The Optimization and Improvement of the Apriori Algorithm

Yiwu Xie, Yutong Li, Chunli Wang, Mingyu Lu
Information Science Technology Department,
Dalian Maritime University, Dalian, Liaoning, 116026, China
dearliyutong@163.com

Abstract

Through the study of Apriori algorithm we discover two aspects that affect the efficiency of the algorithm. One is the frequent scanning database, the other is large scale of the candidate itemsets. Therefore, IApriori algorithm is proposed that can reduce the times of scanning database, optimize the join procedure of frequent itemsets generated in order to reduce the size of the candidate itemsets. The results show that the algorithm is better than Apriori algorithm.

1. Introduction

Data mining also known as the knowledge discovery in database, it is a forecasting method to extract the hidden knowledge from the large-scale database or the data warehouse [1]. Its main methods include: classification, clustering, association analysis and so on [2]. Mining association rules is one of the main contents of data mining research at present and emphasis particularly is finding the relation of different items in database. Apriori [3, 4] is the most famous and basic method in mining association rules. The principle of Apriori algorithm is to find the valuable association rules whose support and confidence must satisfy the minimum support and confidence confirmed by user aforehand.

2. Apriori algorithm description

Apriori algorithm is the most effective method that candidate k+1-itemsets may be generated from frequent k-itemsets according to the nature of Apriori algorithm that any subset of frequent itemsets are all frequent itemsets. The classical Apriori algorithm:

L_1 = {large 1-itemsets};
for ($k=2$; $L_{k-1} \neq \varnothing$;k++) do begin
  C_k=apriori-gen(L_{k-1});
//new candidate itemsets generated
  for all transactions $t \in D$ do begin
    C_t=subset(C_k,t);
//transaction t contains in the candidate itemsets
    for all candidates $c \in C_t$ do
      c.count++;
    end
    $L_k=\{c \in C_k \mid c.count \geq minsup\}$
  end
Answer=$\bigcup_k L_k$;

First, find the frequent 1-itemsets L_1.L_2 is generated from L_1 and so on, until no more frequent k-itemsets can be found and then algorithm ceases. In the algorithm each L_k generated must scan database one time [5-7].Second, C_k is generated from L_{k-1}.Every itemset in C_k is tested whether its all k-1 subsets constitute a large k-1 itemset or not. Therefore, scanning large database many times and generating so many useless candidate itemsets are the bottleneck of the Apriori algorithm.

3. Problem description

Apriori algorithm has three shortcomings below [8]. First, because there are lots of transactions in the database, computing the frequencies of candidate itemsets must scan database frequently and spend much more time. Second, C_k is generated by joining two frequent itemsets that belong to L_{k-1}. If we can reduce the operating frequencies, we can improve the efficiency of Apriori algorithm. Third, the expending in time and space for frequent itemsets is too much. For example, if we have 10^4 frequent 1-itemsets, we can get 10^7 frequent 2-itemsets. So reducing the scale of C_k can improve the efficiency of the Apriori algorithm greatly.

Therefore, it is necessary to delete the useless transactions in the database in order to reduce the scale of database and reduce itemsets generated from C_k by the join procedure. This paper presents an improved Apriori algorithm called IApriori algorithm according to the analysis of Apriori algorithm above.

4. Improved apriori algorithm

Improved Apriori algorithm related in [9] scans database to compute the frequencies of candidate itemsets at the same time to mark the deleting tag if the size of

978-0-7695-3505-0/08 $25.00 © 2008 IEEE
DOI 10.1109/IITA.Workshops.2008.170

transaction t is less than k. In [10, 11], transaction t does not contain any element of candidate itemsets C_k, the algorithm determines to mark the deleting tag on t, so we can skip over the record in next database scanning. The method that reduces the operating frequencies generating C_k by joining two frequent itemsets that belong to L_{k-1} is proposed in [12].The improvement in [13] limits the scale (m+n) of C_k in advance according to the numbers of rules that contain the total items which are respectively m and n.

Due to the generalization of algorithm, IApriori algorithm proposed in this paper does not input more thresholds to improve efficiency, only has improved from some aspects.

(1) Improvement method of optimizing the join procedure to reduce the size of candidate itemsets. When C_{k+1} is generated from $L_k \infty L_k$, each item of the first L_k is joining with every item in the second L_k in classical Apriori algorithm. Since the k^{th} item of the first L_k is the same as the k^{th} item of the second one, the k^{th} item of the first L_k is joining with the $k+1^{th}$ item of the second one, when C_{k+1} is generated from $L_k \infty L_k$. Only in this way can we spend less time than before.

For example: L_1= {I_1, I_2, I_3, I_4}, Apriori algorithm needs computing 16(4*4) times but it only needs 6(3+2+1) times in IApriori algorithm.

(2) Improvement method of reducing the scale of database. Through the process obtaining C_{k+1} from L_k if the size of transaction t is less than k, we can say that it is useless for generating C_{k+1}; if transaction t does not contain any subset of candidate itemsets C_k, mark the transaction t the deleting tag.

IApriori algorithm reduces the scale of database and optimizes the join procedure, so it improves the efficiency of algorithm greatly.

Moreover, the scale of L_1 must be small as soon as possible to reduce the scale of C_2, C_3 and so on. Make sure the proper minsupport; also we can choose interested items for the frequent itemsets generated.

IApriori algorithm is described as follows.

```
1)   L1 = {large 1-itemsets};
2)   for (k=2; Lk-1 ≠ ∅ ; k++)
3)   {
4)       Ck=apriori-gen(Lk-1);
       //generate new candidate itemsets
5)       for all transactions t∈ D  and  t.delete=0
6)       {
7)           if  t.count<k  then
       // if the size of transaction t is less than k, t is
     useless for Ck generated
8)           t.delete=1
       //mark t the deleting tag to skip over the record
     in next database scanning
9)           else
10)          {
11)              Ct=subset(Ck,t);
       //candidate itemsets contained in transcaion t
12)              if  Ct = ∅  then
       // if t does not contain any subset of candidate
     itemsets Ck, mark the deleting tag
13)              t.delete=1;
14)              else
15)              {
16)                  for all candidates c∈ Ct
17)                  c.count++;
18)              }
19)          }
20)      }
21)  Lk={c∈ Ck |c.count ≥ minsup}
22)  }
23)  Answer=□kLk;
```

To implement the improvement, IApriori algorithm is described as follow steps:

(1) Scan the database to get C_1, make sure the proper minsupport to get frequent itemset L_1(k=1).

(2) Generate C_{k+1} though joining two frequent itemsets that belong to L_k. The m^{th} item in the first L_k should join with the $m+1^{th}$ item of the second L_k.

(3) If the size of transaction t is less than k, we give transaction t to mark the deleting tag; if not, compute C_t that contains subsets of candidate itemsets C_k in transaction t.

(4) Judge whether t comprises any subset of C_k; if not, compute the frequencies of C_k. Otherwise, mark t the deleting tag.

(5) Add item of C_k to L_{k+1} if the support of the item is greater than minsupport.

(6) If $L_{k+1}= \varnothing$, the algorithm ceases. Otherwise, k=k+1, continue to the second step in circle until $L_{k+1}= \varnothing$.

5. IApriori algorithm performance test

To analyse the relative performance of the IApriori and Apriori algorithms, we use a small part data from real store database stored 10000 transactions. Figure 1 demonstrates the relative performance of these algorithms. Five experiments are carried out accomplished using the same database with different minimum support factors. The experiment is in WindowsXP Professional operating system, CPU with Intel (R) 2.93GHz, memory with 512MB, the algorithm language used in C #.

Experiments results show that the time needed IApriori algorithm is less than Apriori algorithm under the same support condition. So we can have the conclusion that the proposed algorithm outperforms the Apriori algorithm in computational time.

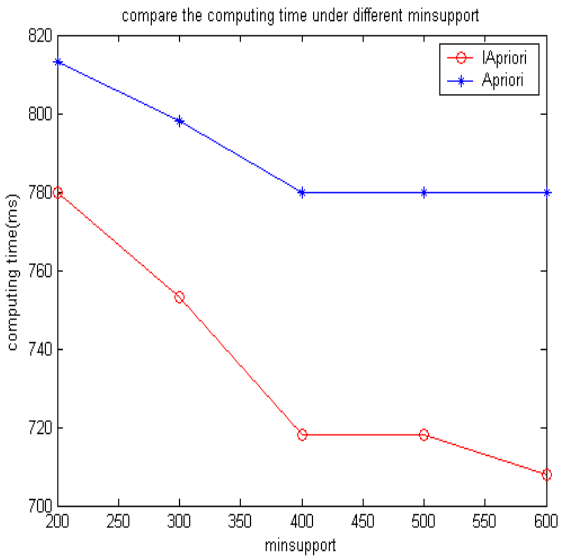

Figure 1 Relative performance under different minsupport

6. Conclusions

In this paper we discuss the problems exist in scanning database frequently and the large scale of candidate itemsets in Apriori algorithm, present an improved algorithm IApriori algorithm. It not only decrease the times of scanning database but also optimize the process that generates candidate itemsets. Experiments results show that the proposed algorithm outperforms the Apriori algorithm in computational time.

Acknowledgement

This research was supported by Natural Science Foundation of China (No.60473115, No.60773084, No. 60603023) and Doctoral Fund of Ministry of Education of China (No. 20070151009).

References

[1] Jiawei Han, M. Kamber, "*Data Mining-Concepts and Techniques*", Morgan Kaufmann Publishers, Sam Francisco, 2001.

[2] S. Brin, R. Motwani, C. Silverstein, "Beyond Market Basket: Generalizing Association Rules to Correlation". In*Proc.1997ACM-SIGMOD Int. Conf. Management of Data (SIGMOD97)*, Tucson, AZ. 1997, pp.265-276.

[3] M.B. Elsen, P.T. Speliman, and P.O. Brown, "Cluster Analysis and Display of Genome-wide Expression Patterns", *Proc. Natl. Acad. Sci*, USA, 1998, pp.14863-14868.

[4] Agrawal, Rakesh, "Fast Algorithms for Mining Association Rules in Large Databases", *Proceedings of the ACM SIGMOD International Conference Management of Data*, Washington, 1993, pp.207-216.

[5] ZAKI, J. Mohammed, PARTHASARATHY, Srinivasan, Wei Li, "New Algorithms for Fast Discovery and Data Rules." *In 3rd Intl. Conf. On Knowledge Discovery and Data Mining*,1997

[6] A. Savasere, E. Omiecinski, and S. B. Navathe, "Mining for Strong Negative Associations in a Large Database of Customer Transactions", *Proceeding of the 14th International Conference on Data Engineering*. Orlando, Florida, USA: IEEE Computer Society Press, 2004, pp.494-502.

[7] Xiao-ping Yang. "Improvement of Apriori Algorithm for Association Rules", *Journal of Zhejiang Ocean Universty (Natural Science)*, 2006, V01.25, No.2, pp.176-182.

[8] Jie Gao, Shao-jun Li, "A Method of Improvement and Optimization on Association Rules Apriori Algorithm", *Proceedings of the 6th World Congress on Intelligent Control and Automation,* 2006, pp.5901-5905.

[9] Sheng Chai, Jia Yang, and Yang Cheng, "The Research of Improved Apriori Algorithm for Mining Association Rules". *Proceedings of the Service Systems and Service Management*, 2007.

[10] Zhen Zhu, Jing-yan Wang, "Book Recommendation Service by Improved Association Rule Mining Algorithm", *Proceedings of the 6th International Conference on Machine Learning and Cybernetics*, 2007, pp.3864-3869.

[11] Peng Gong, Chi Yang, and Hui Li, "The Application of Improved Association Rules Data Mining Algorithm Apriori in CRM", *Proceedings of 2nd International Conference on Pervasive Computing and Applications*, 2007.

[12] Jun-Peng Yuan, Dong-Hua Zhu, "A Survey of Association Rules of Apriori Algorithms", *Computer Science,* Vol.31, No.1, pp.114-117.

[13] Sandro Da Silva Camargo, "MiRABIT: A New Algorithm for Mining Association Rules", *Proceedings of the 12th International Conference of the Chilean Computer Science Society*, 2002.