Survey on FiDoop: Data Hierarchy Mining of Frequent Item Sets using MapReduce

Achyut Bhanudas Rao[1] Amrita Anand[2]
[1]M.Tech Student [2]Assistant Professor
[1,2]Department of Computing and Information Technology
[1,2]Reva University, Bengaluru, India

Abstract— Existing parallel mining algorithms lacks some of the features like parallelization of sequential code, balancing load over cluster of computers, and distribution of data over computers in a cluster. To overcome these problems we have introduced an algorithm called FiDoop. Which uses the MapReduce techenique for parallel frequent itemsets mining algorithm? FiDoop algorithm uses the ultrametric tree pattern for storage of data, it does not uses the FP(frequent pattern) trees like existing algorithm uses it. In FiDoop, mining task is completed with the help of MapReduce technique. This technique incorporates three MapReduce job to mine the large amount of data conventionally and economically. In first MapReduce job , all frequent itemsets are discovered. In second MapReduce job, it removes infrequent itemstes. Third MapReduce is most important, it construct small ultrametric trees which is helpful to mine the frequent data conventionally and economically. FiDoop algorithm is implemented in Hadoop cluster. For high dimensional data, FiDoop-HD is used, it is an improved version of FiDoop. FiDoop algorithm is an improved version of FIUT(frequent itemset ultrametric tree) algorithm.

Key words: Frequent item sets, frequent items ultrametric tree (FUIT), Ha doop cluster, load balance, Map Reduce

I. INTRODUCTION

The main aim of this paper is frequent itemset mining. The existing mining algorithm lacks in some areas like it is expensive to mine the required data, the time required to mine the data is more, it require more storage while processing the data. The existing system uses the FIUT(frequent itemset ultrametric tree), but it lacks some of the features like parallelizing the data. To overcome this problem FiDoop algorithm is introduced.

FiDoop algorithm overcomes these problems. In FiDoop algorithm the data is decomposed and with the help of ultrametric tree the data is stored. With ultrametric tree we can mine our data or we can get our data very easily, we do not have to scan the tree again and again to get our data. FiDoop uses some special scheme to distribute the data over nodes of the cluster. For high-dimentional data, FiDoop-HD is used. The FiDoop has some special features like parallization of sequential data which improve the performance of data mining. To distribute the data over nodes so that it does not degrade the performance by over loading the data at one node in a cluster.

The Map Reduce jobs are performed in FiDoop. In first Map Reduce jobs all frequent itemsets or one itemset are discovered. In second Map Reduce it removes infrequent itemset. The third Map Reduce most important, it construct small ultrametric tree which is helpful or it is easy to mine the frequent data conventionally and economically. It requires less time to process the data compared to other existing algorithms.

II. LITERATURE SURVEY

D. Chen et al.[1], proposed a "Tree partition based parallel frequent pattern mining on shared memory systems". In this paper, we show a tree-segment algorithm for parallel mining of frequent itemset examples. Our work is taking into account FP-Growth algorithm, which consist of construction of tree and mining. The fundamental thought is to built single FP-Tree in the memory, segment it into a few free parts and appropriate them to diverse strings. A heuristic calculation is to equalization the workload over the cluster of computers. this algorithm cannot just reduce the effect of locks amid the tree-building stage, in any case, likewise dodge the overhead that do awesome damage to the mining stage. It show the trials on various sorts of datasets and contrast the outcomes and other parallel methodologies. The outcomes propose that the methodology has extraordinary point of preference in effectiveness, particularly on certain sorts of datasets. As the quantity of processors builds, our parallel algorithm indicates great adaptability.

Y.-J. Tsay et al.[2], proposed a "FIUT: A new method for mining frequent itemsets,". This paper proposes a productive technique, the frequent itemset ultrametric trees (FIUT), for mining incessant itemsets in a database. FIUT utilizes an extraordinary continuous things ultrametric tree (FIU-tree) structure to improve its proficiency in getting continuous itemsets. Looked at to related work, FIUT has four noteworthy advantages. To start with, it minimizes I/O overhead by examining the database just twice. Second, the FIU-tree is an enhanced approach to segment a database, which results from grouping exchanges, and fundamentally diminishes the inquiry space. Third, just incessant things in every exchange are embedded into the FIU-tree for fully packed storage. At long last, all successive itemsets are produced by checking the leaves of each FIU-tree, without crossing the tree recursively, which altogether decreases processing time. FIUT was contrasted and FP-development, an understood and broadly utilized algorithm and the reenactment results demonstrated that the FIUT beats the FP-development.

E.-H. Han, G. Karypis, and V. Kumar[3],depicts "Scalable parallel data mining for association rules,". One of the vital issues in information mining is discovering association rules from databases of exchanges where every exchange comprises of an arrangement of items. The most time devouring operation in this revelation procedure is the calculation of the recurrence of the events of intriguing subset of things (called applicants) in the database of exchanges. To prune the exponentially extensive space of applicants, most existing calculations, consider just those applicants that have a client characterized least backing.

All rights reserved by *www.ijsrd.com*

Indeed, even with the pruning, the errand of discovering all affiliation rules requires a great deal of calculation power and time. Parallel PCs offer a potential answer for the calculation prerequisite of this undertaking, gave proficient and versatile parallel calculations can be composed. In this paper, it exhibit two new parallel calculations for mining affiliation rules. The Intelligent Information Distribution calculation effectively utilizes total memory of the parallel PC by utilizing wise applicant partitioning plan and uses proficient correspondence component to move information among the processors. The Half and half Distribution calculation further enhances the InteUigent Information Distribution calculation by powerfully dividing the hopeful set to keep up great burden equalization. The trial results on a Cray T3D parallel PC demonstrate that the Hybrid Distribution algorithm scales directly furthermore, misuses the total memory better and can create more association rule with a solitary output of database per pass.

K.-M. Yu et al.[4],presented "A load-balanced distributed parallel mining algorithm,". Because of the exponential development in overall data, organizations need to manage a perpetually developing measure of advanced data. A standout amongst the most essential difficulties for information mining is rapidly and effectively finding the relationship among information. The Apriori calculation has been the most well known strategy in finding continuous examples. Nonetheless, while applying this strategy, a database must be checked numerous times to figure the checks of countless itemsets. Parallel and dispersed figuring is a viable technique for quickening the mining process. In this paper, the Distributed Parallel Apriori (DPA) calculation is proposed as an answer for this issue. In this reference, metadata are put away as Transaction Identifiers (TIDs), such that just a solitary sweep to the database is required. The approach likewise takes the element of itemset tallies into thought, along these lines producing an adjusted workload among processors and diminishing processor unmoving time. Probes a PC bunch with 16 processing hubs is likewise made to demonstrate the execution of this approach and contrast it and some other parallel mining calculations. The test results demonstrate that the proposed approach beats the others, particularly while the base backings are low.

L. Zhou *et a.l[5] proposed* a "Balanced parallel FP-growth with MapReduce,". Regular itemset mining (FIM) assumes a key part in mining affiliations, connections and numerous other critical information mining errands. Lamentably, as the volume of dataset gets bigger step by step, most of the FIM calculations in writing get to be ineffectual because of either excessively tremendous asset prerequisite or as well much correspondence cost. In this reference, it propose an adjusted parallel FPGrowth calculation BPFP, in light of the PFP calculation [1], which parallelizes FPGrowth in the MapReduce approach. BPFP includes into PFP load parity highlight, which enhances parallelization and consequently enhances execution. Through exact study, BPFP beat the PFP which utilizes some straightforward gathering procedure.

K. W. Lin, P.-L. Chen, and W.-L. Chang,[6] presented "A novel frequent pattern mining algorithm for very large databases in cloud computing environments," . FPgrowth is the most renowned calculation for finding incessant examples. As the database size developments on the other hand the base bolster diminishes, in any case, both of the memory prerequisite and execution time increment enormously. Numerous scientists attempted to take care of this issue by using conveyed registering procedures to enhance the adaptability and execution proficiency. In this paper, we propose a strategy for finding continuous examples from extensive database in distributed computing situations. To construct the entire FP Tree, we utilize the circle as the auxiliary memory. Since the plate access is much slower than fundamental memory, a proficient information structure for putting away and recovering FPTree from circle is moreover proposed. Through observational assessments on different reproduction conditions, the proposed strategy conveys magnificent execution as far as adaptability and execution time.

S. Hong, Z. Huaxuan, C. Shiping, and H. Chunyan,[7] proposed "The study of improved FP-growth algorithm in MapReduce," . As FP-Growth calculation produces a lot of contingent example bases and restrictive example trees, prompting low productivity, propose an enhanced FP-Growth (IFP) calculation which firstly consolidates the same examples taking into account the circumstance whether the backing of the exchange is more prominent than the base support (min_sup) to mine the continuous examples. Accordingly the IFP eliminates the space and enhances the effectiveness. It likewise makes it simple to be paralleled. Advance more, consolidate the IFP calculation with the MapReduce registering model, named MR-IFP(MapReduce-Improved FP), to enhance the ability to manage the mass information.

III. PROPOSED SYSTEM

The three MapReduce jobs of our proposed FiDoop are described in detail. The first MapReduce job discovers all frequent items or frequent one-itemsets. In this phase, the input of Map tasks is a database, and the output of Reduce tasks is all frequent one-itemsets. The second MapReduce job scans the database to generate *k*-itemsets by removing infrequent items in each transaction. The last MapReduce job—the most complicated one of the three—constructs *k*-FIU-tree and mines all frequent *k*-itemsets.

Comparing References based on different parameters like "automatic parallelization, load balancing, data mining efficiency, data distribution, scanning the database, storage"

References	Advantages	Disadvantages
Reference [1], "Tree partition based parallel frequent pattern mining on shared memory systems"	Parallel mining on dataset, efficient	It lacks in load balancing, need large database for storage, cannot be used on large database
Reference[2],"FIUT: A new method for mining frequent itemsets"	It require less storage space for database, no need to scan the database again and again for mining, parallel mining on	Lacks in automatic parallelization And load balancing

All rights reserved by *www.ijsrd.com*

	database	
Reference[3], "Scalable parallel data mining for association rules"	Distribution of data over data nodes, parallel mining	Not efficient, require more time for mining
Reference[4],"A load-balanced distributed parallel mining algorithm"	Parallel mining on database, work load balance	Lacks in automatic parallelization And is expensive data mining
Reference[5], "Balanced parallel FP-growth with MapReduce"	Parallel FP growth, use MapReduce programming model for large database	Lacks in load balancing and is expensive for mining

Table 1:

IV. CONCLUSION

To illuminate the adaptability and burden adjusting challenges in the current parallel digging calculations for continuous itemsets, we connected the MapReduce programming model to create a parallel incessant itemsets mining algorithm called FiDoop. FiDoop consolidates the continuous things ultrametric tree or FIU-tree as opposed to routine FP trees, in this way accomplishing compacted capacity and staying away from the need to construct restrictive example bases. FiDoop flawlessly incorporates three MapReduce occupations to fulfill parallel mining of incessant itemsets. The third MapReduce work assumes a vital part in parallel mining; its mappers freely break down itemsets though its reducers build little ultrametric trees to be independently mined. We enhance the execution of FiDoop by adjusting I/O load crosswise over information hubs of a group

REFERENCES

[1] D. Chen et al., "Tree partition based parallel frequent pattern mining on shared memory systems," in Proc. 20th IEEE Int. Parallel Distrib. Process. Symp. (IPDPS), Rhodes Island, Greece, 2006, pp. 1–8.
[2] Y.-J. Tsay, T.-J. Hsu, and J.-R. Yu, "FIUT: A new method for mining frequent itemsets," Inf. Sci., vol. 179, no. 11, pp. 1724–1737, 2009.
[3] E.-H. Han, G. Karypis, and V. Kumar, "Scalable parallel data mining for association rules," IEEE Trans. Knowl. Data Eng., vol. 12, no. 3, pp. 337–352, May/Jun. 2000.
[4] K.-M. Yu, J. Zhou, T.-P. Hong, and J.-L. Zhou, "A load-balanced Distributed parallel mining algorithm," Expert Syst. Appl., vol. 37, no. 3, pp. 2459–2464, 2010.
[5] L. Zhou et al., "Balanced parallel FP-growth with MapReduce," in Proc. IEEE Youth Conf. Inf. Comput. Telecommun. (YC-ICT), Beijing, China, 2010, pp. 243–246.
[6] K. W. Lin, P.-L. Chen, and W.-L. Chang, "A novel frequent pattern mining algorithm for very large databases in cloud computing
[7] S. Hong, Z. Huaxuan, C. Shiping, and H. Chunyan, "The study of improved FP-growth algorithm in MapReduce," in Proc. 1st Int. Workshop Cloud Comput. Inf. Security, Shanghai, China, 2013, pp. 250–253.
[8] M.-Y. Lin, P.-Y. Lee, and S.-C. Hsueh, "Apriori-based frequent itemset mining algorithms on MapReduce," in Proc. 6th Int. Conf. Ubiquit. Inf. Manage. Commun. (ICUIMC), Danang, Vietnam, 2012, pp. 76:1–76:8. [Online]. Available: http://doi.acm.org/10.1145/2184751.2184842
[9] L. Liu, E. Li, Y. Zhang, and Z. Tang, "Optimization of frequent itemset mining on multiple-core processor," in Proc. 33rd Int. Conf. Very Large Data Bases, Vienna, Austria, 2007, pp. 1275–1285.
[10] A. Javed and A. Khokhar, "Frequent pattern mining on message passing multiprocessor systems," Distrib. Parallel Databases, vol. 16, no. 3, pp. 321–334, 2004.
[11] J. Dean and S. Ghemawat, "MapReduce: A flexible data processing tool," Commun. ACM, vol. 53, no. 1, pp. 72–77, Jan. 2010.
[12] L. Cristofor. (2001). Artool Project[J]. [Online]. Available: http://www.cs.umb.edu/laur/ARtool/, accessed Oct. 19, 2012.
[13] E.-H. Han, G. Karypis, and V. Kumar, "Scalable parallel data mining for association rules," IEEE Trans. Knowl. Data Eng., vol. 12, no. 3, pp. 337–352, May/Jun. 2000.
[14] I. Pramudiono and M. Kitsuregawa, "Parallel FP-growth on PC cluster," in Advances in Knowledge Discovery and Data Mining. Berlin, Germany: Springer, 2003, pp. 467–473.

All rights reserved by *www.ijsrd.com*