ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 11, November 2013

PARALLELIZED COMPRISING FOR APRIORI ALGORITHM USING MAPREDUCE FRAMEWORK

A.Pradeepa[1], Dr.Antony selvadoss Thanamani[2]

Research scholar , Computer Science(Aided), NGM College, Pollachi, India [1]

Head & Associate Professor , Computer Science(Aided), NGM College, Pollachi, India [2]

Abstract: An integrating classification and association rule mining can produce more efficient and accurate classifiers than traditional techniques. The recently introduces MapReduce based association rule mining for extracting strong rules from large datasets. This mining is used later to develop a new large scale classifier. Map Reduce simulator was developed to evaluate the scalability of proposed apriori algorithms on MaReduce. The developed associative rule mining inherits the MapReduce scalability to huge datasets and to thousands of processing nodes. For finding frequent item sets, it uses hybrid approach between miners that uses counting methods. The new miner generates same rules that usually generated using apriori algorithms. Map Reduce classifier that based MapReduce associative rule mining. For the purpose of data mining to big data, parallel comprising this algorithm employs different approaches in rule discovery, rule from frequent itemsets, and rule pruning methods in these research fields. The present Map Reduce was developed to measure the scalability of MapReduce based applications easily and quickly, in this paper comprehensive to evaluate an accurate and effective classification technique, highly competitive and scalable if compared with other traditional and associative classification approaches.

Keywords: Data mining, MapReduce, MRApriori algorithm, Association rule mining.

I. INTRODUCTION

Data processing and knowledge discovery for massive data has always been an active research area in data mining [1]. There are many application associated with massive data, such as association rule mining [2], sequential pattern mining [3], text mining [4] and temporal data mining [5], traditional techniques among the many algorithm based on MapReduce. Dean and Ghemawat from Google firstly presented a parallel comprising model in Map Reduce, which was a framework for processing huge data sets on certain kinds of distributable problems using an associations rule. Data mining, a technique to understand and convert raw data into useful information, is increasingly being used in a variety of fields like marketing, business intelligence, scientific discoveries, biotechnology, Internet searches, and multimedia. Data mining is an interdisciplinary field combining ideas from statistics, machine learning, and natural language processing. Data mining in such environments requires a utilization of the available resources. Map Reduce is a popular computing model for

implemented in several systems. There are many research papers related to MapReduce combined with the traditional techniques [6]. MapReduce is an emerging programming model to write applications that run on distributed environments. Several implementations such Apache Hadoop are currently used on clusters of tens of thousands of nodes [7]. MapReduce design and the implementation of data mining techniques relating to associative rules. This trend to use distributed complex, heterocomplex, heterogeneous computing environments has given rise to a range of data mining research challenges. This method explores the different methods and trade-offs when designing and implementing distributed data mining algorithms.

II. OVERVIEW OF MAPREDUCE CONCEPT

The parallel data processing system called MapReduce. Jeffrey and Lammel [8] introduced the easy and abstracted programming model, Map Reduce. Many computation problems can be expressed using this model. It is inspired by functional programming languages. The input and output data have a specific format of key/value pairs. The users express an algorithm using two functions: the Map functions and the Reduce function. The Map function is written by the application developer. It iterates over a set of the input key/value pairs, and generates intermediate output key/value pairs. The Map Reduce library groups all intermediate

ISSN (Print) : 2319-5940
ISSN (Online) : 2278-1021

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 2, Issue 11, November 2013

values by key and introduces them to the reduce function. The Reduce function is also written by the application developer, it iterates over the intermediate values associated by one key. Then it generates zero or more output key/value pairs. The output pairs are sorted by their key value.

III.MAPREDUCE FRAMEWORK

MapReduce [9] is a linearly scalable programming model. The programmer writes two functions a map function and a reduce function each of which defines a mapping from one set of key-value pairs to another. MapReduce framework offers clean abstraction between data analysis task and the underlying systems challenges involved in ensuring reliable large-scale computation. The MapReduce framework and the Hadoop distributed file system are running on the same set of nodes.

IV.MAPREDUCE IMPLEMENTATIONS

While the programming model is abstracted, it is the job of the implementation to deal with the details of parallelization, fault tolerance, data distribution, load balancing, etc. The Apache Hadoop [10] is the most popular and widely used open-source implementation of Google's MapReduce. It is written in Java for reliable, scalable, distributed computing. The code is available as the Apache License Version 2.0 [11]. Hadoop is being used by known enterprises e.g. Facebook, Yahoo, Amazon and many others [12].

V. MINING FREQUENT ITEMSET AND ASSOCIATIONS

Frequent items are patterns of itemsets or sequences that frequently appear in a data set. For example, a set of items in shopping basket, such as milk and bread that appear frequently together in a transaction data set is a frequent itemsets. Finding such frequent item plays an essential role in mining associations, correlations, and many other interesting relationships among data. Moreover, it helps in data classification as well. Several classifiers [13] [14] [15] [16] are built based on association rules. Thus, frequent pattern mining has become an important data mining task and a focused theme in data mining research. The Apriori algorithm is one of the first algorithms used for mining frequent itemsets to get association rules. It employs the mining Apriori property that subsets of a frequent itemset are also frequent items. It iterates over the data to generate frequent k-itemset candidates based on the frequent (k-1)-itemsets. Variations involving hashing [17] and transaction reduction can be used to make the procedure more efficient. Other variations include partitioning the data [18] (mining on each partition and then combining the results) and sampling the data [19] [20] (mining on a subset of the data). These variations can reduce the number of data scans required.

VI. MRAPRIORI REPRESENTATIONS ALGORITHM

MRApriori Algorithm using MapReduce framework. It can be considered as hybrid approach between Eclat [21] and apriori with HT pruning [17]. It first discusses distributing apriori in MapReduce. MapReduce for frequent itemsets counting in apriori specified good scalability for algorithm. However, repeated scanning of dataset is still needed. MRApriori eliminate the need to iterative scanning of the data to find all frequent items. Instead, MRApriori repeats scanning other intermediate data that usually keep shrinking per iteration. Number of iterations is same as number of iterations in Apriori. But usually, MRApriori scan less data rather than scanning whole data as in Apriori. Thus, the main differences between Apriori and MRApriori are:

- MRApriori do only one scan for the data in original format.
- MRApriori uses new data structure to represent the dataset.
- MRApriori uses batch set intersection using MapReduce framework where Apriori uses counting.
- MRApriori uses batch rule extracting based on MapReduce framework.

MRApriori consists of three steps, data initialization, frequent items discovery, and rule extraction for frequent items.

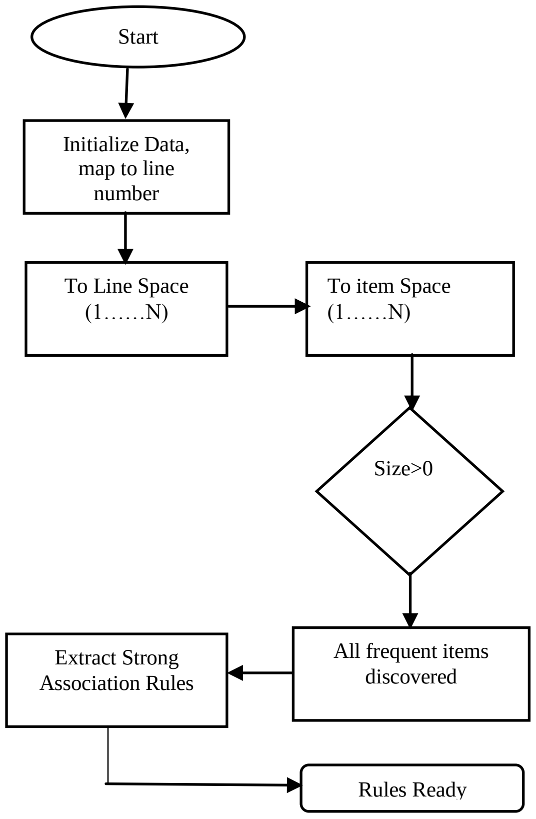
Start
Initialize Data, map to line number
To Line Space (1……N)
To item Space (1……N)
Size>0
All frequent items discovered
Extract Strong Association Rules
Rules Ready

Fig.1 Workflow of MRApriori Algorithm

Data Initialization

MRApriori uses integer values to represents the items in dataset. This makes the algorithm faster and takes less memory sizes. Mapping items into integer values can be delayed and merged to the step of finding frequent item sets of size one. Dataset consists of transactions or records. Each record contains several items. Items in each transaction may be spars.

Frequent Items Discovery and Rule Pruning

Frequent ruleitem discovery phase in MRApriori works by applying the support condition while repeating the transformation of the input data between the Line space and the Frequent Item space until discovering all frequent ruleitem. Data transformation from a Line space to a frequent space is performed using the MapReduce methods –"ToFrequent.Mapper" and –"ToFrequent.Reducer". The input for the ToFrequent.Mapper method is <line, list of ItemId>, and the output is <ItemId, Line>, which then gets inputted to the ―ToFrequent.Reducer‖ and this method outputs <ItemId, set of lines>.

Generate Strong Association Rules Form Frequent Itemsets

MRApriori has collected the set of all frequent items of all sizes that survived the support threshold. Then it follows the apriori definitions to extract strong associated rules from frequent item set. MRApriori also uses Map Reduce framework to extract significant rules form all frequent item sets.If all frequent item sets can fit in computer memory and if the processing time is not that big then Hash table data structure can be used to hold the data thrown from the map function. In this case, the key will be the left-part and the value will be set of (right-part: supp) entries for frequent item fi. In the distributed implementation of this step, data are thrown to distributed file system and the Map-Reduce middleware is responsible to sort the entries and to fetch them grouped to the reduce functions.

Algorithm Features

(i) All elements, either in line space or in frequent item space, are saved in one virtual collection that has same data structure. This produces simpler abstracted data that is easy to be serialized and to be distributed among the cluster nodes. Also, this helps to develop more abstracted algorithms such as MRApriori which does not impose restrictions on how to save and coordinate the distribution the data. Data chunks can have arbitrary sizes with no effect on MRApriori accuracy. This allows the underlined middleware (Hadoop in our implementation) to split the data dynamically to achieve load balancing execution with no accuracy consequences. This is an advantage over other algorithms that uses bagging [19] and boosting [20] are affected very much with the sizes of the splits in parallel implementations.

(ii) All candidate frequent items of all degrees are represented in the same way. In the special cases where number of attributes is less than hundreds of attributes, MRApriori can use binary format as to hold the values of dataset. Thus one integer number is sufficient to represent the ColumnIds and another integer is sufficient to represent RowId of the item. This is used heavily in MCAR algorithm [15]. In cases of all twenty datasets used in experiments from UCI [22], one integer number of 32 bits memory size was sufficient to represent any frequent item of any degree.

(iii) In MRApriori, all data are saved on file system. Processing the data is done in stream I/O reading. This is much faster than accessing the datasets in random access way.

VII. CONCLUSION

The MapReduce simulator that targets a Hadoop environment. This is due to the lack of tools to investigate the algorithms behavior on MapReduce. In this paper has presented and evaluated MRApriori, a distributed associative rule algorithm that capitalizes on the scalability, parallelism and resiliency of MapReduce for large scale frequent items discovery. MRApriori keeps reducing the data (using support thresholds) while transforming the data between the measures till it discovers all frequent items of long lengths. Then it derives association rules from discovered frequent items. MRApriori generates same number of rules generated by all association rule miners that use same support and confidence concepts. MRApriori jobs that run on Hadoop applications are naturally balanced and can optimize resource utilization in highly heterogeneous computing environments. MRApriori is open source and available for the community to download and to use for further investigation and development. Future work will involve multi support levels can be introduced to MRApriori as the intermediate data has the clarity and independence to apply different support levels on it per different iteration.

REFFERENCES

[1] J. Han, M. Kamber, Data Mining: Concepts and Techniques, second ed., Morgan Kaufman, San Francisco, 2006.
[2]P.Y. Hsu, Y.L. Chen, C.C Ling, Algorithms for mining association rules in bag databases, Information Sciences 166(2004) 31-47.
[3] Y.C Hu, G.H. Tzeng, C.M. Chen, Deriving two-stage learning sequences from knowledge in fuzzy sequential pattern mining, Information Sciences 159(2004) 69-86.
[4] K. Lagus, S. Kaski, T. Kohonen, Mining massive document collections by the websom method, Information Sciences 163(2004) 135-156.
[5] Y. Li, S. Zhu, X.S. Wang, S. Jajodia, Looking into the seeds of time: discovering temporal patterns in large transaction sets, Information Sciences 176(2006) 1003-1031.
[6] J. Berlin ska, M. Drozdowski, Scheduling divisible MapReduce computations, Journal of Parallel and Distributed Computing 71(2011) 450-459.
[7]O.O. Malley and A.C. Murthy, -Winning a 60 second Dash with a Yellow Elephant Hadoop implementation, March 2009, URL http://sortbenchmark.org/Yahoo2009.pdf.
[8] R. Lammel, - Google's MapReduce programming model - Revisited, Science of Computer Programming, vol. 70, 2008, pp. 1-30.

[9] J. Dean and S. Ghemawat, -MapReduce: simplified data processing on large clusters, Communications of the ACM, vol. 51, Jan. 2008, pp. 107–113.
[10]Apache Software Foundation., -Apache Hadoop, Jan.2010, URL ttp://hadoop.apache.org/.
[11]Apache Software Foundation, -Apache License, Version 2.0, 2004, URL http://www.apache.org/licenses/LICENSE-2.0.
[12]PoweredBy Hadoop, June 2010 URL http://wiki.apache.org/hadoop/PoweredBy
[13]B. Liu, W. Hsu, and Y. Ma, ―Integrating classification and association rule mining, Knowledge discovery and data mining, 1998, pp. 80–86.
[14] Z. Tang and Q. Liao, -A New Class Based Associative Classification Algorithm, Training, 2007, pp. 1-5.
[15] F. Thabtah, P. Cowling, and Y. Peng, -MCAR: multi-class classification based on association rule, Computer Systems and Applications, 2005. The 3rd ACS/IEEE International Conference on, 2005, pp. 33.
[16] F.A. Thabtah, P. Cowling, and Y. Peng, -MMAC: A New Multi-Class, Multi-Label Associative Classification Approach,‖ Proceedings of the Fourth IEEE International Conference on Data Mining, 2004, p. 217–224.
[17]J.S. Park, M.-S. Chen, and P.S. Yu, -An effective hash-based algorithm for mining association rules, Proceedings of the 1995 ACM SIGMOD international conference on Management of data - SIGMOD '95, New York, New York, USA: ACM Press, 1995, pp. 175-186.
[18]S. Brin, R. Motwani, J.D. Ullman, and S. Tsur, -Dynamic itemset counting and implication rules for market basket data, Proceedings of the 1997 ACM SIGMOD international conference on Management of data - SIGMOD '97, New York, New York, USA: ACM Press, 1997, pp. 255-264.
[19]L. Breiman, ―Bagging predictors‖ Machine Learning, vol. 24, Aug. 1996, pp. 123-140.
[20]Y. Freund, ―Boosting a Weak Learning Algorithm by Majority, Information and Computation, vol. 121, Sep. 1995, pp. 256-285.
[21]M.J. Zaki, S. Parthasarathy, M. Ogihara, and W. Li, ―New Algorithms for Fast Discovery of Association Rules, 3rd Intl. Conf. on Knowledge Discovery and Data Mining, vol. 20, 1997, pp. 283--286.
[22]C.L. Blake and C.J. Merz, ―UCI Repository of machine learning databases,UCI Repository of Machine Learning Databases, 1998, URL http://archive.ics.uci.edu/ml/.