Simplified Parallel Data Processing on Large Weighted Itemset using Mapreduce

Sunitha S[1], Sahana Devi K J[2]
[1, 2] Department. Of Computer Science and Engineering
[1, 2] EWIT

Abstract- *the collections of data are often wont to growth and accessibility of the information. Its necessary to have been with business and for the society. frequent item set mining is associate with the data processing techniques. That has been exploited to the itemset extract perennial co-occurrences. That knowledge items, the many several application are contexts things enriched with weights denoting with their relative weight that importance within the analysed knowledge, pushing item with weights into the itemset mining method, In the itemset mining algorithms are obtainable, particular in literature, there's an, absence of parallel and distributed solution that are ready to scalable towards massive clusters of weighted knowledge.*

Keywords- Clustering, Classification, Data mining, Mapreduce concept.

I. INTRODUCTION

In massive knowledge the data comes from multiple, heterogeneous, autonomous sources and unendingly growing. Its completely for different characteristics like its massive volume, velocity supply with parallel and distributed with the centralised management, get to explore advanced and evolving relationship among with the data[1]. These are completely different characteristics of challenge for the locating helpful data or information from it. The revolution of business and scientific technologies.

ICT architecture to treat with large amount of information. as an example, e-commerce platform, social networks systems, and search engines normally want for storing and process terabytes of information. The non inheritable knowledge area of the unit large-scale, heterogeneous, and advertising with e-commerce corporations unendingly ought to store and analyze countless electronic transactions created by their customers morever as compelling want sensible large-scale data. Hence, within the previous few years an increasing supervised and unsupervised data can be processing algorithms are able (eg., classification algorithm[2], and that aggregation algorithms[1] that area unit ready to scale towards massive knowledge. for eg. driver[4] and MLIB[5]. Since most traditional algorithms are not able towards with the scale towards Big data, in research effort has been devoted towards the developing new large-scale item set mining algorithm techniques are analyzed with the data. A weight is can be associated with the each data item and local significance within each transaction. Since, to the best of our knowledge, existing towards the large-scale item set algorithm are unable to consider item weight during the mining process, the recent developement in technology, science, user habit, etc. gave rise to production and storage of the large amount of that knowledge, not satisfactory, the intelligent analysis in main memory. in such cases, one answer is to program[5] , wherever frequency investigating is achieved by reading the dataset over and another time for every size of the candiate itemsets. The memory needs for handling the entire set of candiate itemset blows up quick and rendors Aproiri based mostly schemes terribly inefficient to use on single machines. Secondly, current approaches tend to stay the output and runtime in restraint by increasing the minimum frequency threshold, machanically reducing the quantity of candiate and frequent itemsets. However, studies in recommendation system have that itemsets with lower frequencies rea unit of a lot of[12]. Therefore, we have to tendency to still see that tranparent with lower frequencies are unit a lot of the large amount of knowledge, that is created and consumed by lot of the each day. algorithms are classified into two majour sub categories: shared memory and distributed.

parallel and ditributed computing supply a possible answer for the on top of issues if the economical and scalable parallel and distributed solution to by means of thefault-tolerence by that manner[3]. Hadoop-mapreduce are going to establish the Mapreduce programming and hadoop framework. in hadoop framework it made to organized by the Mapreduce.

II. PROBLEM STATEMENT

The problem of extracting the frequent itemsets from Big Weighted Datasets. Although there are many in-memory weighted itemsets with mining algorithms available in literature (e.g., [5], [6], [7], [8]), to the simplest of our information no parallel and distributed answer for mining frequent weighted with the itemsets from huge Weighted knowledge has never been projected to the date. we tend to

propose the itemsete into the Parallel Weighted Itemset labourer(Parallelly), a along with the replacement parallel and distributed framework to extract frequent weighted itemsets parallelly from probably terribly massive transactional datasets are enriched with item weights of itemsets [19]. The framework depends on a parallel associate degreed distributed-based implementation running on an Hadoop cluster [10]. to create the itemsets mining method scalable towards huge knowledge, most analytical steps performed by the system are mapped to the MapReduce programming paradigm.

III. RELATED WORK

A. Apiriori algorithm

Apriori is the most classic and most generally algorithmic rule for the mining frequent itemsetsin the existing system to Mathematician association rules, proposed by R. Agrawal and R. Srikant in 1994 [2]. rule The pseudo-code of Apriorri algorithmic exploition is made Apriori is understood to be less ascendible than projection-based and vertical algorithms. Apriori[2] is that most established with algorithmic rule or finding frequent itemset from a transactional dataset; but, it must scan the dataset persistently and to come up with several candiate itemsets, once the dataset size is large, each memory use the unit of machine price will still be terribly high-ticket.

Accordingly the Parallel and distributed computing provide a large weighted items possible resolution for the on itemset top of issues if itemsets in the economical and ascendible through the weights of parallel anddistributed algorithmic rule are oftend. Such simple and economical results of the itemset implementation are often achieved by exploitation Hadoop-MapReduce that model that may be a programming model for simply and writing to the applications that method large quantity of the items of knowledge in-parallel on massive datasets.

B. Mapreduce model

MapReduce is one of the earliest and best renowned models in parallel and distributed data computing space, created by Google in thr year 2004, supported C++ generating giant knoeledge sets in a very massively parallel and distributed manner[4]. Ithis techniques used for the proposed system, is an one in every qualities of Mapreduce model is its simplicity: a Mapreduce application job consists of two function, Map operates that processes a key/value try to come up with a group of intermediate key/value pairs, and also the cut back operates that merges all intermediate values related to an equivalent intermediate algorithm are enforced supported with the Mapreduce model[5,6,7].The main job of these two function are sorting and filtering input data. During Map phase data is distributed to mapper machines and by parallel processing the subset it produces the <key,value> pairs for each record. Next Shuffle phase is used for repartitioning and sorting that pair within each partition. So the value corresponding same key grouped into {v1, v2,....}values.

C. Why to use Hadoop Framework

Hadoop ia an Apache open source framework written in java that allows distributed processing of large datasets across the datasets of computers using the simple programming models. It's designed to rescale from single servers to the thousands of machines, every providing native computations and storage. istead of accept hardware to deliver high-availability service on prime of a datasets controles the computers. the mapreducing model, Hadoop Disrtibted File System(HDFS); the distributed storage model that desighned once Google filling system(GFS).Currently it supports extra models and system such as HBase, ZooKeeper; a superior coordination service for distributed application, and pig; a level data-flow language[9].

3.4 BIGFIM Algorithm

It is a proposed algorithm and it addresses two complementary research topics:

(i) large-scale itemset mining and
(ii) weighted itemset mining.

Large-scale itemset mining. Frequent itemset and association rule mining are the techniques can be used for the large-scale itemsets have been devoted to finding the processing data can be able to the parallel and distributed system completely different on the Apriori algorithm. The BigFIM can be asscіated with the entails subsequent steps; intialize, associate rule can be able to that can be degree horizontal set of the information distributed.

To balance the computation load on completely different machines the authors in[5] projected to think about the support of singletons, where as the work best owed into that is[6] and[7] exploited techniques and k elements.

Hadoop-MapReduce is a programming model for easily and efficiently writing applications which process vast amount of data (terabyte or more data sets) in-parallel on large clusters of commodity hardware in a reliable, fault-tolerance manner. A MapReduce program (job) partitions the input

dataset into independent splits which widely can be exploratory data mining techniques which were first introduced in [6]. To scale towards large datasets most significant efforts have beendevoted to studying parallel and distributed itemset mining strategies. For example, in [3] the authors exploited a prefix-tree-like structures to drive the parallel itemset mining process. The mining process entails the following steps: first, an horizontal subset of the data is analyzed and a local FP-Tree is built; then the itemset mining process is performed on the local FP-Tree. Finally, the candidate pattern bases from different processing flows are then merged together.

.

An attempt to discover misleading patterns from Big datasets using MapReduce has been made in [10]. The idea is to compare frequent (unweighted) itemsets mined at different Data abstraction levels to highlight potentially critical situations. Unlike all the aforesaid approaches, this paper addresses weighted itemset mining instead of traditional itemset mining. To scale towards towards Big Data, the Parallel framework relies parallel and that the distributed-based implementation running on an Hadoop cluster [10], where most mining steps are mapped to the MapReduce programming paradigm [9]. Weighted itemset mining. In the traditional itemset and rule mining tasks items belonging to each transaction of the source dataset are treated equally. To differentiate items based on their relevance within each transaction, in the authors first addressed the issue of mining more informative association rules, i.e., the Weighted Association Rules (WAR).

fig 1: architecture for parallel retrieved itemset.

WARs are association rules enriched with weights denoting item significance. Weights were introduced only during the rule generation step after performing the traditional frequent itemset mining process. To improve the efficiency of the mining process, the authors in [6] pushed item weights deep into the itemset mining process by exploiting with the anti-monotonicity of the weighted support measure in an Apriori-based itemset mining process [9]. In [8] a FP-Growth-like weighted itemset mining algorithm process is presented. Unlike [5], [6], the algorithm proposed in [8] extracts infrequent (rare) itemsets rather than frequent ones. A parallel issue is the extraction of weighted itemsets and rules when coping with data not equipped with preassigned weights. For example, to generate appropriate item weights, in [7] the dataset is modeled as a bipartite can be

hub-authority graph and evaluated by means of a well-known indexing strategy.

IV. PARALLEL WEIGHTED ITEMSET MINING FROM BIG DATA

Parallel Weighted Itemset Miner (Parallel) is a new data mining environment aimed to analyze Big Data equipped with item weights. The main environment blocks are briefly introduced below. A more detailed description is given in the following sections.

A. Data Processing.

To prepare the source data to the itemset mining process, data are required, enriched with item weights, and transformed using the established preprocessing techniques (e.g., data filtering). The result is stored into an HDFS data repository

B. Weighted itemset mining process.

This step entails the extraction of all frequent weighted and unweighted itemsets from the prepared datasets. To scale towards Big data, the extraction process relies on a parallel and distributed-based itemset miner running on an Hadoop cluster.

C. Itemset ranking.

To ease the manual exploration of most interesting patterns, the results of the weighted and unweighted itemset mining process are compared with each other and most interesting patterns are selected based on a new quality measure which combines

traditional with weighted with the support counts.

V. FREQUENT ITEMSET MINING ON MAPREDUCE

We propose two new methods for mining frequent itemsets in parallel on the MapReduce framework where frequency thresholds can be set low. Our first method, called Dist-Eclat, is a pure Eclat method that distributes the search

space as evenly as possible among mappers. This technique is able to mine large datasets, but can be prohibitive when dealing with massive amounts of data. Therefore, we introduce a second, hybrid method that first uses an Apriori based method to extract frequent itemsets of length k and later on switches to Eclat when the projected databases fit in memory. We call this algorithm BigFIM. even while finding long frequent patterns. In contrast, Apriori, the breadth-first approach, has to keep all k-sized frequent sets in memory when computing k+1-sized candidates. Secondly, using diffsets limits the memory overhead approximately to the size of the original tid-list root of the current branch [10].

Table I:EXAMPLE OF UNWEIGHTED DATASET: ITEM BOUGHT BY CUSTOMERS

Customer id	Purchased items
1	X,Y,Z
2	X,Y,Q
3	X,Y,Z
4	X,Y,Q
5	X,W,Z

Table II:EXAMPLE OF WEIGHTED DATASET: ITEM RATING GIVEN BY CUSTOMERS

Customer id	Purchased items
1	<X,3> <Y,1> <Z,5>
2	<X,2> <Y,2> <Q,2>
3	<X,4> <Y,3> <Z,5>
4	<X,3> <Y,3> <Q,2>
5	<X,2> <W,5> <Z,4>

Table 1: Example for Weighted and unweighted itemsets for the customers

The above example shows the example for the itemsets can be retrieve the parallel itemset with weights.

Definition: I: Weighted support. Let D be a weighted transactional dataset, I be a weighted itemeset, <i, w>, I and f an arbitrary aggregation function defined by the items with weights.

The weighted support of I in D is defined as

$$\text{wsup}(I, D) = \sum_{t_q \in \mathcal{T}(I)} f_{j,k,z | i_j, i_k, \ldots, i_z \in I}(w_j, w_k, \ldots, w_z)$$

Distributed tracsaction splitting.

BigFIM relies on the items miners, the equvalence set of weighted transaction is given in[12]. In that two discrete transactions have disjoint splitting process is paralleized.

Weighted support counting

To obtain the tid-list for this node, so, at most the tids that can be subtracted on a complete branch extension is the size of the original tid-list. Dist-Eclat operates in a three step approach as shown in Figure 1. Each of the steps can be distributed among multiple. The weights consists of <transaction id, weight> .

VI. CONCLUSION

This presents the datasets parallel and distributed resolution data to the matter of extracting the frequent to itemsets from massive Weighted items of Datasets. The dataset running on the Hadoop cluster, overcomes the constraints of state-of-the art approaches in dealing with datasets enriched with the item weights. during this paper we have a tendency to studied and enforced frequent itemset mining on algorithms for the MapReduce paradigm. Apriori focuses on speed by employing a straightforward load equalisation of the theme supported k-FIs. In the second rule, BigFIM, algorithm focuses on the mining terribly giant databases by utilizing a hybrid approach. k-FIs area unit deep-mined by the Apriori variant itemsets unit distributed to the mappers. At the mappers frequent itemsets area unit deep-mined victimisation mapreduce.

REFERENCES

[1] D. Agrawal, S. Das, and A. El Abbadi, “Big data and cloud computing: Current state and future opportunities,” in Proceedings of the 14th International Conference on Extending Database Technology, ser. EDBT/ICDT ’11. New York, NY, USA: ACM, 2011, pp. 530–533. [Online]. Available: http://doi.acm.org/ 10.1145/19513 65.1951432

[2] H. T. Lin and V. Honavar, “Learning classifiers from chains of multiple interlinked RDF data stores,” in IEEE International Congress on Big Data, BigData Congress 2013, June 27 2013-July 2, 2013, 2013, pp. 94–101. [Online]. Available: http://dx.doi.org/10.1109/BigData.Congress.2013.22

[3] J. Hartog, R. Delvalle, M. Govindaraju, and M. J. Lewis, “Configuring a mapreduce framework for performance heterogeneous clusters,” in 2014 IEEE International Congress on Big Data, Anchorage, AK, USA, June 27

– July 2, 2014, 2014, pp. 120–127. [Online]. Available: http://dx.doi.org/10.1109/BigData.Congress.2014.26

[4] S. Moens, E. Aksehirli, and B. Goethals, "Frequent itemset mining for big data," in SML: BigData 2013 Workshop on Scalable Machine Learning. IEEE, 2013.

[5] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules in large databases," in VLDB'94, Proceedings of 20th International Conference on Very Large Data Bases, J. B. Bocca, M. Jarke, and C. Zaniolo, Eds. Morgan Kaufmann, 1994, pp. 487–499.

[6] S. Moens, E. Aksehirli, and B. Goethals, "Frequent itemset mining for big data," in SML: BigData 2013 Workshop on Scalable Machine Learning. IEEE, 2013.

[7] M. Malek and H. Kadima, "Searching frequent itemsets by clustering data: Towards a parallel approach using mapreduce," in Web Information Systems Engineering - WISE 2011 and 2012 Workshops - Combined WISE 2011 and WISE 2012 Workshops, Sydney Australia, October 12-14, 2011 and Paphos, Cyprus, November 28-30, 2012, Revised Selected Papers, 2012, pp. 251–258.

[8] Paul S. & Saravanan V. (2008). Hash Partitioned Apriori in Parallel and Distributed Data Mining Environment with Dynamic Data Allocation Approach. Proc. of the International Conference on Computer Science and Information Technology (ICCSIT '08). Singapore, IEEE: 481 – 485.

[9] Yu K. & Zhou J. (2008). A Weighted Load-Balancing Parallel Apriori Algorithm for Association Rule Mining. Proc. of the International Conference on Granular Computing (GrC '08). Hangzhou, IEEE: 756 – 761.

[10] Li N., Zeng L., He Q. & Shi Z. (2012). Parallel Implementation of Apriori Algorithm Based on MapReduce. Proc. of the 13th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel & Distributed Computing (SNPD '12). Kyoto, IEEE: 236 – 241.

[11] Yang X.Y., Liu Z. & Fu Y. (2010). MapReduce as a Programming Model for Association Rules Algorithm on Hadoop. Proc. of the 3rd International Conference on Information Sciences and Interaction Sciences

[12] L. Zhou, Z. Zhong, J. Chang, J. Li, J. Huang, and S. Feng. Balanced parallel FP-Growth with MapReduce. In Proc. YC-ICT, pages 243–246, 2012.