Association Rule Mining with Parallel Frequent Pattern Growth Algorithm on Hadoop

Zhigang Wang [1,2], Guiqiong Luo[3,*] ,Yong Hu[1,2], ZhenZhen Wang[1]

[1]*School of Software Engineering*

Jinling Institute of Technology

Nanjing, Jiangsu, China

[2]Jiangsu Key Laboratory of Creativity and Application of Digital Media Art

Jinling Institute of Technology

Nanjing, Jiangsu, China

[3]*Department of Information Engineering*

Hunan Radio&TV University

Changsha 410007,China

*corresponding author is Guiqiong Luo

***Abstract* — Although the association rules mining algorithm FP-Growth is more efficient than Apriori, it has two disadvantages. The first one is that the FP-tree could be too large to be created in memory, the other one is its serial processing. A novel improved version Parallel Frequent Pattern Growth (P-FP-Growth) is proposed in this paper. It does not need to create global FP-tree, and implements parallel processing in every step during association rule mining. We deploy it on LAN computer cluster and on Hadoop, and obtain the result of frequent pattern mining and compare the execution time. Our experiments show that P-FP-Growth is: i) much faster than FP-Growth if we run it with multiple computing nodes, ii) it can be deployed on cloud computing platform such as Hadoop, iii) it is faster to deploy it on Hadoop than on LAN computer cluster with increasing computing nodes.**

Keywords- Data Mining, Association Rule, P-FP-Growth Algorithm, Cloud Computing, Map Reduce Programming, Hadoop Platform

I. INTRODUCTION

Cloud computing is a Internet application mode which can realize accessing to a Shared resource pool anytime and anywhere on demand[1]. The resources from the shared resource pool can be computing facilities, storage devices, application programs, and so on. Typically, computer cluster is adopted to form data and computing center, and provided to the users in the form of services[2]. MapReduce[3] is a cloud computing model proposed by Google. Hadoop[4] is an Apache open source project, and it is a distributed software architecture which can process a large amount of data. Hadoop is open source implementation of Google cloud computing system, and it is composed of HDFS, MapReduce and HBase. Hadoop implements the infrastructure of cloud computing software platform, which including the distributed file system, MapReduce framework, and integrates a series of platform such as database, management of cloud computing, data warehouse and so on.

Data mining is a type of information analysis task which has high requirement of storage capacity and computing capability. Data mining of massive application data can be implemented with parallel algorithm, relying on cloud computing which has high performance and high cost performance. Association rule is an important topic of data mining which is first proposed by Agrawal in the literature[5] in 1993. Apriori algorithm is a famous association rule mining algorithm proposed by R.Agrawal and R.Srikant in 1994[6]. Apriori algorithm must spend a lot of time to deal with huge candidate item sets and repeat scanning database. Aiming at the shortcomings of the Apriori algorithm, Han put forward an association rule mining algorithm FP - Growth[7]. FP - Growth algorithm is approximately one order of magnitude faster than the Apriori algorithm, but it has two disadvantage: the first is that its frequent pattern tree may be too big to be created in the memory; the second is its serial processing approach. The process of classic FP - Growth algorithm is shown in figure 1, and the whole process is serial. In addition if the data sets are very large, the FP - tree may be too big, and the memory can't accommodate.

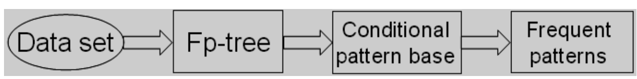


Figure 1. The procedure of classic FP-Growth algorithm

To realize association rule mining on Hadoop, we need to improve the traditional association rule mining algorithm. Literature[8] presents a parallel algorithm for mining frequent item sets based on Apriori. Literature[9] improves CD (Count Distribution) algorithms which is based on Apriori, and realizes parallel association rule mining using MapReduce on Hadoop. Literature[10] and [11] also realize parallel association rule mining using MapReduce on Hadoop with Apriori class algorithms.Literature[12] proposes data partitioning (data division), literature [13-14] use multiple thread share memory (multithreaded memory - sharing).Literature[15] divides global FP - tree into sub tree for parallel processing, literature[16] regards the mining of each condition pattern library as a sub task, and assigns these sub tasks to computing node in computer cluster.Literature[17] uses multiple local FP - tree instead of global FP - tree, and puts forward a distributed parallel association rules mining algorithm which named P - FP - Growth on the basis of classic FP - Growth algorithm.

FP - Growth class algorithms are approximately one order of magnitude faster than the Apriori class algorithms. In FP - Growth class parallel algorithms, some only realize the parallel process on transaction item counting, filtering or sorting, some only realize the parallel process on the condition pattern library mining. Due to use multiple local FP - tree instead of global FP - tree, P - FP - Growth Algorithm not only avoids that the global FP - Tree is too big to be stored in memory, but also implements the parallel processing of building FP - Tree.

P - FP - Growth Algorithm breaks through the capacity bottleneck of FP - Growth algorithm, and implemented parallel processing in every step. If it is deployed on cloud computing platform such as Hadoop, the association rule mining task of mass data will be solved, and the implementation can be provided as a service to users so that they can achieve association rule mining but do not need a lot of computers or any special data mining software.

II. P -FP -GROWTH ALGORITHM

Parallel association rules mining algorithm P -FP -Growth uses multiple local FP - tree instead of global FP - tree, so as to avoids that the global FP - Tree is too big to be stored in memory. And in this algorithm the whole data mining processing is paralleled, including transaction item counting, filtering, sorting, local FP - Tree building, and condition pattern library mining.

The process of P -FP -Growth algorithm is shown in figure 2.

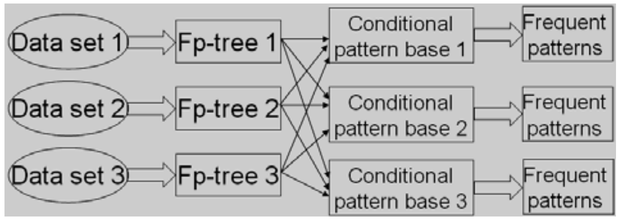


Figure 2. The process of P -FP -Growth algorithm

Set $I = \{i_1, i_2, ..., i_n\}$, I is the collection of all item.

Transaction T is a item set and the item is from I , $T \subseteq I$,Every transaction has a unique ID. D is a database which is made up of more than one transaction, and $D = \sum_{j=1}^{m} d_j$, d_j ($j = 1,2,...m$) are distributed in different storage nodes M_j . C_j （j=1,2…p） are a group of computing nodes with powerful ability of calculation. In physics, a storage nodes and a computing node can be completely coincidence, parts overlap or completely different. Now we need to make full use of the computing nodes C_j (j = 1, 2... p) to find out the frequent patterns as soon as possible from the global transaction database D in order to get association rules.

A. The step of P -FP -Growth algorithm

(1) In each of the storage nodes M_j :get the count of every items in d_j .

(2) In the center computing node: get together and sum the count of every items in all d_j , so as to get the count of every item in the global database D. Order the items according to its count descending. Delete items which's count is less than the given minimum support count.
The result is global 1-item frequent set L.

(3) In the center computing node: In order to distribute the frequent pattern mining task to many computing nodes by items to realize parallel processing in the follow step, the center computing node needs to assign computing node for every item in L, and adds the assign results to L. Finally we obtain L such as table 1.

TABLE 1. GLOBAL 1-ITEM FREQUENCY SET AND COMPUTING NODE ASSIGN RESULT

item	count	computing node
…	…	…
i_2	10	C_3
i_5	8	C_4
i_9	5	C_5
…	…	…

The center computing node distributes L to each storage nodes. There are a lot of strategies to assign computing node for every item. The simplest way is that assign computing node for items by their count order. For example:

Items are named 1,...,n according to their count in descending order. The number of computing node is j, which in turn called C1,…,Cj. Assign computing node Ci+1 for item n when n mod j = i.

(4) Each storage node M_j: processes every transactions in d_j according to global 1-item frequent set L. The main points of the procession as the following:

① Filter out the items without in L.

② Order the rest of the items by their count in L in descending.

③ Add transactions to local FP-tree.

After this step, we can get several local FP-trees and local Header Tables that are bound by global 1-item frequency set L.

(5) Each storage node M_j :according to the additional computing node assign information in L, sends items in local Header Tables with their condition mode library (or said that condition model base) in local FP-trees to corresponding computing nodes.

(6) Each computing node C_j :aggregates condition mode library group by item and do frequent pattern mining. If the condition mode library of an item is very large, can recursively invoke this algorithm to disintegrate the frequent pattern mining task of this item so as to take parallel processing.

(7) Each computing node C_j :sends the frequent patterns to the center computing node, which summarizes and gets the global association rules set.

B.Deploy P -FP -Growth Algorithm in MapReduce compute mode

P -FP -Growth algorithm can been deployed with MapReduce compute mode as the follow:

// item counting

Mapper (j, d_j) { Count(d_j , i_i) }

Reduce (i_i , Count)

// item descending sorting and delete those whose count are less than the minimum support count

Map (j, d_j) { Sort_del(d_j , T)}

Reduce ( ) //copy the intermediate data to the output directly

//build local FP-tree_j for every d_j

Map (j, d_j) { Build_T (d_j , FP-tree_j)}

Reduce ( ) //copy the intermediate data to the output directly

// distribute and gather condition pattern library by item

Map (FP-tree_j , i_i) {Send (i_i , Conditional_pattern_base)}

Reduce (i_i , Conditional_patter_base)

// frequent patterns mining by item and summarize mining results

Map (i_i ,Conditional_patter_base) { Mining (i_i , Conditional_patter_base)}

Reduce (Result，Frequent Pattern).

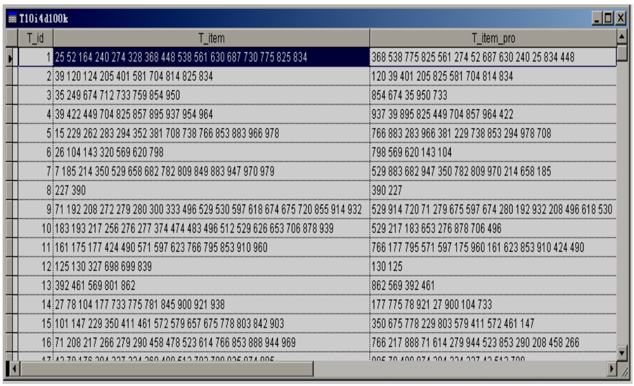
III. EXPERIMENT

Used T10I4D100K.dat (http://fimi.ua.ac.be/data/T10I4D100K.dat) as the experimental data which had 100000 transactions. The minimum serial number of the items is 0, and the maximum serial number is 999. The amount of the items which actually appear in all transactions is 871. It means that there are 29 items does not appear at all in all transactions. The total times of all items appear in all transactions is 1010221, and the average length of all transactions is 10.1.

Select multiple computers in a laboratory as computer cluster. Set the Minimum Support is 1% which means that the Minimum Support count is 1000.

A.The frequent patterns mining result of data set T10I4D100K

The transaction data is shown as figure 3. Every line is a transaction. T_id is the ID of a transaction. T_item is the original data from T10I4D100K.dat line by line. T_item_pro is the intermediate results after sorting and deleting those whose count are less than the minimum support count.


T10i4d100k

T_id	T_item	T_item_pro
1	25 52 164 240 274 328 368 448 538 561 630 687 730 775 825 834	368 538 775 825 561 274 52 687 630 240 25 834 448
2	39 120 124 205 401 581 704 814 825 834	120 39 401 205 825 581 704 814 834
3	35 249 674 712 733 759 854 950	854 674 35 950 733
4	39 422 449 704 825 857 895 937 954 964	937 39 895 825 449 704 857 964 422
5	15 229 262 283 294 352 381 708 738 766 853 883 966 978	766 883 283 966 381 229 738 853 294 978 708
6	26 104 143 320 569 620 798	798 569 620 143 104
7	7 185 214 350 529 658 682 782 809 849 883 947 970 979	529 883 682 947 350 782 809 970 214 658 185
8	227 390	390 227
9	71 192 208 272 279 280 300 333 496 529 530 597 618 674 675 720 855 914 932	529 914 720 71 279 675 597 674 280 192 932 208 496 618 530
10	183 193 217 256 276 277 374 474 483 496 512 529 626 653 706 878 939	529 217 183 653 276 878 706 496
11	161 175 177 424 490 571 597 623 766 795 853 910 960	766 177 795 571 597 175 960 161 623 853 910 424 490
12	125 130 327 698 699 839	130 125
13	392 461 569 801 862	862 569 392 461
14	27 78 104 177 733 775 781 845 900 921 938	177 775 78 921 27 900 104 733
15	101 147 229 350 411 461 572 579 657 675 778 803 842 903	350 675 778 229 803 579 411 572 461 147
16	71 208 217 266 279 290 458 478 523 614 766 853 888 944 969	766 217 888 71 614 279 944 523 853 290 208 458 266

Figure3. The intermediate results after items sorting and deleting

Either using FP -Growth or using P -FP -Growth Algorithm, We get the same frequent patterns mining results as shown in table 2.

TABLE 2 THE FREQUENT PATTERNS MINING RESULT OF DATA SET T10I4D100K

Frequent pattern	Count of support	Support degree
217 346	1336	1.3360%
368 829	1194	1.1940%
829 789	1194	1.1940%
368 682	1193	1.1930%
39 825	1187	1.1870%
39 704	1107	1.1070%
825 704	1102	1.1020%
390 227	1049	1.0490%
722 390	1042	1.0420%
39 825 704	1035	1.0350%

B. The comparison of FP -Growth and P -FP -Growth

Classic FP -Growth is a algorithm of serial processing, and it needs a global FP - Tree. We run classic FP -Growth on a single computer. P -FP -Growth is a algorithm of parallel processing, and implements parallel processing in every step. We run P -FP -Growth on a local area network (LAN) multiple nodes computer cluster which has ten computer. The executing time comparison of FP -Growth and P -FP -Growth is shown as table 3.

TABLE 3 EXECUTING TIME COMPARISON OF FP -GROWTH AND P -FP -GROWTH（UNIT OF TIME:SECOND）

	FP-Growth	P-FP-Growth (10 computing nodes)
Items count of all transaction	5	0.6
Items filter and sort	112	12
Build FP -Tree	289	30
Frequent patterns mining	729	78
Auxiliary work	0	3
Total time	1135	123.6

C. The comparison of P -FP -Growth deployed on LAN computer cluster and on Hadoop

Install Hadoop in the Windows system, and configure them as fully distributed mode. Program for P - FP - Growth algorithm accordance with the MapReduce framework. Execute the Program of P - FP - Growth algorithm on Hadoop platform for frequent pattern mining.

comparing P - FP - Growth algorithm’s executing time of the local area network (LAN) multiple nodes parallel processing and of the cloud computing on Hadoop platform, the result are shown in figure 4. In figure 3 the y is coordinate for the algorithm execution time, and x is abscissa for the number of machine node which participate in calculating.

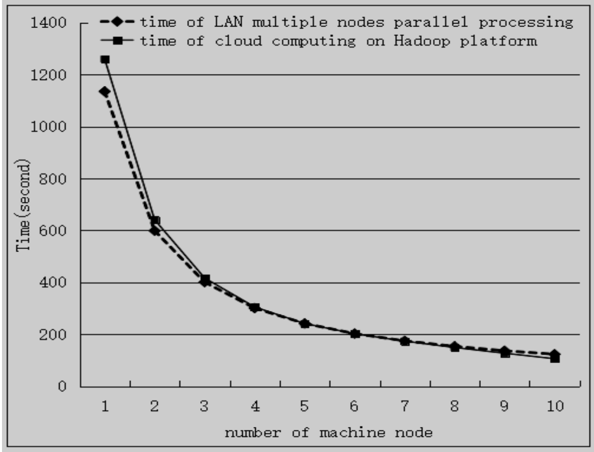


Figure 4. The comparison of executing time.

VI. CONCLUSION

P - FP - Growth uses multiple local FP - tree instead of global FP - tree, so as to avoid that the global FP - Tree is too big to be stored in memory, and implements parallel processing in every step, including transaction item counting, filtering, sorting, local FP - Tree building, and condition pattern library mining. P - FP - Growth is much more faster than FP - Growth as Table 3 shows if we run it with multiple computing nodes.

P - FP - Growth Algorithm breaks through the capacity bottleneck of FP - Growth algorithm. If it is deployed on cloud computing platform such as Hadoop, the association rule mining task of mass data will be solved, and the implementation can be provided as a service to users so that they can achieve association rule mining but do not need a lot of computers or any special data mining software.

Experiment shows that P - FP - Growth algorithm can been deployed on cloud computing platform Hadoop, and figure 3 tell us that it is faster to deploy P - FP - Growth on Hadoop than on LAN multiple nodes parallel processing with the count of computing nodes increasing.

ACKNOWLEDGMENTS

This work was supported by Top-notch Academic Programs Project of Jiangsu Higher Education Institutions,and Jiangsu province 2016 college students' innovative entrepreneurial training program(201613573047X).

REFERENCES

[1] Mell P,Grance T.The NIST definition of cloud computing.National Institute of Standard and Technology,U S Department of Commerce, 2010.

[2] Frank E. Gillett. Future View: The New Technology Ecosystems of Cloud, Cloud Services and Cloud Computing. Forrester Report, August 2008, 6(1):59-62.

[3] Dean J,Ghemawat S.MapReduce: simplified data processing on large clusters.Proceedings of the 6th Symposium on Operating System Design and Implemention.San Francisco, California,USA,2008,103-115.
[4] Agrawal R,Imielinski T,Swami A.Mining association rules between sets of items in large databases.Proceedings of ACM SIGMOD International Conference on Management of Date,1993:207-216.
[5] Agrawal R,Srikant R.Fast algorithms for mining association rules.Proceedings of the 1994 International Conference on Very Large Data Bases,1994,487-499.
[6] Han J,Pei J,Yin Y.Mining Frequent Patterns Without Candidate Generation.Proceedings of ACM SIGMOD International Conference on Management of Data,2000:1-12.
[7] Ye Yan-bin, Chiang C-C. A Parallel Apriori Algorithm for Frequent Item sets Mining.Proceedings of the Fourth International Conference on Software Engineering Research Management and Applications. 2006,87-94.
[8] YU Chuli，XIAO Ying-yuan，YIN Bo．A parallel algorithm for mining frequent item sets on Hadoop.Journal of Tianjin University of Technology,2011,27(1):25-28.
[9] MA Jie.Research on Association Rule Data Mining Algorithm under Cloud Computing Environment.Journal of Chongqing Technology Business University,2012,29(11):36-39.
[10] RONG Xiang, LI Ling -juan.A method for frequent set mining based on MapReduce.Journal of Xi'an University of Posts and Telecommunications, 2011.16(4):37-39.
[11] Pramudiono I,Kitsuregawa,M.Parallel FP -Growth on PC cluster.Proceedings of International Conference on Internet Computing,2003:467-473.
[12] Zaïane O R,Mohammad E H,Lu P.Fast parallel association rule mining without candidacy generation.Proceedings of 1st IEEE International Conference on Data Mining,2001,665–668.
[13] Liu L,Li E,Zhang Y,Optimization of frequent item-set mining on multiple-core processors.Proceedings of 33rd International Conference on Very Large Data Bases,2007,1275–1285.
[14] Tan, K.-L., Sun, Z.-H. An algorithm for mining FP-trees in parallel. Computer Engineering and Applications (13), 2006: 155-157.
[15] Chen, M., Li, H.-F. FP - Growth parallel algorithm in cluster system. Computer Engineering (20), 2009:71-75.
[16] Wang Zhi-gang, Wang Chi-she. A Parallel Association-Rule Mining Algorithm.Proceedings Of 2012 Web Information Systems and Mining. 2012:125-129.