Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 183 (2021) 641–647

Procedia Computer Science

www.elsevier.com/locate/procedia

10th International Conference of Information and Communication Technology (ICICT-2020)

Research on parallelization of Apriori algorithm in association rule mining

Huan-Bin WANG*, Yang-Jun GAO

College of Equipment Management And Unmanned Aerial Vehicle Engineering of Airforce Engineering University, ShanXi Xi'an 710051, China

Abstract

Aiming at the performance bottleneck of traditional Apriori algorithm when the data set is slightly large, this paper adopts the idea of parallelization and improves the Apriori algorithm based on MapReduce model. Firstly, the local frequent itemsets on each sub node in the cluster are calculated, then all the local frequent itemsets are merged into the global candidate itemsets, and finally, the frequent itemsets that meet the conditions are filtered according to the minimum support threshold. The advantage of the improved algorithm is that it only needs to scan the transaction database twice and calculate the frequent item set in parallel, which improves the efficiency of the algorithm.

Keywords: Association rules, Apriori algorithm, MapReduce, Parallelization;

1. Introduction

Because the traditional parallel association rule algorithm can not meet the needs of large-scale data sets, and the Apriori algorithm will cause task execution failure due to computer memory overflow when processing large-scale data sets, so it is urgent to improve the Apriori algorithm to better effectively mine the data sets. On the one hand, Apriori algorithm has the condition of parallel improvement, on the other hand, MapReduce model can analyze and process the data set with larger scale[1]. Therefore, based on the previous research, this paper proposes to improve the traditional Apriori algorithm by using the partition idea and MapReduce model.

* Corresponding author. Tel.: 029-84788410; fax: 029-84788410.
*E-mail address:*whbkgd@163.com

10.1016/j.procs.2021.02.109

2. The idea of improving Apriori algorithm

Through the analysis and research of Apriori algorithm, it is found that when the data scale is a little large, the algorithm is inefficient because of the limitation of computer memory and frequent scanning of database. Traditional parallel improvement schemes need a lot of computer resources and communication overhead. Generally speaking, these traditional parallel schemes can not achieve good execution efficiency. But it is better to improve Apriori algorithm by MapReduce model. The specific design strategy is divided into two stages[2]:

(1) Get global frequent itemset

1) The transaction database D is divided into n data sub blocks of approximately equal size in a horizontal way, and then the divided data sub blocks are sent to m nodes. m is far less than n, and the conventional random partition strategy is adopted here[3].

2) On the m nodes, the transaction data blocks allocated to them are calculated respectively. These partitioned databases generate their local candidate k-term sets and save the calculation results on the m nodes.

3) All the local frequent k-term sets generated on m nodes are merged into a global candidate frequent term set. According to the analysis of Apriori algorithm, the local frequent itemset may not be the frequent itemset in database D, but the local frequent itemset must contain all the frequent itemsets in database D.

4) The minimum support threshold for defining item sets in transaction database D is S. The minimum support count of each data block divided is equal to the product of threshold S and the total number of transactions of each data block, which is recorded as Min_conf[4]. In order to calculate the actual support of all candidate itemsets in the global candidate frequent itemsets, D must be scanned for a second time. If the calculated support is not less than the defined threshold S, it is saved to the final frequent itemset P.

(2) Get association rules

1) For the frequent itemset P obtained in the first stage, all the non empty subsets p contained in it are calculated.

2) The minimum support threshold is defined as S, and the minimum confidence threshold is C. the confidence of all non empty subsets P is calculated, and the association rules that meet the requirement of C are filtered out.

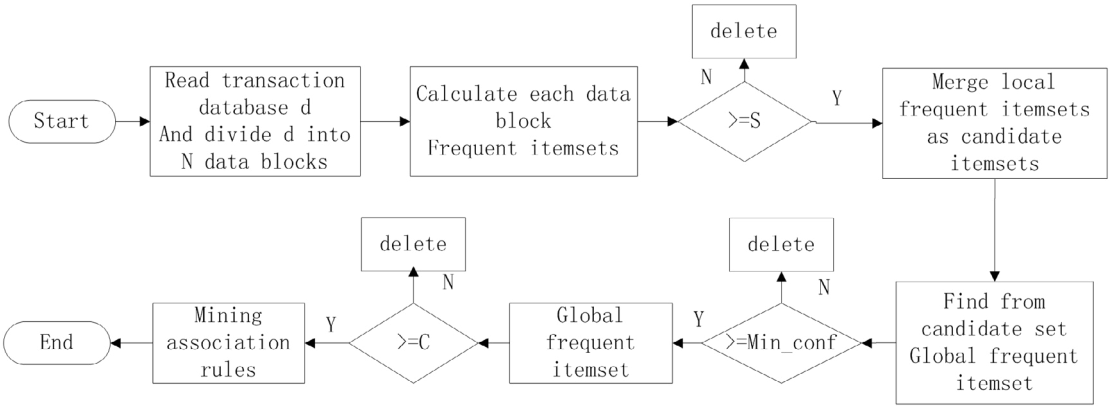

Fig. 1. Flow chart of improved Apriori algorithm.

The flow of the improved Apriori algorithm is shown in Fig. 1. From the above process, we can see that the improved Apriori algorithm has two advantages: Firstly, it only needs to read database D twice to mine all frequent item sets, which reduces the resource overhead of the algorithm in the running steps[5]. Secondly, it can perform the discovery of local frequent itemsets on each node without interference. This design can not only meet the requirements of parallel computing, but also reduce the network communication overhead caused by data transfer between nodes.

3. Improved Apriori algorithm implementation process

In this paper, Apriori algorithm and MapReduce computing model are combined, mainly including two core steps: one is to find out frequent item set, the other is to find association rules. The flow of finding frequent itemsets is shown in Fig. 2:

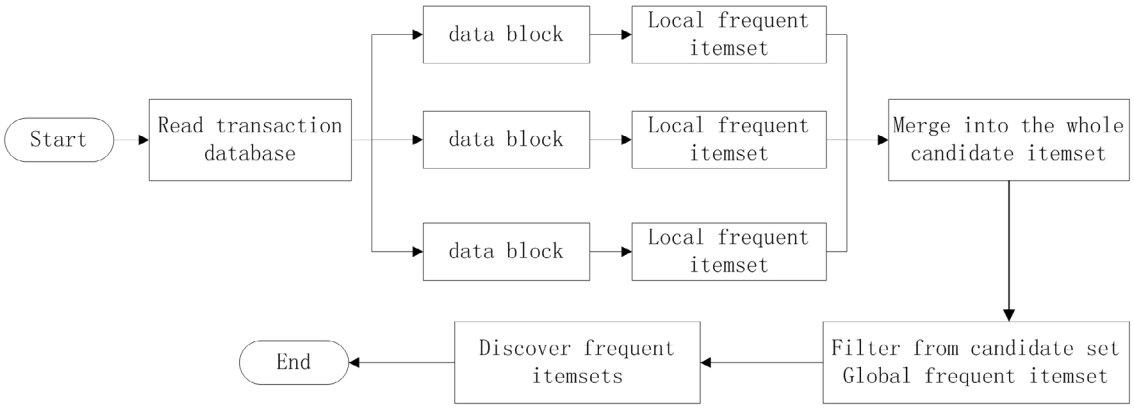

Fig. 2. Flow chart of frequent item set discovery

(1) The execution process of finding frequent item set as follows:

1) The transaction database is read through MapReduce programming model, and then it is divided into n sets of splits with small difference in a horizontal way[6]. In the cluster master-slave architecture mode, the cluster will assign m map tasks and x reduce tasks to each node to process these data.

2) Processing each data block: Because the data form of transaction database includes transaction identifier (ID) and item set, the resolution result is (ID, item) key value pair[7].

3) Execute map task: on the Data Node, execute the assigned map task to parse the data block into (ID, item) key value pairs. First, we calculate the frequent 1-term set (this process is completed by simple word frequency counting), then we calculate the frequent k-term set on each data block, and finally calculate the local frequent k-term set.

4) Write combiner function: from map function to reduce function, a lot of intermediate data of type (list, 1) will be generated. It is necessary to define a combiner function on the map side. The combiner function merges these (list, 1) key values to the data on the local disk, resulting in (list, sum).

5) Execute reduce task: according to the task information assigned by the master node, the key value pair data after merging from the map side is read. The local frequent itemsets generated by each computing node are combined by reduce function. After the combination, they are the global candidate frequent itemsets.

6) Read transaction database: map function is written to scan all item sets from transaction database, and check whether the item set i (i represents an item set) is in the global candidate item set generated in step 5). If itemset i can be found in the global candidate itemset, then (i, 1) will be sent to reduce end for statistics.

7) Merge the output results of the reduce task: the results of the reduce end is merged. Compared with the initial minimum support threshold s given by the program entry, if the support of a candidate frequent itemset is greater than or equal to s, the candidate frequent itemset is stored in the frequent itemset file[8].

(2) The implementation process of discovering association rules is shown in Fig. 3:

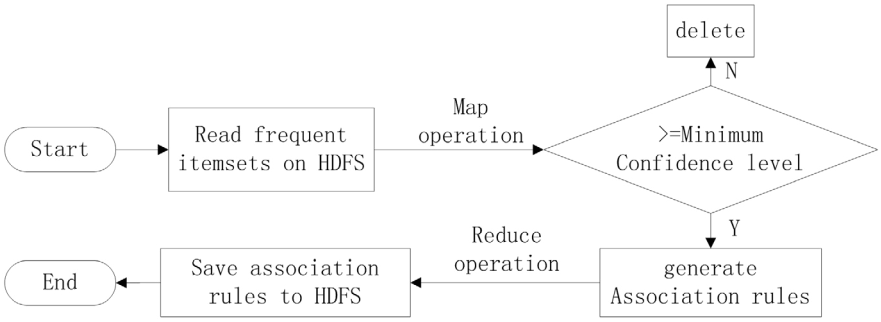

Fig. 3. Flow chart of discovering association rules

1) The data is stored on HDFS, and the file data on HDFS is read through MapReduce calculation model. And the data in the file is parsed into (key1, item1) key value pairs[9]. Where key1 represents the offset of each row, and item1 represents the row data (that is the frequent item set in the row).

2) Using the custom map method, read the key value pair (key1, Item1) in the first step, perform logical calculation, and get the key value pair (key2, Item2). Among them, key2 represents frequent items, and Item2 represents association rules corresponding to frequent items.

3) Using the custom reduce method, the output of map method in step 2) is taken as the input parameter. the frequent item set is deleted which is less than the minimum confidence, the logical specification processing is performed on the qualified results, (Key3, Item3) key value pairs is output, and finally the calculation results are stored on HDFS[10]. Where Key3 represents association rules and Item3 represents confidence. The improved Apriori algorithm is finished.

4. Example analysis of improved Apriori algorithm

In order to verify the execution process of the improved Apriori algorithm, the transaction database D is analyzed as an example. Here, the threshold is set to 2, that is, min_sup=2/10=0.2 [11]. The details of transaction database D are shown in Table 1:

Table 1. Transaction database D

TID	Items
T01	X1,X2,X5
T02	X2,X4
T03	X1,X2,X3,X5
T04	X1,X3
T05	X2,X3
T06	X1,X2,X3
T07	X2,X5
T08	X1,X3
T09	X3,X4,X5
T10	X1,X2,X5

The original transaction database d is divided into three sub transaction databases D1, D2 and D3. The first group D1 includes T01, T02 and T03, the second group D2 includes T04, T05 and T06, and the third group D3 includes T07, T08, T09 and T10. According to the calculation method of local minimum support threshold, the minimum support thresholds of D1, D2 and D3 are 0.6, 0.6 and 0.8. The steps are as follows:

(1)Find frequent itemsets.

1) The primary Node1 processes the original transaction database, it is divided into three sub transaction databases, which are recorded as D1, D2, D3. The sub transaction databases are submitted to three slave nodes Node2, Node3 and Node4 for analysis and calculation[12]. They are parsed into (key, value) pairs,the results are as follows:

Table 2. Analysis results

Node	Key value pair
Node2	{T01,(X1,X2,X5)},{T02,(X2,X4)},{T03,(X1,X2,X3,X5)}
Node3	{T04,(X1,X3)},{T05,(X2,X3)},{T06,(X1,X2,X3)}
Node4	{T07,(X2,X5)},{T08,(X1,X3)},{T09,(X3,X4,X5)},{T10,(X1,X2,X5)}

2) Each node uses the map function to process the input sub transaction database, the key value pairs are split and one candidate set of each transaction database is obtained. The output result is (key, 1) key value pair. The split results are as follows.

Table 3. Candidate 1 item set

Node	(key,value) pair	(key,1) pair
Node2	{T01,(X1,X2,X5)}	(X1,1),(X2,1),(X5,1)
	{T02,(X2,X4)}	(X2,1),(X4,1)
	{T03,(X1,X2,X3,X5)}	(X1,1),(X2,1),(X3,1),(X5,1)
Node3	{T04,(X1,X3)}	(X1,1),(X3,1)
	{T05,(X2,X3)}	(X2,1),(X3,1)
	{T06,(X1,X2,X3)}	(X1,1),(X2,1),(X3,1)
Node4	{T07,(X2,X5)}	(X2,1),(X5,1)
	{T08,(X1,X3)}	(X1,1),(X3,1)
	{T09,(X3,X4,X5)}	(X3,1),(X4,1),(X5,1)
	{T10,(X1,X2,X5)}	(X1,1),(X2,1),(X5,1)

In order to improve the cluster operation efficiency, the custom combiner function is used to merge the output result of map function on each DataNode. The result is one local candidate set. The results are as follows.

Table 4. Local candidate 1 item set

Node	Combiner function	(key,value) pair
Node2	{X1,list(1,1)},{X2,list(1,1,1)}	(X1,2),(X2,3)
	{X3,list(1)},{X4,list(1)}	(X3,1),(X4,1)
	{X5,list(1,1)}	(X5,2)
Node3	{X1,list(1,1)},{X2,list(1,1)}	(X1,2),(X2,2)
	{X3,list(1,1,1)}	(X3,3)
Node4	{X1,list(1,1)},{X2,list(1,1)}	(X1,2),(X2,2)
	{X3,list(1,1)},{X4,list(1)}	(X3,2),(X4,1)
	{X5,list(1,1,1)}	(X5,3)

3) After the above calculation, the local one frequent itemset of three calculation nodes can be obtained, and the calculation results are arranged as follows:

Table 5. Local frequent 1 item set

Node	Frequent 1 item set
Node2	{X1,X2,X3,X4,X5}
Node3	{X1,X2,X3}
Node4	{X1,X2,X3,X4,X5}

The Apriori algorithm is used to generate two local candidate sets and three local candidate sets. Here, only three local candidate sets are given, and the results are as follows.

Table 6. Local candidate 3 item set

Node	Local candidate 3 item set
Node2	{(X1,X2,X3),1},{(X1,X2,X5),2},{(X2,X3,X5),1}
Node3	{(X1,X2,X3),1}
Node4	{(X1,X2,X5),1},{(X3,X4,X5),1}

4) The Reduce function is used to reduce the support degree of the above three sets of local frequent items. The results are as follows:

Table 7. Protocol results

Reduce operation	(key,value) pair
{(X1,X2,X3),list(1,1)}	{(X1,X2,X3),2}
{(X1,X2,X5),list(2,1)}	{(X1,X2,X5),3}
{(X2,X3,X5),list(1)}	{(X2,X3,X5),1}
{(X3,X4,X5),list(1)}	{(X3,X4,X5),1}

After the above steps, the global candidate sets are obtained as follows: {(X1, X2, X3), (X1, X2, X5), (X2, X3, X5), (X3, X4, X5)}.

5) Read the original transaction database D again, compare the global candidate 3 item set calculated in the previous step with the minimum support threshold Min_sup, and finally filter out the frequent 3 item set that is not less than the minimum support threshold and L3 = {(X1, X2, X3), (X1, X2, X5)}.

(2)Find association rules.

1) Define the minimum confidence Min_conf = 0.65. Upload the frequent 3 itemsets generated by the frequent itemsets found in the first step to the Node1 and Node2. After the analysis of the two nodes, the results are recorded as {T01, (X1, X2, X3)}, {T02, (X1, X2, X5)}.

2) The map function is used to analyze the key value pairs in the previous step, and the output result is a key value pair of the form (frequent item set, rule). The calculation results are shown in Table 8:

Table 8. Analysis results

Node	Map operation	(key,value) pair
Node1	{T01,(X1,X2,X3)}	{(X2,X3)=>X1,0.67}
		{(X1,X2)=>X5,0.75}
Node2	{T02,(X1,X2,X5)}	{(X1,X5)=>X2,1}
		{X5=>(X1,X2),0.6}

The above results are sent to the reduce function and stored on the distributed file system. The result of the reduce function is: {(X2, X3) X5, 0.67}, {(X1, X2) X5, 0.75}, {(X1, X5) X2, 1}, {X5 (X1, X2), 0.6}. The implementation of the improved Apriori algorithm is completed. The example analysis shows that it is feasible.

5. Concluding remarks

The Apriori algorithm is improved by using the idea of MapReduce model, the idea and process of the improved Apriori algorithm are given. The execution process of the improved Apriori algorithm is verified by a specific example. The analysis of an example shows that this method is feasible.

References

1. Prajapati D J, Garg S, Chauhan N C. Interesting association rule mining with consistent and inconsistent rule detection from big sales data in distributed environment[J]. Future Computing & Informatics Journal, 2017, 2(1):19-30
2. Huh J H, Kim H B, Kim J. A Method of Modeling of Basic Big Data Analysis for Korean Medical Tourism:A Machine Learning Approach Using Apriori Algorithm[C]//Information Science and Applications 2017, Singapore:Springer, 2017:784-790
3. She X, Zhang L. Apriori Parallel Improved Algorithm Based on MapReduce Distributed Architecture[C]//Sixth International Conference on Instrumentation & Measurement, Computer, Communication and Control, Harbin, China:IEEE, 2016:517-521
4. Li J S, Qin S J. An Improved Apriori Algorithm for Mining of Association Rules[C]// International Conference on Computer Science and Artificial Intelligence, Jakarta, Indonesia: ACM Press, 2017:244-249
5. Wang W, Li Q.Algorithm for Map/Reduce-based association rules data mining[C]//Proceedings of 2013 International Conference on Information Science and Computer Applications, Hong Kong, China:CRC Press, 2013:334-339
6. Atta-ur-Rahman, Das S. Data Mining for Student's Trends Analysis Using Apriori Algorithm[J]. International Journal of Control Theory & Applications, 2017, 10(18):107-115
7. Pandagale A A, Surve A R.Hadoop-HBase for finding association rules using Apriori MapReduce algorithm[C]//IEEE International Conference on RecentTrends in Electronics, Information & Communication Technology, Bangalore,India:IEEE, 2017:795-798
8. Govada A, Patluri A, Honnalgere A. Association Rule Mining using Apriori for Large and Growing Datasets under Hadoop[C]// Vi International Conference, NewYork, USA:ACM Press, 2017:14-17
9. Liao J, Zhao Y, Long S.MRPrePost-A parallel algorithm adapted for miningbig data[C]//IEEE Workshop on Electronics, Computer & Applications, Ottawa, Canada:IEEE, 2014: 564-568
10. Rustogi S, Sharma M, Morwal S. Improved Parallel Apriori Algorithm for Multi-cores[J]. International Journal of Information Technology & ComputerScience, 2017, 9(4): 18-23
11. Lin C W, Li T, Fournier-Viger P,et al. Efficient Mining of Multiple Fuzzy Frequent Itemsets[J]. International Journal of Fuzzy Systems, 2016, 19(4):1-9.
12. Yun U, Kim D. Mining of high average-utility itemsets using novel list structure and pruning strategy[J]. Future Generation Computer Systems, 2017, 68:346-360.