many Input/output and implementation scheduling strategies for parallel data mining submission has been investigated. The objective is to determine strategies that balance the data processing load and enhanced operate a multi-core cluster system for data mining application. Problems that impact the performance have been explored. The experimental results demonstrate that a significant performance improvement can be obtained particularly with a multi-core cluster system when a proper Input/output and job execution progression scheduling has been employed.

Y. Xun, J. Zhang, and X. Qin[9] designed a parallel frequent itemsets mining algorithm called FiDoopwith MapReduce programming model. To achieve compressed storage space and avoid building provisional pattern bases, FiDoopintegrates the frequent items ultra metric tree, quite than conventional FP trees. In FiDoop, three MapReduce jobs are executed to finish the mining job. In the vital third MapReduce job, the mappers separately decompose itemsets, the reducers execute combination operations by building asmall ultra metric trees, and the concrete mining of these trees independently. To implement FiDoop on in-house Hadoop cluster. They demonstrated that FiDoop on the cluster is responsive to data distribution and sizes, because itemsets with dissimilar lengths have different decomposition and building costs. To improve FiDoop's performance, to expand a workload balance metric to compute load balance across the cluster's computing nodes.

Yue Liu, Kang Wang, Wang Wei, Bofeng Zhang, HailinZhong [10] discussed a *k* nearest neighbor join (kNN join), considered to search*k* nearest neighbors from a dataset *S* for every object in an additional dataset *R*, is aancientprocessextensively adopted by many data mining applications. As a grouping of the *k* nearest neighbor query and the joint operation, kNN join is an expensive operation. Given the increasing volume of data, it is difficult to perform a kNN join on a centralized machine efficiently. In this paper, authors investigated how to execute*k*NN join using MapReduce which is a well-accepted structure for data-

intensive applications over clusters of computers. In brief, the mappers cluster objects into groups; the reducers execute the *k*NNconnect on all collection of objects separately. To intend an effective mapping methods that exploits pruning rules for distance filtering, and hence reduces both the shuffling and computational costs. To reduce the shuffling cost, authors proposed two approximate algorithms to minimize the number of replicas. Extensive experiments on our in-house cluster demonstrate that our proposed methods are efficient, robust and scalable.

M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica[11] discussed MapReduce and its alternatives have been highly successful in implementing large-scale data-intensive applications on commodity clusters. However, most of these systems are building around an acyclic data flow model that is not appropriate for other accepted applications. This paper focused on one such class of applications: those that reuse a working set of data across multiple parallel operations. This includes many iterative machine learning algorithms, as well as interactive data analysis tools. The authors proposed a new framework called Spark that supports these applications while retaining the scalability and fault tolerance of MapReduce. To achieve these goals, Spark introduces an abstraction called resilient distributed datasets (RDDs). An RDD is a read-only collection of objects partitioned across a set of machines that can be rebuilt if a partition is lost. Spark can outperform Hadoop by 10x in iterative machine learning jobs, and can be used to interactively query a 39 GB dataset with sub-second response time.

3. COMPARISON ANALYSIS

This paper aims to collect and consider papers that deal with Data Partitioning in Frequent Itemset Mining on Hadoop Clusters techniques. Our objective is not to undertake a logical review, but quite to provide a broad state-of-the-art view on these related fields. Many different approaches have been projected to assist FrequentItemset Mining, which has mentioned in a body of literature that is spread over a wide variety of fields and periodical locations. The majority of comparison study has been available in Big data domain, and particularly in the in Frequent Itemset Mining on Hadoop clusters maintenance literature.

Table 1: SUMMARY TABLE FOR COMPARISON OF DATA PARTITIONING IN FREQUENT ITEMSET MINING ON HADOOP MAPREDUCE CLUSTERS TECHNIQUES

Title	Algorithm	Key-Idea	Techniques	Results	Performance
Apriori-based frequent itemset mining algorithms on mapreduce [4]	Apriori based DPC	DPC: Dynamic Passes Combined-counting a balance between reducing the number of map-reduce phases.	Frequent Item Mining	Dynamically collects candidates of variable lengths for counting by mappers according to the number of candidates and the Execution time.	To identify execution time performs 89 % better.
MR-Apriori: Association Rules Algorithm Based on MapReduce [5]	Distributed association rules algorithm	Computing power shortage in dealing with massive datasets.	Data mining; MapReduce; Hadoop	MR-Apriori algorithm has better performance on mining frequent itemsets from the mass data.	Execution time in larger nodes is 45%.
Balanced parallel FP-growth with mapreduce [6]	Balanced Parallel FP-Growth	Parallelizes FP-Growth in the MapReduce approach	Distributed computing and MapReduce	Improves performance of the original PFP algorithm by balancing load of the parallel FP-Growth phase.	FP-Growth has a great influence on 50% performance of the algorithm.
Parma: A parallel randomized algorithm for approximate association rules mining in MapReduce [7]	Frequent Itemsets and Association Rules Mining (FIM)	Minimizing data replication and Communication cost	Random Sampling Approach	The presence makes the distribution more robust to outliers	30-55% runtime improvement over Parallel FP-Growth (PFP).
FiDoop: Parallel mining of frequent itemsets using MapReduce [9]	Parallel frequent itemsets mining algorithm (PFM)	Automatic parallelization, data distribution, and fault tolerance on large clusters.	Hadoop cluster, load balance, MapReduce	To improve the performance of FiDoop by balancing I/O load across data nodes of a cluster	Increases communication overhead between mappers and reducers.
Spark: Cluster	Spark	Retaining the	Logistic	To (re)construct the	Useful in

Computing with Working Sets [11]	Framework with resilient distributed datasets	scalability and fault tolerance of MapReduce	Regression	dataset from data available in reliable storage.	developing other abstractions for programming clusters.

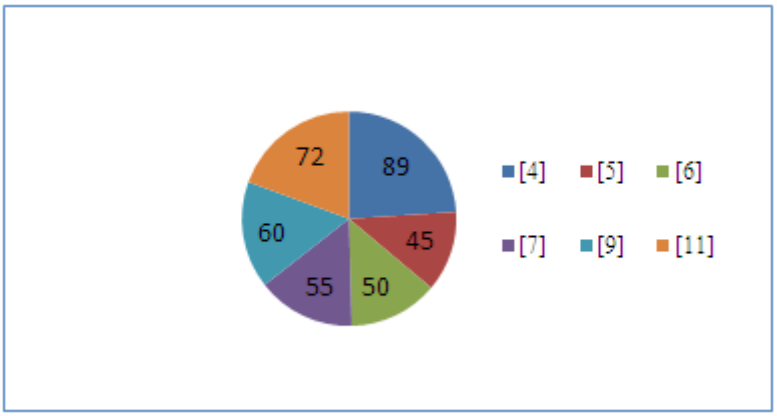

Figure.1: Comparison of Method and its efficiency

4. CONCLUSION

This paper presents acomparative study of the various Frequent Mining Algorithm in MapReduce techniques deviations for Parallel distributed data(Big Data) discussed with the different categories in which algorithms can be classified (i.e., Apriori, Balanced Parallel FP-Growth, Frequent Itemsets and Association Rules Mining (FIM), Parallel frequent itemsets mining algorithm (PFM) and Spark Framework).The discussionis concludedon MapReduce Clusters algorithms with PFMby a comparative study with Spark Framework category. The concept of Hadoop clusters which proves to be the most important criteria for Utility mining is also been discussed here.

The further work enhanced andexpanded for the automation of Enhanced FM deviations for Spark framework using *Distributed advanced load balancing* algorithm.

5. REFERENCES

[1] E Ansari, M.keshatkaran march 2008,Distributed Trie Frequent Itemset Mining, IMECS Vol II

[2] Mohammed j zaki Parallel and distributed association mining :A survey

[3] S. Sakr, A. Liu, and A. G. Fayoumi, “The family of mapreduce and large-scale data processing systems,” ACM Comput. Surveys, vol. 46, no. 1, p. 11, 2013.

[4] M.-Y. Lin, P.-Y. Lee, and S.-C. Hsueh, “Apriori-based frequent itemset mining algorithms on mapreduce,” in Proc. 6th Int.Conf. Ubiquitous Inform. Manag. Commun., 2012, pp. 76:1–76:8.

[5] X. Lin, “Mr-apriori: Association rules algorithm based on mapreduce,” in Proc. IEEE 5th Int. Conf. Softw. Eng. Serv. Sci., 2014, pp. 141–144.

[6] L. Zhou, Z. Zhong, J. Chang, J. Li, J. Huang, and S. Feng, “Balanced parallel FP-growth with mapreduce,” in Proc. IEEEYouth Conf. Inform. Comput. Telecommun., 2010, pp. 243–246.

[7] M. Riondato, J. A. DeBrabant, R. Fonseca, and E. Upfal, “Parma: A parallel randomized algorithm for approximate association rules mining in mapreduce,” in Proc. 21st ACM Int. Conf. Informa. Knowl. Manag., 2012, pp. 85–94.

[8] P. Uthayopas and N. Benjamas, “Impact of i/o and execution scheduling strategies on large scale parallel data mining,” J. Next Generation Inform. Technol., vol. 5, no. 1, p. 78, 2014.

[9] Y. Xun, J. Zhang, and X. Qin, “Fidoop: Parallel mining of frequent itemsets using mapreduce,” IEEE Trans. Syst., Man, Cybern.: Syst., vol. 46, no. 3, pp. 313–325, Mar. 2016, doi: 10.1109/TSMC.2015.2437327.

[10] W. Lu, Y. Shen, S. Chen, and B. C. Ooi, “Efficient processing of k nearest neighbour joins using mapreduce,” Proc. VLDB Endowment, vol. 5, no. 10, pp. 1016–1027, 2012.

[11] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica, “Spark: Cluster computing with working sets,” in Proc. 2nd USENIX Conf. Hot Topics Cloud Comput., 2010, p. 10.