Improving Efficiency of Apriori Algorithm Using Transaction Reduction

Jaishree Singh*, Hari Ram, Dr. J.S. Sodhi*****

Department of Computer Science & Engineering, Amity School of Engineering and Technology, Amity University, Sec-125 NOIDA, (U.P.),India

Abstract- Association rules are the main technique to determine the frequent itemset in data mining. Apriori algorithm is a classical algorithm of association rule mining. This classical algorithm is inefficient due to so many scans of database. And if the database is large, it takes too much time to scan the database. In this paper, we proposed an Improved Apriori algorithm which reduces the scanning time by cutting down unnecessary transaction records as well as reduce the redundant generation of sub-items during pruning the candidate itemsets, which can form directly the set of frequent itemsets and eliminate candidate having a subset that is not frequent.

Index Terms- apriori algorithm, association rules, candidate-itemsets, data mining

I. INTRODUCTION

Data mining also known as Knowledge Discovery in Database(KDD). The purpose of data mining is to abstract interesting knowledge from the large database. From the analysis of abstracted patterns, decision-making process can be done very easily.

Association rule is based mainly on discovering frequent itemsets. Association rules are frequently used by retail stores to assist in marketing, advertising, inventory control, predicting faults in telecommunication network.

Apriori algorithm represents the candidate generation approach. It generates candidate (k+1) itemsets based on frequent k-itemsets. Apriori is a Breadth First Search Algorithm (BFS).

II. BASIC CONCEPTION

A. Association Rules [3]

The concept of the association rules was first proposed by R.Agrawal.It is used to describe the patterns of customers' purchase in the supermarket. The association rules can be formally defined as

Definition 1: Let I={i1,i2,i3,….in}be finite itemsets. D is a transactional database. Where ik(k∈{1,2,… … m}) is an item, and Tid is the exclusive identifier of transaction T in transactional database.

Definition 2: Let X⊂I,Y⊂I, and X∩Y =∅.The implication of the form X=>Y is called an association rules. Definition 3:Let D is a transactional database. If the percentage of transactions in D that contain X ∪ Y is s%, the rule X=>Y holds in D with Support s. If the percentage of transactions in D containing X that also contain Y is c%, the rule X=>Y has Confidence c.The definitions of probability are,

$$\text{Support}(X \Rightarrow Y) = P(X \cup Y) \quad (1)$$
$$\text{Confidence}(X \Rightarrow Y) = P(Y|X) \quad (2)$$

Rules that satisfy both minimum support threshold (minsup) and minimum confidence threshold (minconf) are called strong rules.

Definition 4: If the support of itemsets X is greater than or equal to minimum support threshold, X is called frequent itemsets. If the support of itemsets X is smaller than the minimum support threshold, X is called infrequent itemsets.

III. TYPICAL APRIORI ALGORITHM

A. Description of the Typical Apriori Algorithm

1) Typical Apriori algorithm[7]

Apriori employs an iterative approach known as a levelwise search, where k-itemsets are used to explore (k+1)-itemsets. First, the set of frequent 1-itemsets is found by scanning the database to accumulate the count for each item,and collecting those items that satisfy minimum support. The resulting set is denoted L1. Next, L1 is used to find L2,the set of frequent 2-itemsets, which is used to find L3, and so on, until no more frequent k-temsets can be found. The finding of each Lk requires one full scan of the database.To improve the efficiency of the level-wise generation of frequent itemsets, an important property called the Apriori property, presented is used to reduce the search space. Apriori property: All nonempty subsets of a frequent itemset must also be frequent. A two-step process is used to find the frequent itemsets: join and prune actions.

a) The join step

To find Lk a set of candidate k-itemsets is generated by joining Lk-1 with itself. This set of candidates is denoted Ck.

b) The prune step

The members of Ck may or may not be frequent, but all of the frequent k-itemsets are included in Ck. A scan of the database to determine the count of each candidate in Ck would result in the determination of Lk (i.e., all candidates having a count no less than the minimum support count are frequent by definition, and therefore belong to Lk). To reduce the size of Ck, the Apriori property is used as follows. Any (K-1)-itemset that is not frequent cannot be a subset of a frequent k-itemset. Hence, if any (K-1)-subset of a candidate k-itemset is not in Lk-1, then the candidate cannot be frequent either and so can be removed from Ck.

2) Description of the algorithm

Input: D, Database of transactions; min_sup, minimum support threshold
Output: L, frequent itemsets in D
Method:
(1) L1=find_frequent_1-itemsets(D);
(2) for(k=2; Lk-1≠Φ; k++){
(3) Ck=apriori_gen(Lk-1, min_sup);
(4) for each transaction t∈D{
(5) Ct=subset(Ck,t);
(6) for each candidate c∈Ct
(7) c.count++ ;
(8) }
(9) Lk={ c∈Ck |c.count≥min_sup }
(10) }
(11) return L=UkLk ;

Procedure apriori_gen(Lk-1:frequent(k-1)-itemsets)
(1) for each itemset l1∈ Lk-1{
(2) for each itemset l2∈ Lk-1{
(3) if(l1 [1]= l2 [1])∧ (l1 [2]= l2 [2]) ∧…∧(l1 [k-2]= l2 [k-2]) ∧(l1 [k-1]< l2 [k-1]) then {
(4) c=l1∞l2;
(5) if has_infrequent_subset(c, Lk-1) then
(6) delete c;
(7) else add c to Ck ;
(8) }}}
(9) return Ck;

Procedure has_infrequent_subset(c: candidate k-itemset; Lk-1:frequent(k-1)-itemsets)
(1) for each(k-1)-subset s of c {
(2) if s ∉ Lk-1 then
(3) return true; }
(4) return false;

IV. PRINCIPLE OF IMPROVED ALGORITHM

Classical Apriori algorithm generates large number of candidate sets if database is large. And due to large number of records in database results in much more I/O cost. In this project, we proposed an optimized method for Apriori algorithm which reduces the size of database. In our proposed method, we introduced an attribute Size_Of_Transaction (SOT) , containing number of items in individual transaction in database. The deletion process of transaction in database will made according to the value of K. Whatever the value of K, algorithm searches the same value for SOT in database. If value of K matches with value of SOT then delete only those transaction from database.
Table 1 is generating frequent itemsets by using proposed algorithm very efficiently.

Description of the algorithm

Input: D: Database of transactions; *min_sup*: minimum support threshold

Output: L: frequent itemsets in D
Method:
1) L1=find_frequent_1-itemsets(D);
2) For(k=2;Lk-1≠∅; k++){
3) Ck=apriori_gen(Lk-1, *min_sup*);
4) for each transaction t∈D{
5) Ct=subset(Ck,t);
6) for each candidate c∈Ct
7) c.count++;
8) }
9) Lk={ c∈Ck |c.count≥*min_sup* };
10) if(k>=2){
11) delete_datavalue(D, Lk, Lk-1);
12) delete_datarow (D, Lk); }
13) }
14) return L=UkLk ;

Procedure apriori_gen(Lk-1:frequent(k-1)-itemsets)
1) for each itemset l1∈ Lk-1 {
2) for each itemset l2∈ Lk-1 {
3) if(l1 [1]= l2 [1])∧ (l1 [2]= l2 [2]) ∧…∧(l1 [k-2]= l2 [k-2]) ∧(l1 [k-1]< l2 [k-1]) then {
4) c=l1 ∞l2;
5) for each itemset l1∈Lk-1 {
6) for each candidate c ∈Ck {
7) if l1 is the subset of c then
8) c.num++; }}}}}
9) C'k={ c∈Ck |c.num=k};
10) return C'k;

Procedure delete_datavalue (D:Database; Lk: frequent (k)-itemsets; Lk-1: frequent(k-1) - itemsets)
1) for each itemset i ∈Lk-1 and i ∉ Lk{
2) for each transaction t∈D{
3) for each datavalue∈t{
4) if (datavalue=i)
5) update datavalue=null;
6) }}

Procedure delete_datarow (D: Database; Lk:frequent(k) - itemsets)
1) for each transaction t∈D{
2) for each datavalue∈t{
3) if(datavalue!=null and datavalue!=0){
4) datarow.count++; }}
5) if(datarow.count<k){
6) delete datarow;}
7)}

Example of Algorithm

Transaction database D is shown in Table 1. Suppose the minimum support count *min_sup=3*. The algorithm is as follows:
Step 1: First we have to convert database into the desired database that is with SOT column.

Step 2: In the first iteration of the algorithm, each item is a member of the set of candidate 1-itemsets, C1. The algorithm simply scans all of the transactions in order to count the number of occurrences of each item.

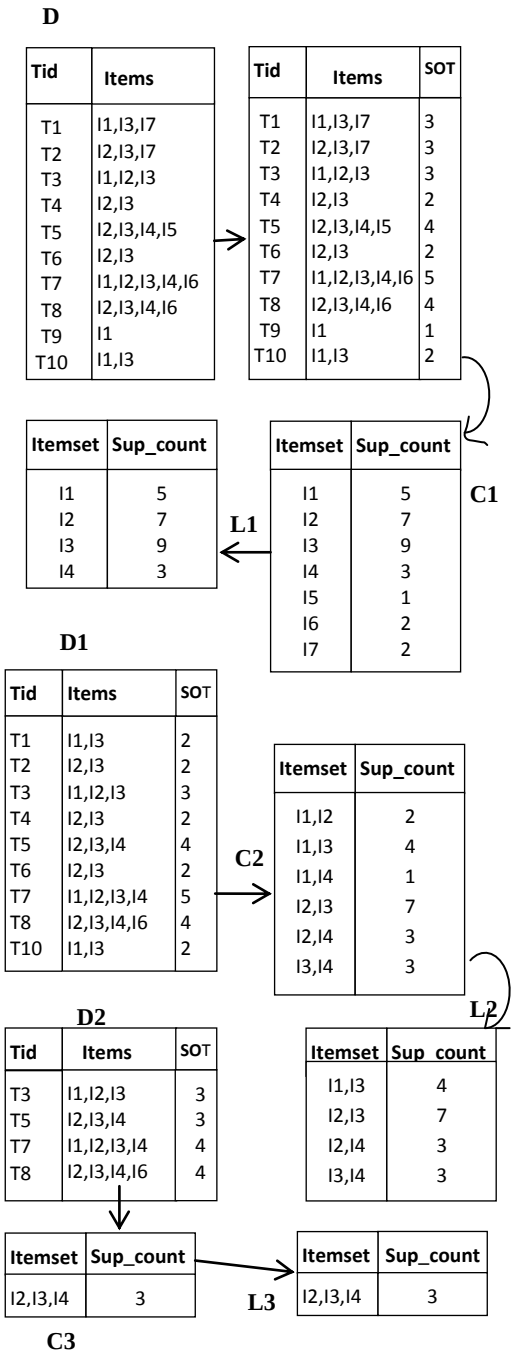

Table 1: Example of algorithm

Step 3: This algorithm will then generate number of items in each transaction. We called this Size_Of _Transaction (SOT).

Step 4: Because of min_sup=3, the set of frequent 1-itemset, L1 can be determined. It consists of the candidate 1-itemset, C1, satisfying minimum support.

Step 5: Since the support count of I5,I6,I7 are less than 3, they won't appear in L1. Delete these data from D. In addition, when L1 is generated, now, the value of k is 2, delete those records of transaction having SOT=1 in D. And there won't exist any elements of C2 in the records we find there is only one data in the T9. We delete the data and obtain transaction database D1.

Step 6: To discover the set of frequent 2-itemsets, L2, the algorithm uses the join L1 ∞L1 to generate a candidate set of 2-itemsets, C2.

Step 7: The transactions in D1 are scanned and the support count and SOT of each candidate itemset in C2 is accumulated.

Step 8: The set of frequent 2-itemsets, L2, is then determined, consisting of those candidate 2-itemsets in C2 having minimum support.

Step 9: After L2 is generated, we can find the transaction record of T1, T2,T4, T6, T10 are only two in D1.Now,the value of k is 2,delete those records of transaction having SOT=2.And there won't exist any elements of C3 in the records. Therefore, these records can be deleted and we obtain transaction database D2.

Step 10: To discover the set of frequent 3-itemsets, L3, the algorithm uses the join L2∞ L2 to generate a candidate set of 3-itemsets C3, where C3= L2 ∞L2= I1, I2, I3},{ I2, I3, I4}}. There are a number of elements in C3. According to the property of Apriori algorithm, C3 needs to prune. Because {I1, I2} not belongs to L2, we remove it from C3. Because the 2-subsets {I2, I3}, I2, I4} and {I3, I4} all belong to L2, they should remain in C3.

Step 11: The transactions in D2 are scanned and the support count of each candidate itemset in C3 is accumulated. Use C3 to generate L3.

Step 12: L3 has only one 3-itemsets so that C4 =Φ. The algorithm will stop and give out all the frequent itemsets.

Step 13: Algorithm will be generated for Ck until Ck+1 becomes empty.

V. CONCLUSION AND FUTURE SCOPE

In this paper, Apriori algorithm is improved based on the properties of cutting database. The typical Apriori algorithm has performance bottleneck in the massive data processing so that we need to optimize the algorithm with variety of methods. The improved algorithm we proposed in this paper not only optimizes the algorithm of reducing the size of the candidate set of k-itemsets, Ck, but also reduce the I / O spending by cutting down transaction records in the database. The performance of Apriori algorithm is optimized so that we can mine association information from massive data faster and better.

Although this improved algorithm has optimized and efficient but it has overhead to manage the new database after every generation of Lk. So, there should be some approach which has very less number of scans of database. Another solution might be division of large database among processors.

REFERENCES

[1] Lu, Lin; Pei-qi;, "Study on improved apriori algorithm and its application in supermarket," Information Sciences amd Interaction Sciences (ICSI), 20103rd International Conference on,vol.,no.,pp.441-443, 23-25 June 2010.

[2] Sheng Chai; Jia Yang; Yang Cheng;," The Research of Improved Apriori Algorithm for Mining Association Rules," Service System and Service Management, 2007 International Conference on, vol., no.,pp.1-4, 9-11 June 2007

[3] Wanjun Yu; Xiaochun Wang; Erkang Wang; Bowen Chen;, "The research of improved apriori algorithm for mining association rules," Communication Technology, 2008. ICCT 2008 11th IEEE International Conference on, vol., no.,pp.513-516, 10-12 Nov. 2008.

[4] Yiwu Xie; Yutong Li; Chunli Wang; Mingyu Lu;, "The Optimization and Improvement of the Apriori Algorithm," Education Tecnology and Training, 2008. And 2008 International Workshop on Geoscience and Remote Sensing. ETT and GRS 20008.

[5] Sixue Bai, Xinxi Dai, "An Efficiency apriori Algorithm: P_Matrix Algorithm," isdpe, pp.101-103, The First International Symposium on Data, Privacy, and E-Commerce (ISDPE 2007), 2007

[6] Yan-hua Wang; Xia Feng; , "The Optimization of Apriori Algorithm Based on Directed Network," Intelligent Information Technology Application, 2009. IITA 2009. Third International Symposium on, vol.3, no., pp.504-507, 21-22 Nov. 2009.

[7] Zhang Changsheng; Li Zhongyue; Zheng Dongsong; , "An Improved Algorithm for Apriori," Education Technology and Computer Science, 2009. ETCS '09. First International Workshop on , vol.1, no., pp.995-998, 7-8 March 2009

[8] Rui Chang; Zhiyi Liu; , "An improved apriori algorithm," Electronics and Optoelectronics (ICEOE), 2011 International Conference on , vol.1, no., pp.V1-476-V1-478, 29-31 July 2011

[9] Du Ping; Gao Yongping; , "A new improvement of Apriori Algorithm for mining association rules," Computer Application and System Modeling (ICCASM), 2010 International Conference on , vol.2, no., pp.V2-529- V2-532, 22-24 Oct. 2010.

[10] 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.

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

[12] 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.

[13] A. Savasere, E. Omiecinski, and S. Navathe, "An Efficient Algorithm for Mining Association Rules in Large Databases", In VLDB'95, pp.432-443, Zurich, Switzerland.

[14] H. Jiawei and M. Kamber, Data Mining Concepts and Techniques, Beijing Higher Education, 2001.

[15] Y. Tang, Y. Wang and H. Yang, "Optimized Method for Mining Maximum Frequent Itemsets," Comput. Eng. Appl., Beijing, vol. 42(31), pp. 171-173, 2006.

[16] L. Cheng and B. B. Wang, "An Improved Apriori Algorithm for Mining Association Rules, " Comput. Eng., Shanghai, vol. 28(7), pp. 104-105, 2002.

[17] Z. Y. Xu and C. Zhang, "An Optimized Apriori Algorithm for Mining Association Rules," Comput. Eng., Shanghai, vol. 29(19), pp. 83-84, 2003.

[18] Luo Ke,Wu Jie.Apriori algorithm based on the improved [J].Computer Engineering and application ,2001,20:20-22.

[19] Li Xiaohong,Shang Jin.An improvement of the new Apriori algorithm [J].Computer science, 2007,34 (4) :196-198. 2007

AUTHORS

First Author – Jaishree Singh is pursuing M. Tech (CS&E) from Amity University, Uttar Pradesh, India. Her research areas include Data Mining and Software Engineering.

Second Author – Hari Ram is a Senior Solution Integrator in Ericsson, Uttar Pradesh, India. He has done MCA. His research areas include Telecommunication and Data Mining.

Third Author – Dr. J.S. Sodhi is a Head-IT & CIO (Assistant Vice President) with AKC Data Systems (An Amity Group and AKC Group Company), Amity University, Uttar Pradesh, India. He is Ph. D. in Information Security Management and Certified Security Compliance Specialist (CSCS). His research area includes Network security. He has published number of research papers in reputed National & International Journals.