Frequent Itemset Generation Using Three Level Map-Reduce

[1]Jhonny Panchal, [2].Vishwanath R. Hulipalled
[1.] *PG. Student, School of Computer Science Engineering, Reva University, Bangalore.*
[2.] *Professor, School of Computing and Information Technology, Reva University, Bangalore.*

Abstract— Over the years, the flow of data has caused a is a major shift in the mechanism of large scale data processing and computing architecture, due to the increasing computational power. The value of these large-scale data can be explored by a tool called Data Mining, which is a mechanism of searching patterns of data from large database set which can be used for various computations as per you needs. Using data mining the patterns of data is automatically discovered, the outcomes are predicted, actionable information is created, and the focus is mainly on large database and datasets.
These large data sets need to be load balanced by equally partitioning the data among the various nodes, and this is done by various algorithms for mining frequent itemset. The mechanism that enables automatic parallelization, fault tolerance, load balancing, and data distribution on large clusters is not present in the existing parallel mining algorithm for frequent mining itemset. To overcome this problem a parallel frequent itemset algorithm using the map reduce programming model was developed. In this algorithm there are three MapReduce job to complete the task of mining. Because the itemset of different lengths have different construction cost and composition, and on a cluster it is sensitive to data distribution. To improve this and increase performance in MapReduce algorithm , we have developed a work load balance Metric call FIUT, which can be used to measure and balance load across the various clusters and computing nodes. In this paper we compare which is the best algorithm to for mining frequent itemset.

Key words:-Data Mining, Hadoop, FP-Tree, FIU-Tree, Fidoop

I. INTRODUCTION

Frequent pattern mining technique has become one of the popular techniques for data mining and other data mining techniques like sequential pattern, correlation and association rule. Frequent pattern mining has a new set of standard in FP-growth. FP-growth has a better performance than previously used algorithm such as Apriori because of the compression of transaction database into FP-Tree which is an on-memory data structure. Parallel execution increases performance and is essential for data warehouse in large scale. A map reduce programming model called Fidoop is used for parallel Frequent Itemset mining. The data is made for efficient by partitioning n data using algorithm. These partitioning of data is carried out in Hadoop cluster and is necessary for high efficiency and scalability in cluster. There are three map reduce steps in Fidoop to complete the mining task. In the final step the map reduce job independently decomposes itemset and reducer performs the combination of operations by constructing small ultra metric trees and separately mining these trees. The load balancing across the computing nodes of the cluster is measured by developing a workload balance metric which increases the FiDoop performance.
Another efficient method for mining frequent itemset in a database is the Frequent Items Ultra metric Tree (FIUT)[12]. In FIU-Tree the efficient in obtaining the frequent itemset is enhanced by using a special frequent items ultra metric tree (FIU-Tree). In this paper we compare the FIUT and FP-growth algorithm and show how FIUT is more efficient when compared to FP-growth.

II. RELATED WORK

M. J. Zaki [1] This paper proposes how assorted issues like decision making in value-based information is taken care of by frequent itemset mining of parallel and distributed datasets. In the cloud, the various calculations in the form of a utility administration, gives us permission to crunch a large mining issue. Inspite of having a wide range of calculations for performing a visit itemset mining, none of it is suitable for the cloud, which requires a very large information structure which needs to be synchronized across the systems. The FP development is one of the best methods of calculation which is meant for liability visit itemset mining.

J. Dean and S. Ghemawat [2] In this paper focuses on data processing for a very big data cluster by proposing an efficient neighbor joint which uses Map Reduce k nearest k closest join (KNN join). The main intention of this join is to find our which neighbor is closest from a dataset, to each item in another dataset. This is one primitive operation which is usually done by many mining applications. To conclude with, the objects are bunched into gatherings by the mappers and independently on every gathering a KNN join is performed by the reducer.

S. Sakr, A. Liu, and A. G. Fayoumi [3] In this paper the data processing in large scale and map reduce is examined using a consistent tree pattern. The most successful method for recognizing the obscure interrelationship among the qualities of the spectra information and its properties is the key of as Association mining. According to this paper, the first step is to utilize the main request to understand the leaning the spectra information. The second step is to provide an idea to enhance the productivity and relevance of tenet mining and also a calculation to develop regular example trees (CFP).
Using the true information sets , the access to the CPU and I/o execution of interrelationship strategy is quantitatively analyzed. The results of this paper explains that its main function is to analyze the law of heavenly bodies that utilize this interrelation technique and find some relation among the physicochemical property and qualities of divine spectra information.

M.-Y. Lin, P.-Y. Lee, and S.-C. Hsueh [4] Appropriated Algorithm for Frequent Pattern Mining using Hadoop MapReduce Framework is proposed in this paper. This is mainly used the quick development of various business applications, and also to discover the relationship that is required to take care of large and scattered database. It also show the mining of Frequent item set, Bart G [10]
This paper projects a similar example which shows the calculation of utilization of Hadoop Map Reduce system. It also demonstrates the most efficient execution for a large database. The proposed calculation shoes the working of every neighborhood hub shares examples of regular header table. This reduces the correspondence slide among the configuration development along with the mining example
The processor unmoving time is reduced by this thing set tally. Hadoop Map Reduce system is successfully utilized to consider a number of computation movements. Investigations are done on a group of 5 PC hubs which shows the execution time of productivity when performed with different calculations. The result of this test shows that the calculation effectively handles vast database versatility.

X. Lin [5] MapReduce based appropriate parallel mining is illustrated in this paper. A time to time developed measure of the computerized data needs to be managed because of the exponential development of the data. The most prominent difficulty of the information mining is effectively and quickly locating the relationship among the various information. The most important procedure to find the incessant example is the Apriori calculation. To apply this strategy the database checking must be performed, to find out the tally of a numerous number of applicant set of items. The successful technique for speeding up the mining procedure can be performed by parallel and appropriate registering of itemset.
The solution to this problem which is Distributed Parallel Apriori is discussed in this paper. This technique replaces metadata with Transaction Identifiers (TID). It then takes the various elements of the tallies of set of items into thoughts and in this way it creates a workload which adjusts among the processors and reduces the unmoving time of the processor. To demonstrate this approach a bunch of PC with 16 registering hub is used and is executed with some parallel mining calculations. The test result f this approach shows how the proposed approach wins over the others while the base backing is very less.

L. Zhou, Z. Zhong, J. Chang, J. Li, J. Huang, and S. Feng [6] In this paper a calculative pattern for large set of database in distributed computational scenarios and which also balances the parallel FP growth is discussed. The most used calculation for finding the continuous examples is the FP development. The execution time and the memory prerequisites increases unexpectedly as the size of database and the base bolster decreases. Many scientists tried to overcome this issue by using many conveyed registers which are used to enhance the execution and adaptability. Finding of the valuable data from the database is the main aim of the information mining. The measurement of the traded information over the system is experienced by the execution of this information method.

L. Zhou, Z. Zhong, J. Chang, J. Li, J. Huang, and S. Feng [6] In this paper a calculative pattern for large set of database in distributed computational scenarios and which also balances the parallel FP growth is discussed. The most used calculation for finding the continuous examples is the FP development. The execution time and the memory prerequisites increases unexpectedly as the size of database and the base bolster decreases. Many scientists tried to overcome this issue by using many conveyed registers which are used to enhance the execution and adaptability. Finding of the valuable data from the database is the main aim of the information mining. The measurement of the traded information over the system is experienced by the execution of this information method.

S. Hong, Z. Huaxuan, C. Shiping, and H. Chunyan [7] This paper proposes DH-TRIE incessant example mining on Hadoop utilizing JPA using which we can study the FP growth. The FP-growth is an understood relentless case's estimation in data

mining when working with high-dimensional, vast scale information sets. It is otherwise called awesome multifaceted nature on memory for the recursively preparing. When all is said in done, FP-growth can't deal with substantial scale information set unless isolating an entire information set into little squares. Taking into account Hadoop, the open distributed computing show, a disseminated DH-TRIE regular example calculation utilizing JPA is proposed, which tackled the three issues (globalization, arbitrary compose and length). The calculation is indicated great adaptability and versatility by correlations with mahout venture. By connected to a virtualization stage Vega Cloud, the calculation will be utilized as a part of far-extending circumstances. Grahne G., J. Zhu [14] FP-Tree generation using FP growth, is one of the efficient algorithms for data mining

M. Riondato, J. A. DeBrabant, R. Fonseca, and E. Upfal [8] PARMA which is a random way of calculating the estimate of the associated rule in MapReduce is discussed in this paper. A random parallel wait for mining the most used set of item and Association rule is introduced in this paper. Parma executes this by making the various examples of the data set which is based on value and is running calculation on the mining separately and in concurrence.

The final result is usually a guessing of the appropriate arrangement as the PARMA usually revolves around the subsets of the database the accuracy and the certainty parameters are usually determined by the end clients and the PARNA mainly ensures that these parameters and the nature of the estimations are met. The calculation of the parallel and Map Reduce structure was found out and executed. The result of this trial shows that the nature of the various estimations is usually very higher than what was initially thought about.

The adaptability and the accuracy of PARMA is exhibited by testing a few set of data of varying sizes and nature. The results are compared with two parallel mining calculations which were already proposed in Map Reduce.

Sandy Moens,Emin Aksehirli and Bart Goethals [11]The main problem in mining of data problems is the Frequent itemset mining. There are multiple algorithms available for the mining of data like the Apriori, FP growth etc. These two algorithms mainly lack parallelism and load balancing. To overcome this issue, multiple algorithms with parallelism were proposed which used a feature called MapReduce, YalingXun, Jifu Zhang and Xian Qin [12]. R. Agrawal, T. Imieli´nski, and A. Swami [13] The various disadvantages of extracting the frequent itemset from large and distributed clusters of data is overcome by the Associate mining rules of items which are present in large database.

J. Dong, M. Han [15] Mining Frequent item sets is one the most crucial and fundamental task and is the most researched topic in data mining. Association rules are the most used technology for data mining which is the used for finding the set of details from huge databases. One of the most used algorithm to solve this problem is Apriori, but this algorithm is very time consuming. Savasere A., E. Omiecinski, S.B. Navathe [16] To overcome this disadvantage, various time efficient algorithms have been developed for mining of frequent itemset.

III. PROBLEM STATEMENT

Load Balancing is the main focus of the Parallel Frequent Itemset Mining Technique. The data distribution and partitioning is equally done among the various cluster nodes. The inappropriate data partitioning decision which is likely creates the redundant transactions and item set mining tasks are shown in this paper. Not only network traffic but also computing loads is affected by FIM's data partitioning. In addition to the issue of load balancing the data partitioning algorithm should pay attention to computing loads and networks. Grouping the highly relevant transactions into a data partition and therefore reducing the number of redundant transactions is a key idea of Fidoop-DP. The distribution and partitioning of large dataset across the various data nodes of a Hadoop cluster in order to reduce computing loads and network which is carried out by performing redundant transactions on remote nodes, is illustrated in the paper. The performance of parallel FIM on clusters is increased by Fidoop-DP algorithm.

IV. SYSTEM ARCHITECTURE

The system architecture consists of three processes, the upload process, preprocessing job and frequent itemset generation using FP-growth and FIUT foe developing the system protocol design and analysis. Collection of data under a privileged authenticated status from independent sources is a feature of this system. The Figure 4.1 shows the proposed system architecture in which the files are divided among the 2 servers where it is executed separately and then the main server combines the individual results to provide the final result.

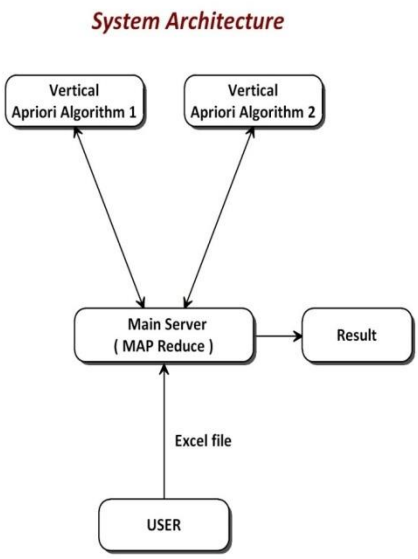


Fig 4.1 System Architecture

V. ALGORITHM

A. *Pseudo Code*

Step 1: Data set Initialization
f:file to upload
K: itemsets generation during the process of mining
n: all itemsets generated during the process of mining
T: tree (FIU) infrastructural tree with dynamic slotting
U: entire file for data mining from a data base
A: Selected Percentage of file for mining under a synchronized manner
B: Unselected Percentage of file for mining under a synchronized manner

Step 2: FIUT Tree Generation and Process initiation

FIUT (л) Tree Generation
T= {K};considering an instance of T for generation of K item sets under the given input sample file for each of K is as shown below.
Where, K= {K1, K2,K3,K4,….Kn}
Tree generation process
R: root node / R ∈ T && R ∈{K}
R→R1 , R2 / R1, R2 R ? && R1 , R2 !=R
R1, R2 ? M /M={M1 , M2, M3, M4…….Mn}
(M,R)=K

Final accessing of Load Balancing $(M, R) = K\sum_{i=0}^{n} \frac{(Ri)}{(Mi-Ri)}$ ----------(1)
Therefore,
$T = \int_0^n (K): (Ri)/(Mi - Ri)$ ----(2)

Step 3: Itemset Generation
On considering a dataset under database module of an desired data mining server under Hadoop single node cluster as
DBi= DataBase
∏= Transaction under DataBase
Tf ⊆ DBi / Tf < 20% (DBi)
For all
∏→ for i=0 to k
{
For each of 'k'→∏
∏i: fetch(Ki)
Call step 2: FIUT (∏)
}

Step 4: Analysis
Under the generated tree, a flatter analysis of groped datasets is considered and mined as,
FIUT (∏) ---------- (3)
$\{ \int_0^n (K): (Ri)/(Mi - Ri)$
Compute confidence and support graph.
Ration Selection (Ri)[0~1]
{
Generate,
$\{\int_0^n (K)^{\pi}$ ----------(4)
Show graph (display) ;}

B. *Methodology*

An implementation method or a programming model which is used for processing and generating of huge volume of sets of data is known as MapReduce. The user defines a map function which generates a key/value pair and the reducer combines all those values which are associated with their respective keys. Programs which consist of this pattern with be parallelized automatically and their execution happens n a very large cluster within the algorithms. Partitioning of the data, scheduling of the jobs, handling and type of machine faults are all taken care of by the run-time system. This makes the programming very easy and the programmer finds it very easy to use the MapReduce functionality in their programs. There are three jobs of MapReduce which are as follows:

1) *First job of MapReduce*

The main task of the first MapReduce job is to create frequent one itemset. The HDFS stores the entire partitioned database in to various input files, among all data nodes f the Hadoop cluster. The mapper then reads all the transactions

sequentially from the local input split in which all the transactions are kept safely in form of pair <Long Writable offset, Text Record>. Then the mapper generates local one itemset by computing the frequency of the items.

2) *Second job f MapReduce*

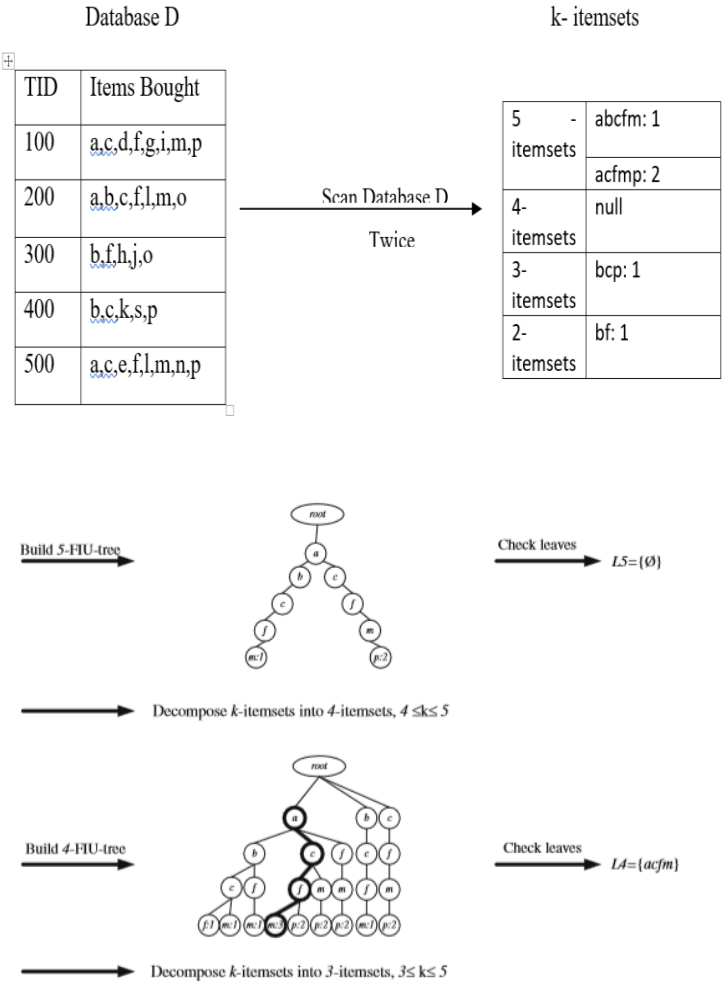
The frequent one itemset which is generated in the first job of MapReduce is sent to the second job where a second round of scanning is applied on the database to remove all the less frequently used items from each of the transaction record. The second job of MapReduce marks each of the itemset as k-itemset when if holds k frequent items.

3) *Third job of MapReduce*

The third job of MapReduce which is usually an expensive phase to compute comprises of:
Decomposing of itemsets
Constructing a k-FIU-Tree
Mining of the frequent itemset
Each mapper consists of a dual aspiration:
Decomposition of the k=itemset which is obtained by previous MapReduce job into small sized sets, in which the quantity og each set is between 2 to k-1
Construction of a FIU-Tree by combining the decomposition results which are of the same size.

4) *FIU-Tree*

The FIU-Tree is basically a hybrid data structure which is mainly designed for management of large data. Changing the hierarchy and morphing the nodes is the distinguished property of this tree. IN the FIUT, the nodes can interchange, for example the master can become a slave node, a slave can become a master node, a master can become a root node etc.
The norms of a complete binary tree were used to construct a FIUT. The expansion of the main principal of a binary tree is utilized and also bi-carts the correctness of all equivalent slave node which is terminated. The nodes distribution with regards to the keywords is performed.
The FIUT usually has t create a root with regards to the keyword it encounters in any location i=0. As the iteration continues, the configuration of the master node and the alignment of the root node are performed. In the due course of the process the FIUT re-alignment of the infrastructure and the reconfiguration of the node is performed based on the mining criteria.
An example of FIUT construction is illustrated in Figure 5,1



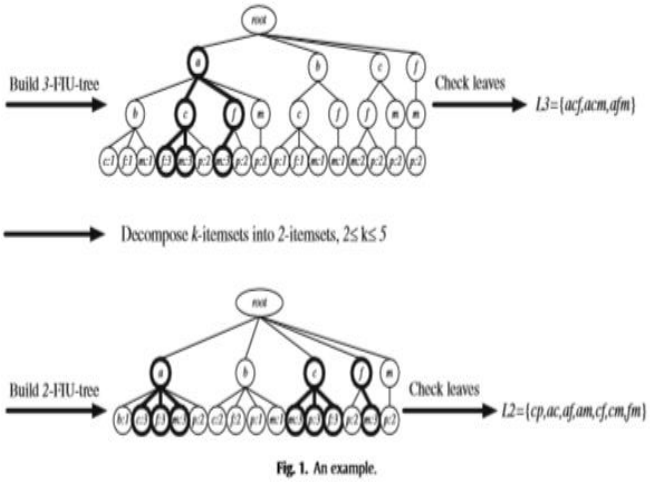


Figure 5.1 FIU-Tree construction

VI. RESULT ANALYSIS

The below graph Figure 6.1 shows the time taken by the FP growth algorithm to complete its execution for the various number of records

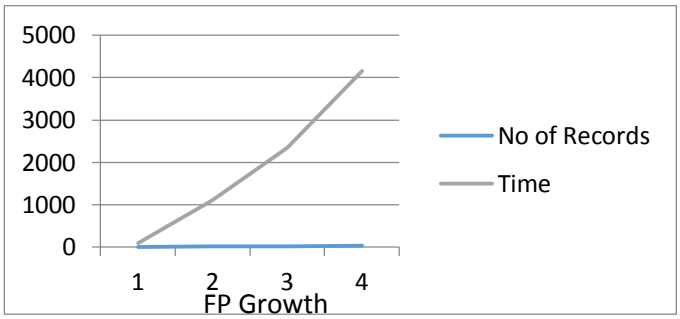


Fig 6.1 Time taken by FP Growth

The below graph Figure 6.2 represents the time required for FIUT to finish execution for various number of records.

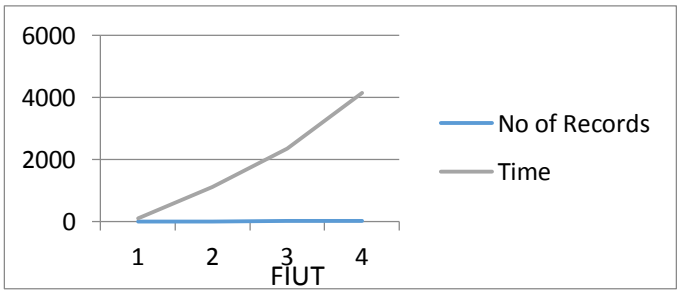


Fig 6.2 Time taken by FIUT

The efficiency of the FP Tree and the FIU- Tree algorithm to generate the frequent itemset is calculated by the time taken to generate it. The below table 6.3 shows the time taken to create the Frequent itemset of different sizes of the transactions. The able also proves the total time taken to generate the frequent itemset by the FP-tree algorithm is 1.5 times lesser than the frequent itemset for FIU-tree algorithm.

FP-Growth Tree				FIU-Tree			
Sl No	No of Records	Support Value	Time	Sl No	No of Records	Support Value	Time
1	10	01	100	1	10	01	38
2	15	01	1112	2	15	01	125
3	20	01	1139	3	20	01	1041
4	30	01	4157	4	30	01	1250

Fig 6.3 Performance Analysis with Varied Record Size

The below bar chart in the Figure 6.4, we show the comparison of the FP-Growth Tree while fetching the records of the different size which we enter. The outcome of the frequent itemset time generating is more fasters as co mpared to the FIU-Tree algorithm. Speed and security is main feature of proposed system.

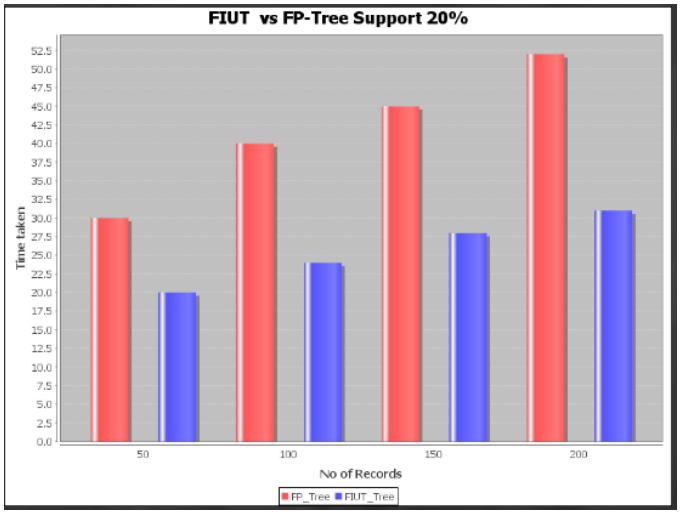


Fig 6.4 Frequent Itemset Generation Time For FIUT and FP -Growth

The below graph shows the comparison of the support level in reference with the Frequent item set generation.

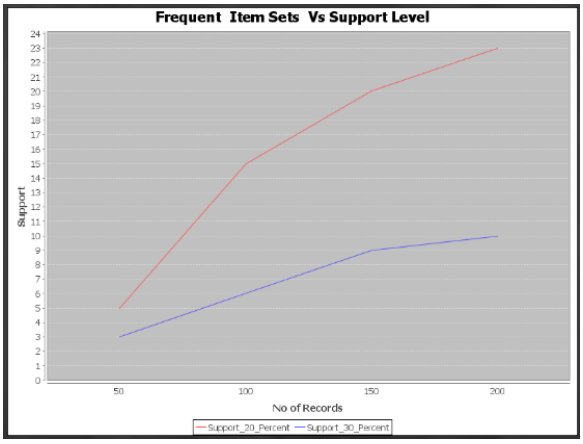


Fig 6.5 Frequent Itemset vs. Support Level

VII. CONCLUSIONS

To reduce the high level of communication and reduce the cost of FIM algorithms based on MapReduce, Fidoop DP is developed which shows the relationship between high-volume partitioning transactions and the various nodes of data in a Hadoop cluster. FiDoop-DP is capable of partitioning high similarity transactions and grouping highly correlated common elements into a list. Ability to reduce network traffic and workload is one of the most salient features of FiDoop-DP. It is done by reducing the number of redundant transactions that are transferred among the different nodes of Hadoop cluster.

FiDoop-DP applies data assignation system in lightweight of a Voronoi chart to accomplish associate information section within which LSH is incorporated to administer associate examination of the relation between's interactions At the hub of the Fidoop-DP is that the second MapReduce work, that isolate bulky catalog to structure a full dataset entry and perform FP-Growth create prepared on restricted partition to supply various patterns

Our experimental results show that FiDoop-DP considerably improves the FIM performance of the prevailing Pfp resolution by 31%, with an average 18% We tend to introduce a similarity metric during this study to change data-aware partitioning As a future analysis direction, we are going to use this metric to check advanced load-balancing methods on a heterogeneous Hadoop cluster

VIII. REFERENCES

[1]. M. J. Zaki, "Parallel and distributed association mining: A survey," IEEE Concurrency, vol. 7, no. 4, pp. 14–25, Oct. 1999.

[2]. J. Dean and S. Ghemawat, "Mapreduce: Simplified data processing on large clusters," ACMCommun, vol. 51, no. 1, pp. 107–113, 2008.

[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. IEEE Youth Conf. Inform. Comput. Telecommun., 2010, pp. 243–246.

[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. Inform. Security, 2013, pp. 250–253.

[8]. 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.

[9]. C. Lam, Hadoop in Action. Greenwich, USA: Manning Publications Co., 2010.

[10]. Bart G., "Survey on Frequent Pattern Mining", Manuskript, 2002.

[11]. Sandy Moens,Emin Aksehirli and Bart Goethals, " "Frequent Itemset Mining for BigData", intl. conf on Bigdata", IEEE 2013.

[12]. YalingXun, Jifu Zhang and Xian Qin, "Fidoop: Parallel mining of frequent Itemsets using MapReduce ", IEEE Trans.on sys.man and cybernetics, Vol. 46,No.3, March 2016

[13]. R. Agrawal, T. Imieli´nski, and A. Swami, "Mining association rules between sets of items in large databases," ACM SIGMOD Rec., vol. 22,no. 2, pp. 207–216, 1993.

[14]. Grahne G., J. Zhu, "Fast algorithms for frequent itemset mining using FP-Trees", IEEE Transaction on Knowledge and Data Engineering, 17 (10), pp. 1347-1362, 2005

[15]. Grahne G., J. Zhu, "Fast algorithms for frequent itemset mining using FP-Trees", IEEE Transaction on Knowledge and Data Engineering, 17 (10), pp. 1347-1362, 2005

[16]. Savasere A., E. Omiecinski, S.B. Navathe, "An efficient algorithm for mining association rules in large databases", in Proceedings of 21th International Conference on Very Large