Algorithm for Map/Reduce-based association rules data mining

Wenqi Wang [1,a], Qiang Li[2,b]

[1]Department of Early Warning Intelligence,Air Force Early Warning Acedamy Air Force Early Warning AcedamyWuhan,China

[2]Department of Early Warning Intelligence,Air Force Early Warning Acedamy Air Force Early Warning AcedamyWuhan,China

[a]wwq505@163.com,[b]ldxy-cj@163.com

Keywords: Apriori algorithm; Map/Reduce; parallel processing matting

Abstract. In order to realize massive information data mining, the traditional Apriori algorithm is updated into a Map/Reduce-based frequent itemsets generating method, so as to distribute the massive data into several servers for parallel processing. The construction of Hadoop platform helps to realize this method which is also compared with Apriori algorithm. The experimental results show that, in the process to generate frequent itemsets of massive data, this method can make full use of the advantages of parallel processing, followed by better timeliness.

Introduction

With the development of computer technology, cloud computing has become the future direction of development of distributed computing, and it is suitable for processing large-scale data, with high efficiency of calculation [1]. According to its basic concept, huge computing program is split into multiple subroutines through network, and, after analysis and processing by the system composed by a plurality of servers, the results are returned to users. Cloud computing is featured in high reliability, high expansibility, large scale and virtualization, thus it has important status and development space in the mass data processing. Generally, the programming mode of cloud computing is Map/Reduce which is proposed by Google and provides a simple model for writing codes needing large-scale parallel processing [2]. Hadoop is firstly developed as the basic platform of Nutch that is an open-source search engine project, and it helps the realization of Map/Reduce. As an open-source software platform, Hadoop uses a distributed file system (HDFS), and divides application into many small task blocks to be performed, so that it is easier to write and apply programs for massive data processing.

Apriori algorithm is used to mine frequent itemsets based on Boolean association rules, and it is an important algorithm for association rules mining. At present, the massive data mining raises requirements for parallel execution of classical algorithm in distributed computing environments, while cloud computing right provides environment for parallel execution of massive data mining. Therefore, according to Apriori algorithm, cloud-computing based Map/Reduce mechanism frequent itemset generation method is designed, then it is implemented on Hadoop platform, and finally its performance is analyzed.

Map/Reduce programming model

Map/Reduce is a programming model to transfer a large-scale distributed computing into a data key / value for serial distributed operating sets [3, 4]. A Map/Reduce computing consists of two stages, respectively map stage and reduce stage. The basic requirement to handle the data set (or task) with Map/Reduce is that the data sets to be dealt with can be decomposed into a series of small data sets all of which can completely realize parallel processing. In the stage of map, the Map/Reduce framework divides the input data into many data segments each of which is assigned with a map task that will calculate its assigned key/value, followed by an intermediate result, then,

of all the intermediate results, the values with the same key will be calculated and transferred to reduce function.

Each phased mission allows fault tolerant, if any one or more node errors arise in the process of calculation, the task will be automatically reallocated to other nodes. Simultaneous running multiple map and reduce tasks can realize good load balance and guarantee the cost of re-operating failed tasks is minimized as low as possible.

Analysis of Apriori algorithm

Apriori algorithm is the basic algorithm of frequent itemsets required in mining Boolean association rules, and it is also a very influential association rule mining algorithm; it is named according to the priori knowledge about properties of frequent item set. This algorithm uses a layer-by-layer search iteration method, generate (k+1) set with K-item set, through two main processing steps, namely connection and remove. The core of Apriori algorithm is the recursive algorithm based on two-stage frequent sets. In the process of this algorithm, all frequent sets are found out firstly, and then strong association rules are generated from frequent itemsets.

The procedure to find out all frequent itemsets with Apriori algorithm is as follows:

- The 1st frequent itemset L1 is generated according to the original transaction set.
- (k+1)th layer of candidate set is generated according to the frequent Kth set.
- Scan the transaction set, and find the (k+1)th layer of frequent sets.
- Repeat step 2 and 3, until the (k+1)th layer frequent set is empty.

The Apriori algorithm is advantageous in compressing the size of frequent set via Apriori properties, and improving mining performance.

Analysis of Apriori algorithm and its parallel implementation shows that three advantages are shown if Apriori algorithm is transplanted into Hadoop framework:

A. There is no memory bottleneck

In the process of Apriori algorithm, no large amount of memory is needed to generate intermediate data, so there is no memory bottleneck, then the computing nodes needn't be equipped with strong configuration; therefore, Apriori algorithm matches well with cloud computing environment.

B. The performance of distributed storage system can be made full use of.

The Apriori algorithm requires frequent transaction set scanning; Hadoop provides HDFS storage with good parallel reading-writing characteristics, thus the time to repeatedly read transaction set is greatly shortened compared with the traditional storage system, which makes it possible to mine large amount of data on Hadoop with Apriori algorithm.

C. Distribution of algorithm

The calculation with Apriori algorithm can be regarded as process of counting, and this process is suitable for Map/Reduce model, typical Map/Reduce model is used in web data frequency mining, and it's also a process of counting, so Apriori algorithm has the natural characteristics of Map/Reduce.

The above characteristics show that the combination of Apriori algorithm and Hadoop platform can display great advantage, but in the implementation process, the distribution of Hadoop framework and parallelization is self-regulated, so, in order to realize MR-Aprior algorithm with high scalability and good properties, the characteristics of Map/Reduce model and Hadoop framework need to be fully considered after overcoming the following three problems:

- Determination of appropriate key/value pair. The parallel distribution in the Map/Reduce distribution model is carried out based on key/value pair, so the basic means to enhance the efficiency of MR-Apriori algorithm is finding the most suitable key/value pair.

- Reduction of data transmission. HDFS provides strong ability of parallel reading and writing, but in actual application, if the data transmission in network can be reduced, the system performance can be greatly affected. Therefore, reduction of read transaction in Apriori algorithm will draw great influence on the efficiency of MR-Apriori algorithm.

Load balancing.In the Map/Reduce process, the load balancing of computing nodes is fully considered, which is also an effective way to improve the efficiency of MR-Apriori algorithm.

Map/Reduce-based frequent itemsets generation method

Apriori algorithm [5, 6] will find all frequent itemsets through repeated scanning database; in case of existence of massive data, scanning the database will cost a lot of time and memory, and the algorithm flow is displayed in Fig 1.

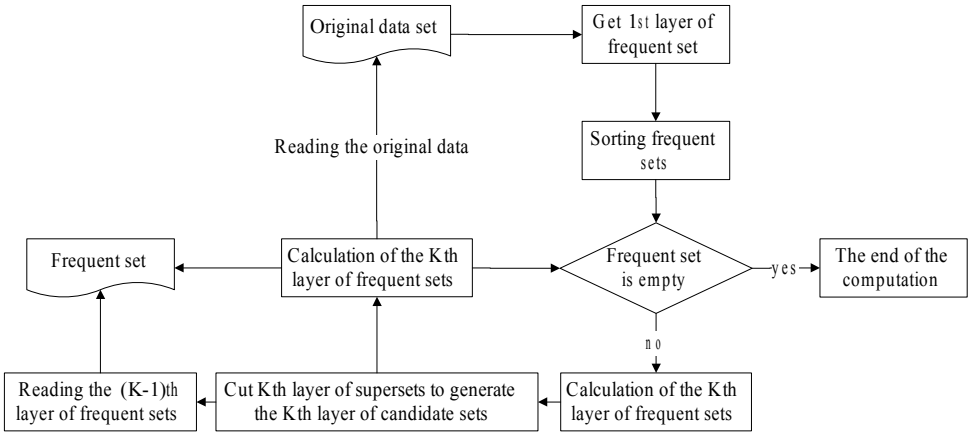

Fig 1. Traditional Apriori algorithm flow

The analysis of Apriori algorithm and the algorithm flow indicates three aspects restricting the speed of Apriori:

- Repeated scanning the transaction set to select frequent K-item sets is time-consuming.

- Generation of (k+1)th superset by frequent Kth itemsets is time-consuming and requires large amount of calculation

- When the k+1 item supersets are clipped into k+1 candidate sets, K-item frequent sets need multiple matching, and this process is time-consuming.

In order to transfer original Apriori algorithm into Map/Reduce implementation, the key data of the original method should be found out, and they should be mapped to Map/Reduce's key and value. Map/Reduce-model-based conversion scheme is proposed according to the above three aspects.

Firstly, the process to compute frequent itemsets with Apriori algorithm is a process of word counting in essence. The transaction set scanning process means to summarize the sets, so, itemset is taken as the value (=1) of this stage in this paper. In the calculation process of Map/Reduce, the framework divides the data sets into several sub blocks and distributes them to all nodes for counting via Map function, and the word statistics is finished in the reduce stage, thus parallel improvement in this stage is achieved and the time of scanning transaction sets is greatly shortened.

The analysis of the second algorithm demonstrates that when kth layer superset is generated from a sorted (k-1)th frequent set, only the itemsets with one different tail item of all frequent sets

will be connected. For example, for frequent itemsets ABC, ABD and BCD, in the production of 4 supersets, only ABC and ABD need connecting to generate ABCD, then ABCD is a 4-item superset.

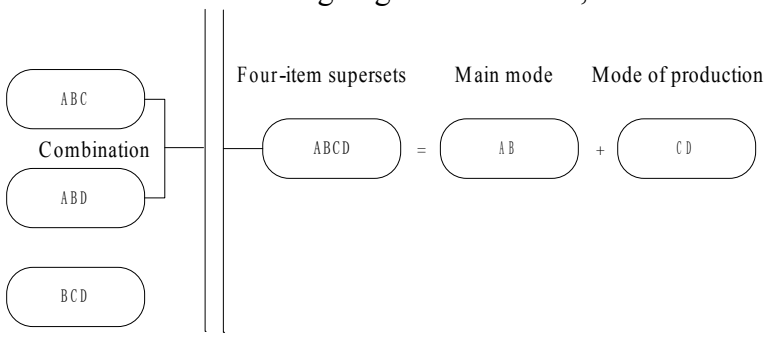

Fig 2. Schematic diagram of superset generation

Fig 2 clearly displays that, in the production of k+1 supersets, frequent set pattern with the same k-1 items is taken as prefix, and then the 2-ietm set in last model of K-item frequent sets is added to the k-1 mode. In this paper, the same k-1 item pattern is regarded as the main pattern, the 2-item set produced by the last pattern is called model group, and its each item is named as generation pattern in which each sub-model is called the pattern base. In the above example, AB is the main mode, C and D are generation model bases, C and D are ordered to generate CD which is the main mode of production, then the main pattern and generation model form an item set of k+1 item superset.

In the process of generating K-item superset, the main pattern plays a key role, so, in this paper, the frequent items with same main pattern is taken as key and sent to the same reduce, and the value is generation model base, then reduce is used as generation pattern, finally K-item frequent sets are grouped according to the main mode for parallel operation, thus the formation of k+1 item superset is accelerated.

As for the third problem, in the process of cutting, each item of k+1 item superset holds K matching, namely, any one item in superset Item pattern is removed, and then we will judge if the pattenk-1 is in item frequent set, if not, then this item is removed from superset.

In the process of generating K-item superset, each k-item superset is composed of main pattern and generation pattern; that is to say, when any item is removed from generation pattern, the main pattern + (generation pattern -1) must be contained in K-item frequent set, so, we only need to check if (main pattern -1) + generation pattern is in K-item frequent set. Therefore, if all the supersets with the same generation pattern are located in a reduce, only the subsets ended with generation pattern in K-item frequent set will be read, followed by cutting, thus the time and space to read K-item frequent set will be shortened greatly, at the same time, due to subset of frequent itemsets read, the time of comparison and cutting is greatly reduced. So in reality, the frequent itemsets with the same production pattern can be classified as a group, while minimal memory can be used to maintain the kth mapping table of generation pattern in Kth layer frequent set, so that the cutting process will be quickly navigated to related frequent subset so as to accelerate itemsets cutting. The improved program flow chart is as follows:

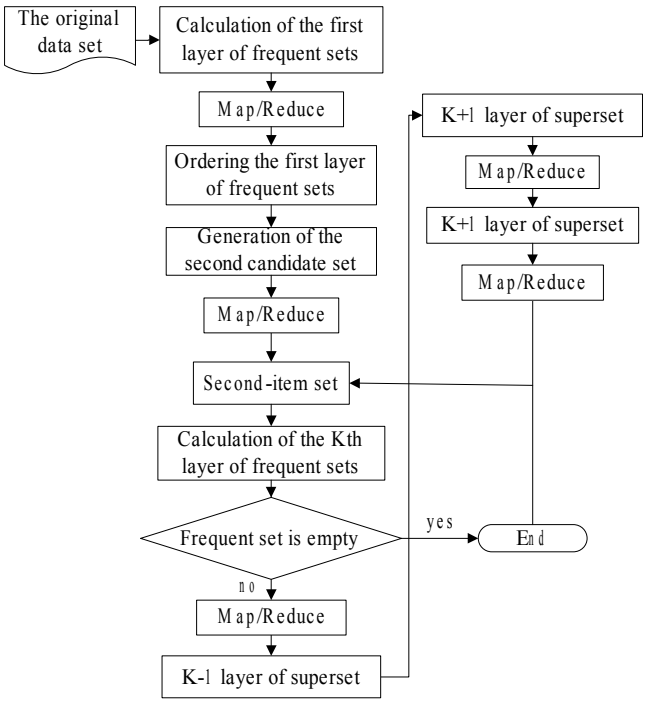

Fig 3. Improved Apriori algorithm flow chart

Experiments and analysis of results

In order to verify extension performance of MR-Apriori algorithm, from respectively the amount of data and computing node quantity, the test time of this algorithm is determined, data mining results are analyzed, and the significance of data mining results is illustrated.

The experimental environment consists of 1 IBM server and 6 Dell desktops, the IBM server is configured with Xeon E5506 processor, 16G memory, dual Gigabit Ethernet and 500G SATA hard disk, and Dell desktop contains i5 760 processor, 4G memory and 500G IDE hard disk. The computers are connected by Huawei Gigabit switches, and all lines are Gigabit network. Operating system for all physical machines is 64-bit ubuntu10.04, the Hadoop version is 0.20.2, virtual platform is Xen, and the virtual machine manager is OpenNeBula2.2.0 version. OpenNeBula [7] directly supports Xen virtual platform, so the simple integration required only installing all components, modifying the oned.conf file and the corresponding parameters in IM_MAD and VM_MAD are changed into Xen.

The experiment uses the cloud test data of MovieLens, the support degree of 10G, 20G and 30G data respectively in 5, 10, 15 and 20 operational nodes is 5%, 10% and 15%. The first group of experiment is performance of 10G data respectively at 5, 10, 15 and 20 operational nodes. 10G data contain about 50000 transactions. The final results are shown in Fig 4.

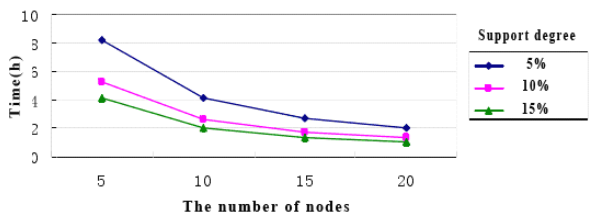

Fig 4. Performance of 10G data

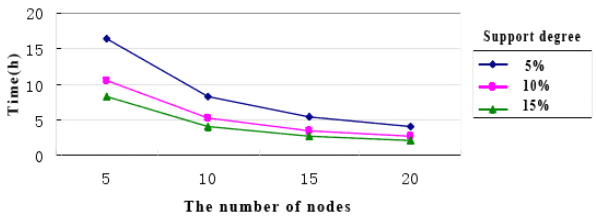

Fig 5. Performance of 20G data

The second group of experiment is performance of 20G data respectively at 5, 10, 15 and 20 operational nodes. 10G data contain about 80000 transactions. The final results are shown in Fig 5.The third group of experiment is performance of 30G data respectively at 5, 10, 15 and 20 operational nodes. 30G data contain about 100000 transactions. The final results are shown in Fig 6.

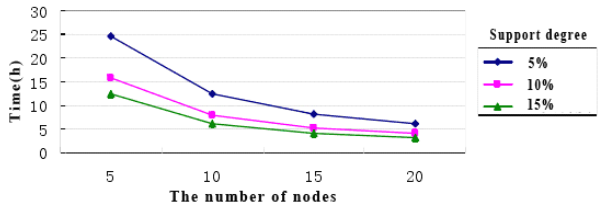

Fig 6. Performance of 30G data

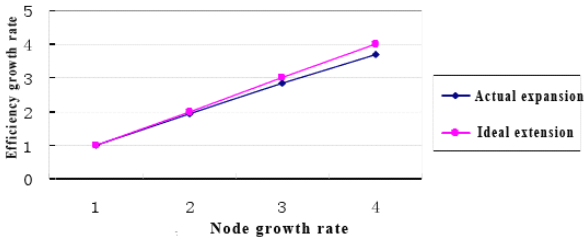

Fig 7. Linear extension of algorithm

The experimental results show that the computing time of MR-Apriori algorithm displays basically linear growth with the increase of data quantity, which indeed indicates the parallel data characteristics of Hadoop framework according to the data block size. In order to expand the capacity of MR-Apriori algorithm, this paper also verifies the extension of algorithms. Fig 7 shows the expansion capability of MR-Apriori algorithm, with 10GB data and 5% support degree. It is shown that with the increase of node quantity, the MR-Apriori algorithm keeps constant high expansion ability, which proves that Hadoop-based MR-Apriori algorithm is suitable for cloud computing and capable of expanding effectively calculation to computing resources, followed by improved overall performance.

Conclusions

In order to realize better generate frequent itemsets in association rule mining, the traditional Apriori algorithm is improved and frequent itemset generation method suitable for Map/Reduce programming model is accordingly designed. With Hadoop-based experiment, the adaptability of new method in Map/Reduce mode and its superiority to traditional Apriori algorithm are verified, and it is demonstrated that the scalability of new method is enhanced with the increase of node quantity.

References

[1] S. Chakrabart. “Data mining for hypertext: a tutorial survey,” SIGKDD Exploration , vol.1, pp. 1-11,Feb 2000.

[2] Jeffrey Dean. “Experiences with Map/Reduce, an abstraction for large-scale computation Proc,” 15th International Conference on Parallel Architectures and Compilation Techniques, , vol.3,pp. 221-225, 2009.

[3] R. Zhou and K. Hwang. “Power Trust: A Robust and Scalable Reputation System for Trusted Peer-to-Peer Computing,” IEEE Transaction. Parallel and Distributed Systems,pp.460-473, Apr 2007.

[4] Zhang Yi. “Design of data mining system for large databases,” Information science, vol.1,pp. 125-128,Sept 2010.

[5] Sun Yan, Tian Zhonghe and Wang Quande. “High-efficiency Dataset-partition association rules mining algorithm,” Computer engineering, vol. 28,pp. 118-120,Dec 2002.

[6] Wang E and Li Ming. ‘Research on mass data mining with cloud computing,” Modern computer, vol.319, pp. 222-225,Jan 2009.

[7] Zhu Zhu. “Research and application of Hadoop-based massive data processing model,” Beijing: Beijing University of Posts and Telecommunications,.vol.25, pp. 40-41, Jan 2008.