Mining of Frequent Itemsets with an Enhanced Apriori Algorithm

V.Vijayalakshmi
Research Scholar
Manonmaniam Sundaranar University
Tirunelveli, T.N

A.Pethalakshmi, Ph. D
Associate Professor of Comp.Science
M.V.M Government Arts College
Dindigul, T.N

ABSTRACT
Apriori algorithm is a classical algorithm of association rule mining and widely used for mining association rule which uses frequent item. 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. To reduce these two limitations, this paper proposes a new technique called TR-BAM for mining frequent patterns in large databases by implementing a Bit Array Matrix. The whole database is scanned only once and the data is compressed in the form of a Bit Array Matrix. The frequent patterns are then mined directly from this Matrix. Appropriate operations are designed and performed on matrices to achieve efficiency.

Keywords
Association Rule, Frequent Item Set, Apriori, Bit Array Matrix

1. INTRODUCTION
The challenging task is to extracting useful information from the large collection of data in Dataware house and data base. Around the world lot of research is underway to discover the knowledge from the large collection of data in data warehouse. In this process many algorithms has been proposed to identify the associations between the data in the database, leads to mine the association rule among the data. Association rules are used for knowledge discovery and to take useful managerial decision in the organization based on the results of associations among data stepping toward to make a smarter system. In this regard, the first algorithm Apriori was proposed in the year 1994 by Agarwal and Srikanth to mine the frequent item set. Time constraint and efficiency of algorithms leads to lot of research in the area of algorithm to build efficient algorithm which takes less time and few number of database scans to mine frequent item set and association rule.

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

The remaining part of this paper is organized as follows: Section 2 contains Apriori Algorithm with worked example. In section 3, elaborate the proposed algorithm for transaction reduction technique and worked example. Section 4 contains conclusion.

2. APRIORI ALGORITHM FOR GENERATING FREQUENT ITEMSETS
Apriori algorithm employs an iterative approach known as level-wise search, where k-item sets are used to explore (k+1)-item sets. First, the set of frequent 1-itemsets L_1 is found. Next, L_1 is used find the set of frequent 2-itemsets L_2. Then L_2 is used to find the set of frequent 3-itemsets L_3.The method iterates like this till no more frequent k-item sets are found.

We can easily understand the concepts used by the Apriori with the help of following example. Table 1 shows a transactional database having 9 transactions. *TID* is a unique identification given to the each transaction. Let minimum support value be 2 (min_sup=2)

TABLE 1

TID	*ITEMS*
T1	I1,I2,I5
T2	I2,I4
T3	I2,I3
T4	I1,I2,I4
T5	I1,I3
T6	I2,I3
T7	I1,I3
T8	I1,I2,I3,I5
T9	I1,I2,I3

C_1		***L_1***	
Itemsets	***Support***	***Itemsets***	***Support***
I1	6	I1	6
I2	7	I2	7
I3	6	I3	6
I4	2	I4	2
I5	2	I5	2

C_2		***L_2***	
Itemsets	***Support***	***Itemsets***	***Support***
I1,I2	4	I1,I2	4
I1,I3	4	I1,I3	4
I1,I4	1	I1,I5	2
I1,I5	2	I2,I3	4
I2,I3	4	I2,I4	2
I2,I4	2	I2,I5	2
I2,I5	2		
I3,I4	0		
I3,I5	1		
I4,I5	0		

C_3		L_3	
Itemsets	***Support***	***Itemsets***	***Support***
I1,I2,I3	2	**I1,I2,I3**	2
I1,I2,I5	2	**I1,I2,I5**	2

Figure 1. Generation of frequent item sets by applying Apriori Algorithm

2.1 *Pseudo Code*

Initialize: k := 1, C_1 = all the 1- item sets;
read the database to count the support of C_1 to determine L_1.
L_1 := {frequent 1- item sets}; k:=2; //k represents the pass number//
while ($L_{k-1} \neq ø$) do
begin
C_k := gen_candidate_itemsets with
the given L_{k-1}
prune(C_k)
for all transactions t € T do
increment the count of all candidates
in C_K that are contained in t;
L_k := All candidates in C_k with
minimum support ;
k := k + 1;
end
Answer := U_k L_k

2.2 *Drawbacks of Apriori Algorithm*

- Large Number of in-frequent itemset are generated which increases the space complexity.
- Too many database scans are required because of large number of itemsets generated.
- As the number of database scans are more the time complexity increases as the database increases.

3. PROPOSED ALGORITHM THOUGHT AND WORKED EXAMPLE

In this method the frequent item sets are generated directly from the matrix which is generated from the transactional database. The major advantage of this approach is that, the number of database scans is greatly reduced, since the number of combinations made is far less compared to the original Apriori algorithm and thus results in reduction of time and space.

The proposed algorithm uses the following properties:

1. All the non empty subsets of a frequent itemset must also be frequent.
So, there is no need to consider those frequent itemsets which are having non frequent subsets.

2. Number of times transaction repeated in the database is represented by the count in the RC column and a new sum row stores the corresponding number of nonzero elements in the column on the bottom of the Bit Array Matrix.

3. Support count for 1 itemset is sum of nonzero elements of each column.
4. Support count for k itemsets (Ii,Ij…….,Ik)

$$Sup\_count(Ii,Ij,I_k)= RC_1x(T_{1i}.T_{1j} .T_{1k})+\ldots\ldots+ RC_nx(T_{ni}.T_{nj}..T_{nk.}) \ldots\ldots 1$$

The steps of proposed algorithm are as follows:
1. First scan the database to find the different items occurring in the database and then make the Matrix by writing all the transactions. Add column RC on the right side of the Bit Array Matrix. Add a sum row at that bottom of the Bit Array Matrix. We sum nonzero elements of each column, putting into the corresponding sum row. In the matrix, each row represents a transaction T_i each column represents one item Ij. In the Bit Array Matrix, if the transaction T_i includes Ij , then $T_{ij} = 1$, otherwise $T_{ij} = 0$.

The Bit Array Matrix is as follows

	I1	***I2***	***I3***	***I4***	***I5***	***RC***
I1,I2,I5	1	1	0	0	1	
I2,I4	0	1	0	1	0	
I2,I3	0	1	1	0	0	
I1,I2,I4	1	1	0	1	0	
I1,I3	1	0	1	0	0	
I2,I3	0	1	1	0	0	
I1,I3	1	0	1	0	0	
I1,I2,I3,I5	1	1	1	0	1	
I1,I2,I3	1	1	1	0	0	
SUM	**6**	**7**	**6**	**2**	**2**	

2. In Bit Array Matrix, the summation of nonzero elements in each column is the supporting count of item I_j. Set the minimum support count as min_ sup=2, when item supporting count is less than min_sup, all itemsets containing the I_j are infrequent itemsets. So the column can be removed directly, which will not affect the solution of frequent itemsets. Move all those transactions from C1 to L1 whose sum value is not less than min_support (min_sup=2).

3. If a transaction is occurring more than once in the transactional database then updates the repetition column (RC) of the Bit Array Matrix. Delete unnecessary repeating row. If the transaction doesn't repeat then RC i.e. repetition column for the transaction is set to 1.

	I1	***I2***	***I3***	***I4***	***I5***	***RC***
I2,I4	0	1	0	1	0`	**1**
I1 I2 I5	1	1	0	0	1	**1**
I1,I2,I4	1	1	0	1	0	**1**
I1,I3	1	0	1	0	0	**2**
I2,I3	0	1	1	0	0	**2**
I1,I2,I3	1	1	1	0	0	**1**
I1,I2,I3,I5	1	1	1	0	1	**1**

Figure 2 Generation of Bit Array Matrix

4. Now for the generation of C_2, consider the Bit Array Matrix again. Calculate the support count for 2 item sets by using the following formula

Sup_count(I_{12})=1x(0.1)+ x(1.1)+1x(1.1)+2x(1.0)+2x(1.0)+1x(1.1)+1x(1.1) = 4

Sup_count(I_{12})=4 Sup_count(I_{13})=4 Sup_count(I_{14})=1
Sup_count(I_{15})=2 Sup_count(I_{23})=4 Sup_count(I_{24})=2
Sup_count(I_{25})=2 Sup_count(I_{35})= 2

Then move only those item sets from C2 to L2 whose support count value is not less than minimum support. $I_{12,}$ $I_{13,}$ I_{15} , $I_{23,}$ $I_{24,}$ $I_{25,}$ I_{35} will be frequent 2 itemsets.

5. Next, Consider all the 3-itemsets combinations of the items The various combinations possible are {I1, I2, I5}; {I1, I2, I4}; {I1, I2, I3}; {I2, I3, I5}; {I1, I3, I5}. Now, by the property i.e. all the non empty subsets of a frequent itemset must also be frequent,it is found that itemsets {I1, I2, I4};{I2, I3, I5};{I1, I3, I5} contain subsets which are not frequent. Therefore these itemsets are not included in C_3. C_3 will contain {I1, I2, I3} and {I1, I2, I5} If a transaction is occurring more than once, then updates the repetition column.

	I1	*I2*	*I3*	*I4*	*I5*	*RC*
I2,I4	0	1	0	1	0`	**1**
I1 I2 I5	1	1	0	0	1	**1**
I1,I2,I4	1	1	0	1	0	**1**
I1,I3	1	0	1	0	0	**2**
I2,I3	0	1	1	0	0	**2**
I1,I2,I3	1	1	1	0	0	**1**
I1,I2,I3,I5	1	1	1	0	1	**1**

6.Now for the generation of $C_{3,}$ consider the Bit Array Marix again. Calculate the support count for 3 item sets by using the following property

Sup_count(I_{123}) = 1x(0.1.0) + 1x(1.1.0) + 1x(1.1.0) + 2x(1.0.1)+ 2x(0.1.1) +1x(1.1.1)+ 1x(1.1.1) = 2

Sup_count((I_{123}) =2 Sup_count((I_{125}) =2

I_{123} & I_{125} will be the collection of 3 itemsets. The frequent 4 itemsets does not exist.

Pseudo Code:
Algorithm : TR-BAM technique.
Input : Dataset D, support value**.**
Output : frequent item set

Step 1: Start
Step 2: Read D = {I_j, T*j*} //
I is items, T is Transaction
Id

//Transform D to Bit Array
Representation
Step 3:
For each transaction in D
if the transaction T_i includes
I_j then
$T_{ij}=1$
else
$T_{ij}=0$
end if
end for
end
Step 4:
Generate the set of frequent 1 temset
L_1
For each column Ij of Matrix
If sum Ij >=min_sup then
L1 = Ij
Else
delete Ij from matrix.
End if

Step 5:
Number of times transaction repeated in the database is represented by the count in the RC column

Step 6:
k:=2; //k represents the pass number//
while ($L_{k-1} \neq ø$) do
begin
for each k itemset compute sup_count

Support count for k itemsets (Ii,Ij…….,I_k

Sup_count(Ii,Ij,I_k)= $RC_1x(T_{1i}.T_{1j}$ $.T_{1k})+……+ RC_nx(T_{ni}.T_{nj}..T_{nk.})$

if sup_count >= min_sup then
L_k := All candidates in C_k with minimum support ;
end if
For each item in L_K
delete infrequent item column from BAM
rearrange BAM
end for
k := k + 1;

end for
end
Answer := U_k L_k
Step 7:
End.

C1		*L1*	
Itemsets	***Support***	***Itemsets***	***Support***
I1	6	I1	6
I2	7	I2	7
I3	6	I3	6
I4	2	I4	2
I5	2	I5	2

C2		*L2*	
Itemsets	***Support***	***Itemsets***	***Support***
I1,I2	4	I1,I2	4
I1,I3	4	I1,I3	4
I1,I4	1	I1,I5	2
I1,I5	2	I2,I3	4
I2,I3	4	I2,I4	2
I2,I4	2	I2,I5	2
I2,I5	2		
I3,I5	1		

C3		L3	
Itemsets	**Support**	**Itemsets**	**Support**
I1,I2,I3	2	I1,I2,I3	2
I1,I2,I5	2	I1,I2,I5	2

Figure 3 Generation of frequent item sets by using Bit Array Matrix

4. CONCLUSION

The Bit Array Matrix discovers the patterns faster and performs very efficiently. It is easy to implement and it can give very good results especially in large databases. Apriori algorithm suffers from two limitations of large number of candidate item sets generation and database is scanned too many times. The proposed method compresses the data, reduces the number of database scans, avoids the connection and pruning process, greatly reduces the time complexity and space complexity of the algorithm, and also improves the execution efficiency of the algorithm. Finally, an illustrated example was given to clearly demonstrate and understand steps of the algorithm.

5. REFERENCES

[1]Agrawal, R., Imielinski, T., and Swami, A. N. Mining

Association Rules Between Sets of Items in Large Databases. Proceedings of the ACM SIGMOD, International Conference on Management of Data, pp.207- 216,

[2] Agrawal. R., and Srikant. R., Fast Algorithms for Mining Association Rules, Proceedings of 20th International Conference of Very Large Data Bases. pp.487-499,1994.

[3]M. S. Chen, J. Han, and P. S. Yu. Data mining: An overview from a database Perspective. IEEE Trans. Knowledge and Data Engineering, 8:866-883, 1996.

[4]U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy. Advances in Knowledge Discovery and Data Mining. AAAI/MIT Press, 1996.

[5]Agarwal, R. Agarwal, C. and Prasad V., A tree projection algorithm for generation of frequent item sets. In J. Parallel and Distributed Computing, 2000

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

[7] 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

[8] 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

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

[10]Wanjun Yu, Xiachun Wang and et.al, (2008), "The Research of Improved Apriori Algorithm for Mining Association Rules", pp. 513-516

[11] PEI Guying. A Fast Algorithm for Mining of Association Rules Based on Boolean Matrix. *Automation & Instrumentation*. 2009; 5: 16-18.

[12]LV Taoxia, LIU Peiyu. Algorithm for Generating Strong Association Rules Based on Matrix. *Application Research of Computers*. 2011; 28(4): 1301- 1303

[13] ZHANG Zhongping, LI Yan, YANG Jing. Frequent Itemsets Mining

Algorithm Based on Matrix. *Computer Engineering*. 2009; 35(1): 84-85.

[14]S.Prakash, R.M.S.Parvathi., An Enhanced Scaling Apriori for Association Rule Mining Efficiency. European Journal of Scientific Research, ISSN 1450-216X Vol.39 No.2 (2010), pp.257-264

[15] Wang Lifeng. An Efficient Association Rule Algorithm Based on Boolean Matrix. *International Review on Computers and Software*. 2012; 7(2): 695-700.