Association Rule Mining with Parallel Frequent Pattern Growth Algorithm on Hadoop

Zhigang Wang 1,2, Guiqiong Luo3,*, Yong Hu1,2, ZhenZhen Wang1
1School of Software Engineering
Jinling Institute of Technology
Nanjing, Jiangsu, China
2Jiangsu Key Laboratory of Creativity and Application of Digital Media Art
Jinling Institute of Technology
Nanjing, Jiangsu, China
3Department 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

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 avoid that the global FP - Tree is too big to be stored in memory. 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.

Set \( I = \{i_1, i_2, \ldots, i_n\} \), \( I \) 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, \quad d_j (j = 1, 2, \ldots, m) \text{ are distributed in different storage nodes } M_j. \quad C_j (j = 1, 2, \ldots, p) \text{ 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, \ldots, p) \text{ to find out the frequent patterns as soon as possible from the global transaction database } D \text{ 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>
<thead>
<tr>
<th>item</th>
<th>count</th>
<th>computing node</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
<td>C3</td>
</tr>
<tr>
<td>5</td>
<td>8</td>
<td>C4</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>C5</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td>...</td>
</tr>
</tbody>
</table>
The center computing node distributes L to each storage node. 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 Mj: processes every transactions in d_j according to global 1-item frequent set L. The main points of the procession as the following:
- Filter out the items without in L.
- Order the rest of the items by their count in L in descending.
- 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 Mj: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 Cj: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 Cj: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_j) }
Reduce (i_j, 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 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_j) {Send (i_j, Conditional_pattern_base) }
Reduce (i_j, Conditional_pattern_base)

// frequent patterns mining by item and summarize mining results
Map (i_j,Conditional_pattern_base) { Mining (i_j, Conditional_pattern_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 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 871. It means that there are 29 items does not appear at all in all transactions.

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.

Figure 3. 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

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

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)

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

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.

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


