COMPARISON AND IMPROVEMENT OF ASSOCIATION RULE MINING ALGORITHM

XIAO-FENG GU [1], XIAO-JUAN HOU [1], CHEN-XI MA[4], AO-GUANG WANG [1], HUI-BEN ZHANG [2], XIAO-HUA WU [1], XIAO-MING WANG [3]

[1] School of Information and Software Engineering, University of Electronic Science and Technology of China, ChengDu, 610054, China
[2] State Grid Inner Mongolia Eastern Electric Power Limited Company Information and Communication Branch, Huhehaote, 010020, China
[3] Liaohe Oil Field, Panjin, 124000, China
[4]Chengdu COMSYS Information Tech. Co. Ltd, ChengDu, 610054, China
EMAIL: 1033203783@qq.com, guxf@uestc.edu.cn

Abstract:

In recent years, the data mining technology has been developed rapidly. New efficient algorithms are emerging. Association data mining plays an important role in data mining, and the frequent item sets are the highest and the most costly. This paper is based on the association rules data mining technology. The advantages and disadvantages of Apriori algorithm and FP-growth algorithm are deeply analyzed in the association rules, and a new algorithm is proposed, finally, the performance of the algorithm is compared with the experimental results. It provides a reference for the extension and improvement of the algorithm of association rule mining.

Keywords:

Association rules; Apriori; FP-growth

1. Introduction

The concept of data mining is was firstly raised at the ACM conference in the United States in 1995. Data mining is also called KDD, It refers to mining a large amount of data from the database to reveal the implicit, unknown, potentially useful information of the non-trivial process [1] [2]. It has a high practical value in the field of database research.

Association rule mining is an important branch of data mining research [3]. Association Rule Mining is used to find a link that is hidden in the interest of large data sets. In this paper, the two algorithms are discussed.

2. Association rule theory

2.1 Basic definition

Assume I= {i1, I2, i3, ... In} is a collection of items, and data set D tasks related is a collection of transaction databases, and each transaction T is a collection of a number of items to make $T \subseteq I$. Each transaction has a representation, called TID. The transaction T contains a collection of items A if and only if $A \subseteq T$. An association rule is a form of $A \Rightarrow B$ logical implication, among them $A \subset I$, $B \subset I$ and $A \cap B = \emptyset$ [5]. The evaluation criteria of association rules are mainly support and confidence.

Support represents frequent degree that rules are used in the collection of transactions. $\text{Support}(A \rightarrow B) = P(A \cup B)$.

Confidence determines the predictability of the rules. $\text{Confidence}(A \Rightarrow B) = P(B|A) = \text{Support}(A \cup B)/\text{Support}(A)$.

2.2 Association rule mining process

The mining process of association rules mainly contains two stages [6]: in the first phase, we must firstly find out all the items whose support is greater than minimum support from the transaction database, that is, to find all frequent item sets; the second stage is to generate the expected association rules from these frequent item sets.

3. Apriori algorithm

3.1 The basic idea

Apriori algorithm is a kind of typical breadth first search combined with direct counting algorithm, using the iterative strategy of layer by layer search. Firstly it will generate candidate set and filter the candidate set, then produce frequent sets of the layer [7].

3.2 Examples of Apriori algorithm

Take the supermarket retail industry as an example, mining the association rules through the analysis of the sales records. Known transaction database D, as shown in table 1, the database has 7 transactions, with minimum support of 2.

Table 1 Transaction database D

TID	Items
T1	beef chicken milk
T2	beef cheese
T3	cheese boots
T4	beef chicken cheese
T5	beef chicken clothes cheese milk
T6	chicken clothes milk
T7	chicken milk clothes

Step 1: scan transaction database D, count the number of each item and get candidate 1 sets C1, with the minimum support for comparison, so that get the frequent 1 sets L1, such as Fig.1;

C1

Item sets	count
{beef}	4
{chicken}	5
{milk}	4
{cheese}	4
{boots}	1
{clothes}	3

$\longrightarrow$

L1

Item sets	count
{beef}	4
{chicken}	5
{milk}	4
{cheese}	4
{clothes}	3

Fig.1 Step 1

Step 2: generate candidate 2 sets C2 by frequent 1 sets L1, and then scan transaction database D, and count the number of each item in C2 , with the minimum support for comparison, then get the frequent 2 sets L2, such as Fig.2;

C2

Item sets
{beef chicken}
{beef milk}
{beef cheese}
{beef clothes}
{chicken milk}
{chicken cheese}
{chicken clothes}
{milk cheese}
{milk clothes}
{cheese clothes}

$\longrightarrow$

C2

Item sets	counts
{beef chicken}	3
{beef milk}	2
{beef cheese}	3
{beef clothes}	1
{chicken milk}	4
{chicken cheese}	2
{chicken clothes}	3
{milk cheese}	1
{milk clothes}	3
{cheese clothes}	1

L2

$\longrightarrow$

Item sets	counts
{beef chicken}	3
{beef milk}	2
{beef cheese}	3
{chicken milk}	4
{chicken cheese}	2
{chicken clothes}	3
{milk clothes}	3

Fig.2 Step 2

Step 3: generate candidate 3 sets C3 by frequent 2 sets L2 and pruning, then scan transaction database D, count the number of each item in C3, with the minimum support for comparison, then get the frequent 3 sets L3, such as Fig.3;

C3

Item sets
{beef chicken milk}
{beef chicken cheese}
{chicken milk clothes}

$\longrightarrow$

C3

Item sets	count
{beef chicken milk}	2
{beef chicken cheese}	2
{chicken milk clothes}	3

L3

$\longrightarrow$

Item sets	count
{beef chicken milk}	2
{beef chicken cheese}	2
{chicken milk clothes}	3

Fig.3 Step 3

Step 4: generate candidate 4 sets C4 by frequent 3 sets L3 and pruning, candidate 4 sets C4 is empty, such as Fig.4;

C4

Item sets
{beef chicken milk cheese}
{beef chicken milk clothes}

Fig.4 Step 4

3.3 Advantages and disadvantages

3.3.1 Advantages

Apriori algorithm generates the candidate item sets by using Apriori, which greatly compress the candidate item sets and the size of the frequent item sets, and obtain good performance.

3.3.2 Disadvantages

1) The need for multiple scan database, system I/O load is quite large. The time of each scanning will be very long, resulting in a relatively low efficiency of Apriori algorithm.
2) It maybe produce huge candidate item sets. In the worst case, it produces the considerable proved to be non-frequent candidate item sets and the cost of counting is quite high, especially when the candidate set is relatively long, time and space is a challenge.

4. FP-growth algorithm

4.1 The basic idea

Aiming at the bottleneck problem of Apriori algorithm, a method called FP-growth is proposed by Wang. In 2000[8].The proposed algorithm only scans the database for 2 times. FP-growth algorithm is a depth first search algorithm combined with direct counting, using recursive strategy of pattern growth, it need not generate candidate sets, instead, the transaction database is compressed into a tree structure that stores only the frequent items. All the frequent item sets are obtained by recursively mining FP-tree [4] [9].

4.2 Examples of FP-growth algorithm

This instance uses the transaction database D of table 1, with minimum support of 2.

Step 1: firstly scan the database, each element is sorted descending by frequency, and delete the element whose frequency is less than the minimum support, and get frequent 1 set, as shown in table 2;

Table 2 F

Item sets	count
{chicken}	5
{beef}	4
{milk}	4
{cheese}	4
{clothes}	3

Step 2: secondly scan database, for each transaction, resort according to the order of F1, and delete elements that not belong to the frequent 1 sets of F1, such as table 3;

Table 3 Transaction database D

TID	Items
T1	chicken beef milk
T2	beef cheese
T3	cheese
T4	chicken beef cheese
T5	chicken beef milk cheese clothes
T6	chicken milk clothes
T7	chicken milk clothes

Step 3: insert each transaction record from the second step into FP-Tree, finished FP-Tree is shown in Fig.5. The suffix mode is empty at first. The frequent item table after

Sorting is [p|P]. p is the first frequent item, and P is the remaining frequent items. Call insert tree (P] [p|, T) recursively.

Step 4: find frequent items from the FP-Tree, traverse the header in each ("cheese: 4" as an example):

1) Find all the "cheese" node from the FP-Tree, traverse its ancestors, and get conditional mode, the suffix mode at this time is: (cheese:1);
2) build the FP-tree on the basis of the conditional model, as shown in Fig.6:

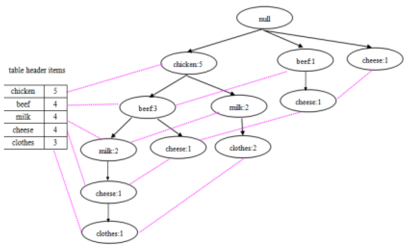

Fig.5 FP-Tree

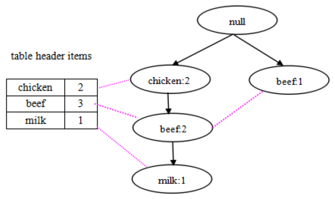

Fig.6 Conditional FP-tree

Repeat the above operation, and mine frequent patterns, such as table 4:

Table 4 Frequent pattern table

item	frequent patterns
clothes	{milk clothes:5} {clothes:5} {chicken milk clothes:5} {chicken clothes:5}
cheese	{chicken cheese:4} {cheese:4} {chicken beef cheese:4} {beef cheese:4}
milk	{beef milk:4} {milk:4} {chicken beef milk:4} {chicken milk:4}
beef	{chicken beef:4} {beef:4}
chicken	{chicken:5}

4.3 Advantages and disadvantages

4.3.1 Advantages

1) the tree - FP algorithm uses a compressed storage tree structure to access transaction records, only for the two scan, the scan time consumption is less than Apriori algorithm;
2) FP - tree algorithm does not generate candidate sets completely, and does not count the candidate item sets, so the time performance is better than the Apriori algorithm;

4.3.2 Disadvantages

1) FP-tree algorithm in mining frequent patterns inevitably need to create additional data structures, which will consume a lot of time and space;
2) For the FP - tree algorithm, the performance of the algorithm will be affected if the condition tree is very rich (in the worst case).
3) The FP - tree algorithm can only be used to excavate the Boolean association rules of a single dimension.

5. Algorithm improvement

Based on the -growth FP algorithm, the main idea is to inherit the advantages of FP-growth algorithm that need not the generation of candidate sets, firstly count the total number of frequent 1 sets marked as n, then scan the database n times, each time a database subset of every item in C1 is obtained. Then, the -growth FP algorithm is used to constrain the frequent item sets in the database subsets, and obtain the frequent item sets containing the frequent item sets .In the end, merger these frequent items and get all the frequent item sets.

6. Conclusion

In order to verify the efficiency of the algorithm, Eclipse is used to implement the above algorithms. Test data contains a number of transactions 4627, and each transaction contains 217 attributes. Support was respectively 0.1, 0.2, 0.3, 0.4 and 0.5. The test results are shown in Fig.7.

Obviously, when the minimum support degree is small, the running speed of FP-growth algorithm is much faster than the Apriori algorithm. When the minimum support degree is large, the running speed of FP-growth algorithm is faster than the new algorithm. But the new algorithm runs faster than the FP-growth algorithm when minimum support is small to a certain degree.

The reasons for this phenomenon are as follows: the time cost of FP-growth algorithm depends on the construction and mining frequent item sets, but the time cost of the new algorithm depends on the generation of the database subsets, the trees building of the database subsets and the mining of frequent item sets. With the decrease of the minimum support degree, the total number of the total number of FP-growth algorithm increases, the memory overhead and time overhead of the FP-growth algorithm are becoming more and larger, so the digging speed is slower and slower. At this time, the time cost of the new algorithm in database subset generation is increasing, but in terms of building the database subsets and constrained frequent item sets mining, due to the relatively small memory overhead, time overhead is relatively small, the overall mining speed is faster than FP-growth. The bigger database, the more obvious advantages.

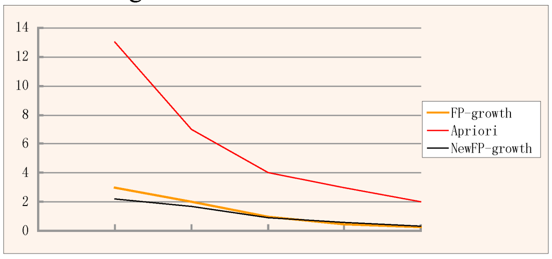

Fig.7 Results

Acknowledgments

The research work was supported by 2012 National Science and technology support program, 2012BAH87F03.

References

[1] Ke Luo, Biye Cai, Shengxian Bu. Research on data mining and its development [J]. database and information processing, 2002 14:182-184.
[2] Xiaoyi Li, Zhaodi Xu. Analysis of association rule mining algorithm [J]. Journal of Liaoning Technical University, 2006, 24 (2): 319-320.
[3] JOCHEN H, ULRICHG, GHOLAMREZA. Algorithms for association rule mining－a general survey and comparison[J]. SIGKDD Explorations，2001，2(1): 58- 64．
[4] Min Li, Chunping Li. Analysis and comparison of [4] frequent pattern mining algorithm 2005 12.
[5] Han JW, Kamber Mi,Ming Fan, Xiaofeng Meng. Data mining concepts and techniques. Beijing: Mechanical Industry Press, 2006.
[6] Xiaodong He, Weiguo Liu. Association rule algorithm in data mining [J]. computer engineering and design.2005.
[7] Lili KUANG. Discussion on Apriori algorithm and FP-tree algorithm. Huaibei Coal Industry Teachers College (NATURAL SCIENCE EDITION), 2010, 31 (2).
[8] Usama M Fayyad,Gegory Piatetsky Advance In knowledge discovery and data mining California AAAI/MIT Press 1996.
[9] Zhixin Feng, Cheng Zhong. Mining maximal frequent patterns based on FP-Tree algorithm. Computer Engineering, 2004, 6 (11): 123-126.