FIDOOP-DP: DATA PARTITIONING IN FREQUENT ITEMSET MINING ON HADOOP CLUSTERS

Divya B M[1], Vidya Shree D.C[2], Rakshitha C.R[3], Shruthi H.H[4], Vanalakshmi A[5]
[1]Assistant Professor
Department of Computer Science Engineering, BGSIT College of Engg.

Abstract: Traditional parallel FP-growth algorithms for mining frequent itemsets aim to balance load by equally partitioning data among a group of computing nodes. We start this study by discovering a serious performance problem of the existing parallel Frequent Itemset Mining algorithms[1]. We address this problem by developing a data partitioning approach called FiDoop-DP using the MapReduce programming model with te use of bigdata tools. The overarching goal of FiDoop-DP is to boost the performance of parallel Frequent Itemset Mining on Hadoop clusters[1]. This algorithm enables a better Performance of hardware and interconnection in a multicore cluster system in the data mining application. The idea is to choose a proper task execution sequence combine with a communication scheduling that avoids the communication conflict in the interconnection network switch[2]. At the heart of FiDoop-DP is the diagram-based data partitioning technique, which exploits correlations among transactions. In corporating the similarity rock band metrics and the Locality-Sensitive Hashing technique, FiDoop-DP places highly similar transactions into a data partition to improve locality without creating an excessive number of redundant transactions[1]. We implement FiDoop-DP on a 24-node Hadoop cluster. FiDoop-DP significantly improves the performance of the existing parallel frequent-pattern scheme by up to 31 percent with an average of 18 percent[1].

Key words: Frequent itemset mining,Data partitioning model,mapreduced programming model, Hadoop frame work,Fp growth FIUT.

I. INTRODUCTION

Traditional parallel Frequent Itemset Mining techniques (a.k.a., FIM) are focused on load balancing; data are equally partitioned and distributed among computing nodes of a cluster. More often than not, the lack of analysis of correlation among data leads to poor data locality. The absence of data collocation increases the data mixing costs and the network overhead, reducing the effectiveness of data partitioning. In this study, we show that redundant transaction transmission and itemset-mining tasks are likely to be created by inappropriate data partitioning decisions. As a result, data partitioning in FIM affect not only network traffic but also computing loads. Our evidence shows that data partitioning algorithms should pay attention to network and computing loads in addition to the issue of load balancing. We propose a parallel FIM(Frequent Itemset Mining) approach called Fidoop-DP using the MapReduce programming model. The key idea of Fidoop-DP is to group highly applicable transactions into a data partition; thus, the number of redundant transactions is significantly slashed. Importantly, we show how to partition and distribute a large dataset across data nodes of a Hadoop cluster to reduce network and computing loads induced by making redundant transactions on remote nodes. FiDoop-DP is helpful to speeding up the performance of parallel FIM on clusters.

II. RELATED WORK

[1] This Paper proposes how frequent itemset mining finds much of the time happening itemsets in value-based information.This is connected to assorted issues, for example, decision backing, specific promoting, money related gauge and medicinal analysis. The cloud, calculation as a utility administration, permits us to crunch expansive mining issues. There are several calculations for doing visit itemset mining, yet none are out-of-the-crate suited for the cloud, requiring huge information structures to be synchronized over the system. The greatest calculations meant for liability visit itemset mining are the famous FP-development (Frequent Patterns development).We build up a cloud-empowered algorithmic variation on behalf of incessant itemset taking out that scale by means of almost no correspondence and computational overhead and even, with one and only specialist hub, is quicker than FP-development. We build up the idea of a postfix way and show how this permits us to bring down the communicational expenditure and prompts flexible jobsize. This idea gives an exceptionally adaptable algorithmic arrangement that can be connected to a large assortment of various issue sizes and setups.Consistent itemset mining discovers a great part of the time happening itemsets in quality based data. This is associated with different issues, for instance, decision backing, particular showcasing, budgetary figure and therapeutic examination.

[2] This paper proposes MapReduce is a programming model for handling and producing extensive information sets. We fabricated a framework around this programming model in 2003 to disentangle development of the upset list for taking care of hunts at Google.com. From that point forward, more than 10,000 particular projects have been actualized utilizing MapReduce at Google, including calculations for extensive scale diagram handling, content preparing, machine learning, and factual machine interpretation. The Hadoop open source execution of MapReduce has been utilized widely outside of Google by various associations.

[3] This paper proposes Efficient get ready of neighbor joins using Map Reduce k-nearest k closest neighbor join (kNN join), intended to discover k closest neighbors from a dataset

S for each itemset in another dataset R, is a primitive operation generally received by numerous information mining applications. To sum things up, the mappers posy objects into gatherings; the reducers perform the kNN join on every gathering of items independently. We outline a workable mapping instrument that try to pruning rules for separation sifting, and thus diminishes both the rearranging and computational expenses. To decrease the rearranging cost, we propose two rough calculations to minimize the quantity of reproductions. Broad inquire on our in-house group exhibit that our proposed strategies are proficient, strong and adaptable.

III. SCOPE OF THE PROJECT

1)We made a complete update to FIUT (i.e., the frequent itemsets ultrametric trees technique), and tended to the execution issues of parallelizing FIUT.
2)We built up the parallel continuous itemsets mining strategy(i.e.,Fidoop) utilizing the MapReduce programming model.
3)We proposed an information conveyance plan to adjust load among computing hubs in a bunch.
4)We further upgraded the execution of Fidoop and decreased running time of preparing high dimensional datasets.
5)We directed broad trials utilizing an extensive variety of manufactured and certifiable datasets, and we demonstrate that Fi.Doop is productive and adaptable on Hadoop groups.

IV. SYSTEM DESIGN

A meaningful representation of the system to be developed in any research work is known as design. The interaction among the modules requires high I/O and multiple threads. The main examination is to make the model and the system more like-minded so that both the entity proves to be efficient. The process by which this task is carried out merge the standard result based o case studies which are a set of input to this research.

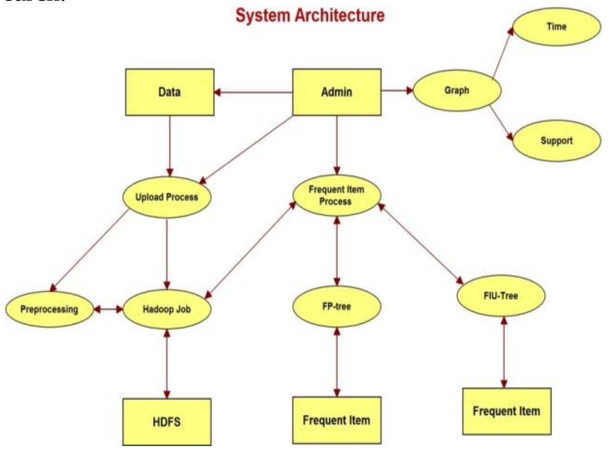

Fig : System Architecture

Fidoop architectural overview is been focused and demonstrated in the following section; fig projects the architected diagram. The system architecture consists of a file upload process, pre-processing job, generation of frequent item sets using FP-growth and FIUT technique for development of system protocol design and analysis. This system is featured to collect the data from the individualistic sources under a prosperous authenticated status. The graph generated to shows the time taken by the two algorithms to do the frequent item set process based on the support value and number of records. The graph projects the overall level on developing and designing the system requirement as per the resource availability. In our proposed system we have discussed about online retail. Each time a demand system is generated and thus its obtained results are analyzed and added.

V. IMPLIMENTATION

Implementation is the phase of the project where the theoretical outline is transformed into a working framework. At this step the principle workload and the real effect on the current existing framework schedule to the client division. If the implementation is not consciously arranged and controlled, it can bring about confusion and mystification.
The execution stage requires the accompanying responsibilities,

- Careful planning.
- Analysis of framework and limitations.
- Design of strategies to accomplish the substitution.
- Evaluation of the substitution strategy.
- Correct choices with respect to option of the platform.
- Suitableoption of the dialect for application advancement.

The process of converting a standard system prototype into a practically working module is demonstrated in this section; basically a Hadoop cluster is installed under Ubuntu environment.

5.1 METHODOLOGY

1 First MapReduce Job : The first MapReduce job is responsible for creating all frequent one-itemsets. A t.ransaction d.atabase is partitioned into multiple input files stored by the HDFS across data nodes of a Hadoop cluster.

2 Second MapReduce Job : 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 in frequent items from each transaction record.

3 Third MapReduce Job : The third MapReduce job a computationally expensive phase is dedicated to:
1) Decomposing itemsets;
2) Constructing k-FIU trees;
3) Mining frequent itemsets.

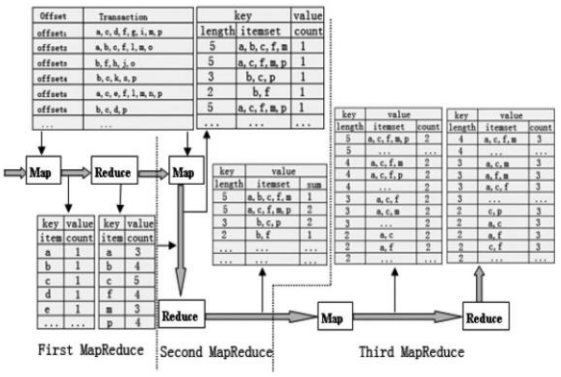

Fig : Over.view o.f MapRed.uced-b.ased Fi.Doop

VI. CONCLUSION AND FUTURE ENHANCEMENT

The current frequent itemset mining algorithm is facing big challenges in load balancing, efficiency and scalability, to overcome these challenges our proposed system uses MapReduce techniques and Hadoop. Our proposed algorithm performs three MapReduce jobs to get Frequent Item sets mining job. The data is composed from UCI datasets which are pre-processed and file uploaded into Hadoop .When frequent item sets mining algorithm is invoked, the datasets are fetched from Hadoop and mining algorithms are processed on data and produced frequent itemsets mining. Fidoop system has been committed to produce an accurate data mining results under Hadoop single node cluster environment,

The system records high efficiency gain for providing stable information resources for dynamic and critical data under big data mining. Results are detailed and discussed in previous chapters with overall system design and analysis. This System is developed as a web based application in MVC architecture on Java platform. For comparison FP-Tree and FIU-Tree techniques are used.The present dataset used belongs to online retail, in future we can use different datasets from different verticals, also we can develop mobile app to get results in our hand. This system in future can be enhanced with a political thought analysis and redefine process of computation under big data environment.

REFERENCES

[1] Rizwana Kowsar M.S and Somesekhar T “Data Heirarchy Mining under MapReduce Techniques for Frequent Itemsets”2016.
[2] Rizwana Kowsar M.S and Somesekhar T “Dataset based MapReduce technique under Chronic Mining and Confidence analysis”2016.
[3] W.Lu, Y.Shen, S.Chen, and B.C. Ooi, “Efficient processing of k nearest neighbor joins using MapReduce,”2012.
[4] J.Zhang, X.Zhao, S.Zhang, S.Yin, and X.Qin, “Interrelation analysis of celestial spectra data using constrained frequent pattern trees,”2013.
[5] K. Yu and J. Zhou, “Parallel TID-based frequent pattern mining algo- rithm on a PC cluster and grid computing system,”2010.
[6] “Distributed Algorithm for Frequent Pattern Mining using HadoopMap Reduce Framework” Suhasini A. Itkar1, Uday V. Kulkarni2 1.2013.
[7] K.-M. Yu, J. Zhou, T.-P.Hong, and J.-L. Zhou, “A load-balanced dis- tributed parallel mining algorithm,”2010.
[8] K. W. Lin, P.-L. Chen, and W.L. Chang, “A novel frequent pattern mining algorithm for very large databases in cloud computing environ- ments,”2011.