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. 7, Issue 2, February 2019

Distributed Efficient Economic Analysis of Bigdata using Frequent Pattern Hadoop

Gokul Bharath C, Gowtham A, Akilan S.S, Aravind V, Dr.S.Madhavi MCA.,M.E.,Ph.D.,

Dept. of Computer Science and Engineering, K.S.Rangasamy College of Technology, Tiruchengode, Tamilnadu, India

ABSTRACT: To solve the scalability and load balancing challenges in the parallel mining algorithms for frequent itemsets. This project applied the MapReduce programming model. This project develop a parallel frequent itemsets mining algorithm called FiDoop. FiDoop incorporates the frequent items ultrametric tree or SIU-tree rather than conventional FP trees, thereby achieving compressed storage and avoiding the necessity to build conditional pattern bases. FiDoop seamlessly integrates three MapReduce jobs to accomplish parallel mining of frequent itemsets. The third MapReduce job plays an important role in parallel mining; this project deals with the analysis of agricultural data which helps in better understanding of agriculture in India. The outcome of the crops depends on the several factors like rainfall, season, temperatures, and also the price announced by the government. In the project, the data will categorize the crops based on the season, Minimum price announced by the government and temperature. The data also allows the user to visualize a different kind of representation and it provide different crops planted in Area (Hectares) and Production (Tonnes). The project also visualizes the different categories of data for the better of understanding of Agriculture in India.

1. INTRODUCTION

In this project to solve scalability and load balancing challenges in the parallel mining algorithms for frequent item sets and second to focus on agriculture problems by using big data. Big Data is an emerging technology to process large amount of data. Big Data, massive volumes of data with a variety that can be captured, analysed and used for making decision. Therefore, studying enormous data processing and storage become more and more popular now a days. By using the data and the
Dataset are precipitation, minimum temperature, average temperature, maximum temperature and reference crop evapotranspiration, area, production and yield for the season from January to December for the years 2000 to 2017. Engine education systems can be used to improve prediction of crop yield under different climatic scenarios. This prediction will help the farmer to choose whether the particular crop is suitable for that area and in the particular season. This prediction is carried out by the Bayesian network algorithm where high accuracy can be achieved. The Bayesian Network classification analysis technique is used for exploring the dataset. It creates Net groups from a set of objects so that the members of a group are more similar. Bayesian Network classification represents the probability relationship between crop and its productivity. It will constructs an network with collection of different Rice crop Coconunt, Arecanut, Black pepper and Dry ginger crop data on different climatic conditions and predict its probability value. The Bayesian Network classification is a supervised learning model which means temperature and rainfall analyzes the crop data used for classification and accuracy values of crops details.n generated in the past 3 years.

II. PROPOSED SYSTEM

To overcome the above problem in the existing system this project a new proposed system that reduce middle men problem and farmers can learn new methodologies from the experts through this project. They can also plan for planting the best crop with the help of weather alerts that gives information about the climatic conditions and which crop is suitable for that particular climate. This new findings suggest that it is flexible for data mining users to judiciously switch a solution between FiDoop and FiDoop-HD depending on the size of input itemsets and the number of dimensions. In this paper, introduced a metric to measure the load balance of FiDoop. This project will apply this metric to investigate advanced load balance strategies in the context of FiDoop. For example, this project plan to

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. 7, Issue 2, February 2019

implement a data-aware load balancing scheme to substantially improve the load-balancing performance of FiDoop. Our data placement scheme is favorable to matching the amount of data stored in each heterogeneous node to attain improved data-processing performance. This project will integrate FiDoop with the data-placement mechanism on heterogeneous clusters. One of the goals is to investigate the impact of heterogeneous data placement strategy on Hadoop-based parallel mining of frequent itemsets. Instead of using the data mining techniques this will use the Map reduction Techniques of Big Data Analysis. The Big data Analytics provides following advantages compared to data mining techniques

III. MODULE DESCRIPTION

3.1 PRE-PROCESSING

As a volume of database rises day by day traditional frequent itemset mining algorithms becomes ineffective. As a solution to this problem parallel mining of frequent itemsets using FIUT algorithm is implemented on MapReduce framework. this project using FIUT algorithm rather than traditional FP-Tree algorithm because to avoid building conditional patterns and to achieve compressed storage. This project build this using Hadoop framework. The working flow of FIUT algorithm on MapReduce framework consists of three MapReduce job. Synthetic datasets are used for the experimental analysis.

3.2 FREQUENT ONE ITEMSETS GENERATION

The first MapReduce job is responsible for mining all frequent one-item sets. A transaction database is partitioned into many input split files stored by the HDFS diagonally multiple data nodes of a Hadoop cluster. Number of mapper will be executed based on number of input split. Each mapper sequential delivers each transaction from its local input split, where each transaction is stored in the set-up of key value pair <Long Writable offset, Text record> by the record reader. 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 minsupport; and consequently, global frequent one-itemsets are generated and written in the form of pair <Text item, LongWritable count> as the output from the first MapReduce job. Importantly, frequent one-itemsets along with their counts are stored in a local file system, which becomes the input of the second MapReduce job in FiDoop.

3.3 ALL K ITEMSETS GENERATION

Given frequent one-itemsets generated by the first MapReduce job, the second MapReduce job applies a second round of scanning on the database to prune infrequent items from each transaction record. The additional job scripts an itemset as a k-itemset if it contains k frequent items ($2 \leq k \leq M$, where M is the maximal value of k in the pruned transactions). Each mapper of the subsequent job takes transactions as input. Then, the mapper emits a set of pair <ArrayWritable itemsets, Longwritable ONE>, in which itemsets is composed of the number of the items produced by prun ing and the set of items. These pairs obtained by the second MapReduce job's mappers are combined and shuffled for the second job's reducers. After execution the combination process, each reducer emits key/value pairs, where the key is the number of each itemset and the value is each itemset and its count. More formally, the output of the second MapReduce job is pair <IntWritable> item number, MapWritable <ArrayWritable> k-item, LongWritable SUM>> outlines the pseudocode of the second job's Map and Reduce functions. It is imperative to safeguard that frequent items in each transaction should retain their lexicographical order in order to facilitate the next phase.

3.4 FREQUENT K ITEMSETS GENERATION

The third MapReduce job a computationally expensive phase is dedicated to: 1) decomposing itemsets; 2) constructing k-SIU trees; and 3) mining frequent itemsets. The main goal of each mapper is twofold: 1) to decompose each k-itemset obtained by the second MapReduce job into a list of small-sized sets, where the number of each set is anywhere between 2 to $k - 1$ and 2) to construct an SIU-tree by merging local decomposition results with the same length. The third MapReduce job is highly scalable, because the decomposition procedure of each mapper is independent of the other mappers. In other words, the multiple mappers can perform the decomposition process in parallel. Such an SIU-tree construction recovers data storage efficiency and I/O presentation; the improvement is made possible thanks to

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. 7, Issue 2, February 2019

merging the same itemsets in advance using small SIU trees. The Map function of the third job generates a set of key/value pairs, in which the key is the number of items in an itemset and the value is an SIU-tree that is comprised of non leaf and leaf nodes. Non leaf nodes include item-name and node-link; leaf nodes include item-name and its support. In doing so, itemsets with the same number of items are delivered to a single reducer. By parsing the key-value pair (k2, v2), the reducer is responsible for constructing k2-SIU-tree and mining all frequent itemsets only by checking the count value of each leaf in the k2-SIU-tree without repeatedly traversing the tree.

IV. ALGORITHM

4.1FREQUENT ITEMSET ULTRI-METRIC TREE

FIUT is a new practice for mining frequent itemsets from the database. FIUT has four major advantages over traditional FP-tree like: it involves only two round of scanning which minimizes I/O overhead. Then the FIUT is a highly improved way to partition a database, which considerably reduces the search space. Next is here only frequent items in each transaction are inserted as nodes into the FIUT for compressed storage. Atlast all frequent itemsets are generated without traversing the tree recursively by checking the leaves of each FIUT which reduces computing time significantly. FIUT consists of two phases to generate the frequent itemsets from the transactions by two rounds of scanning the database.

4.2 FREQUENT ITEMSET ULTRA-METRIC TREE

FIUT is a new way for mining frequent itemsets from the database. FIUT has four major advantages over traditional FP-tree like: it involves only two round of scanning which minimizes I/O overhead. Then the FIUT is a highly improved way to partition a database, which considerably reduces the search space. Next is here only frequent items in each transaction are inserted as nodes into the FIUT for compressed storage. Atlast all frequent itemsets are generated without traversing the tree recursively by checking the leaves of each FIUT which reduces computing time significantly. FIUT consists of two phases to generate the frequent itemsets from the transactions by two rounds of scanning the database.

V. CONCLUSION

To solve the scalability and efficiency in the existing parallel mining algorithms for frequent itemsets for frequent itemsets, this project applied the parallel mining of frequent itemsets using Frequent Itemset Ultrametric Tree on MapReduce framework. This project in corporate the Frequent Itemset Ultrametric Tree rather than conventional FP trees, thereby achieving compressed storage and avoiding the necessity to build conditional pattern bases. The proposed algorithm integrates three MapReduce jobs to accomplish parallel mining of frequent itemsets. At the end of the third MapReduce job all frequent K itemsets are generated. To evaluate the performance of the proposed FIUT algorithm on MapReduce framework this project use synthetic datasets in our experiments. this project have kept some good amount of work in order to find the inner relationships between the agriculture parameters and this project are hoping that this project have done that. The main objective of this paper is to help the farmers or agriculture workers such that they can do agriculture more smartly in a much better calculated way

ACKNOWLEDGEMENT

We Acknowledge DST-File No.368. DST – FIST (SR/FIST/College-235/2014 dated 21-11-2014) for financial support and DBT –STAR-College-Scheme-ref.no:BT/HRD/11/09/2018 for providing infrastructure support.

REFERENCES

1. Chang E.Y., Li H., Wang Y., Zhang D. and Zhang M. (2008), 'PFP: Parallel FP-growth for query recommendation', in Proc. ACM Journal. Recommend.Syst., Lausanne, Switzerland, pp. 107–114.
2. Chang W.L., Chen P.L. and Lin K.W. (2011), "A novel frequent pattern mining algorithm for very large databases in cloud computing environments", in Proc. IEEE Int. Journal. Granular Comput. (GrC), Kaohsiung, Taiwan, pp. 399–403.

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. 7, Issue 2, February 2019

3. Chunyan H., Hong S., Huaxuan Z. ,Shiping s. (2013), "The study of improved FP-growth algorithm in MapReduce" in Proc. 1st Int. Workshop Cloud Comput. Inf. Security, Shanghai, China, pp. 250–253.
4. Cong S, Han J., Hoeflinger J. and Padua D. (2005) "A sampling-based framework for parallel data mining", in Proc. 10th ACM SIGPLAN Symp. Prin. Pract. Parallel Program., Chicago, IL, USA, pp. 255–265.
5. Dean J. and Ghemawat S. (2008), "MapReduce: Simplified data processing on large clusters", Commun. ACM, vol. 51, no. 1, pp. 107–113.
6. Dean J. and Ghemawat S. (2010), "MapReduce: A flexible data processing Tool", Commun. ACM, vol. 53, no. 1, pp. 72–77.
7. Han E H., Karypis G. and Kumar V. (2000) "Scalable parallel data mining for association rules" , IEEE Trans. Knowl. Data Eng., vol. 12, no. 3, pp. 337–352.
8. Han J., Mao R., Pei J. and Yin Y. (2004), "Mining frequent patterns without candidate generation: A frequent-pattern tree approach", Data Min. Knowl. Disc., vol. 8, no. 1, pp. 53–87.
9. Hsueh S.C., Lin M.Y. and Lee P.Y. (2012), 'Apriori-based frequent itemset mining algorithms on MapReduce', in Proc Journal. Ubiquit. Inf. Manage. Commun. (ICUIMC), Danang, Vietnam, pp. 76:1–76:8.
10. Hsu T.J., Tsay J.Y. and Yu J.R. (2009), "FIUT: A new method for mining frequent itemsets", Inf. Sci., vol. 179, no. 11, pp.0 1724–1737.
11. Kitsuregawa M. and Pramudiono I. (2013), "Parallel FP-growth on PC cluster", in Advances in Knowledge Discovery and Data Mining. Berlin, Germany: Springer, pp. 467–473.
12. Kitsuregawa M. and Shintani T. (2014) "Hash based parallel algorithms for mining association rule"' , in Proc Journal. Parallel Distrib. Inf. Syst., Miami Beach, FL, USA, 1996, pp. 19–30.
13. Liang F., Kirsh I., Shi Z. and Yang L. (2011), "DH-TRIE frequent pattern mining on Hadoop using JPA", in Proc. IEEE Journal Granular Comput. (GrC), Kaohsiung, Taiwan,pp. 875–878.
14. Li E., Liu L., Tang Z. and Zhang Y. (2007), "Optimization of frequent itemset mining on multiple-core processor", in Proc. IEEE journal. Very Large Data Bases, Vienna, Austria, 2007, pp. 1275–1285.
15. Tang P. and Turkia P.M. (2006), "Parallelizing frequent itemset mining with FP-trees", in Proc. IEEE Journal. Comput. Appl., Seattle, WA, USA, pp. 30–35.
16. Yu K. and Zhou Y. (2010), "Parallel TID-based frequent pattern mining algorithm on a PC cluster and grid computing system", Expert Syst. Appl., vol. 37, no. 3, pp. 2486–2494.
17. Zhou L. (2010), "Balanced parallel FP-growth with MapReduce", in Proc. IEEE Youth Journal. Inf. Comput. Telecommun. (YC-ICT), Beijing, China, pp. 243–246.