A New Efficient Matrix Based Frequent Itemset Mining Algorithm with Tags

Harpreet Singh and Renu Dhir

***Abstract*—The main aim of this paper is to present a new method based on transactional matrix and transaction reduction for finding frequent itemsets more efficiently. The association rule mining is based mainly on discovering frequent itemsets. Apriori algorithm is the most classical algorithm in association rule mining, but it has two fatal deficiencies: generation of a large number of candidate itemsets and scanning the database too many times. Apriori and other popular association rule mining algorithms mainly generate a large number of candidate 2-itemsets. To remove these deficiencies, a new method named Matrix Based Algorithm with Tags (MBAT) is proposed in this paper which finds the frequent itemsets directly from the transactional matrix which is generated from the database to generate association rules. Proposed algorithm greatly reduces the number of candidate itemsets, mainly candidate 2-itemsets**

***Index Terms*—Apriori algorithm, Association rule, Frequent itemsets, Transactional matrix, Transaction reduction**

I. Introduction

A set of items that appear frequently together in a transaction data set is a frequent itemset. Frequent itemset mining [1] leads to the discovery of associations and correlations among items in the large transactional or relational data sets. The problem of mining association rules can be reduced to that of mining frequent itemsets [1]. An association rule [1] specifies the interesting relationship between different data elements of the database or data sets. An association rule is of the form:

$X \rightarrow Y$ [Support= s%, Confidence=c%] where

1) Support s, is the probability that rule contains $\{X, Y\}$

$$\text{Support } (X \rightarrow Y) = P \text{ (XUY)},$$

2) Confidence c, is the conditional probability that specify the c% of the transactions of database considered must specify $X \rightarrow Y$

$$\text{Confidence}(X \rightarrow Y) = P(Y/X) = P \text{ (XUY)/support\_count}(X)$$

Minimum Support and Minimum Confidence are needed to eliminate the unimportant association rules. The association rule holds IFF it has the support and confidence value greater than or equal to minimum support and minimum confidence threshold value.

APRIORI algorithm [2] has been proposed by R. Agrawal and R. Srikant is one of the classical algorithms for finding frequent itemsets and then generating association rules from these itemsets. However, Apriori algorithm has the limitation of producing a large number of candidate itemsets and scanning the database too many times. Many researchers have given different approaches for improving the performance of Apriori algorithm. Changsheng Zhang and Jing Ruan [3] have worked on the improvement of Apriori algorithm by applying dataset reduction method and by reducing the I/O spending. Changsheng and Jing Ruan have applied the modified algorithm for instituting cross selling strategies of the retail industry and to improve the sales performance. Wanjun Yu, Xiaochun Wang and *et.al* [4] have proposed a novel algorithm called as Reduced Apriori Algorithm with Tag (RAAT), which improves the performance of Apriori algorithm by reducing the number of frequent itemset generated in pruning operation, by applying transaction tag method. Dongme Sun, Sheohue Teng and et.al [5] have presented a new technique based on forward and reverse scan of database. It produces the frequent itemsets more efficiently if applied with certain satisfying conditions. Sixue Bai, Xinxi Dai [6] have presented a method called P-matrix algorithm to generate the frequent itemsets. It is found that the P-Matrix algorithm is more efficient and fast algorithm than Apriori algorithm to generate frequent itemsets. Zhi Lin, Guoming Sang, Mingyu Lu [7] proposed a vector operation based method for finding association rules. The proposed algorithm finds the association rule more efficiently and requires only one database scan to find all the frequent itemsets.

In this paper, a new method based on transactional matrix is presented to find the frequent itemsets from a large transactional database. In this approach a transactional matrix is generated directly from the database and then frequent itemsets and support of each frequent itemset is generated directly from the transactional matrix. It is found that the new proposed approach finds the frequent itemsets more efficiently. The performance of new method is compared with that of Apriori algorithm with the help of an example.

II. Description of the Classical Apriori Algorithm

Apriori algorithm employs an iterative approach known as level-wise search, where k-itemsets are used to explore (k+1)-itemsets. First, the set of frequent 1-itemsets $L1$ is found. Next, $L1$ is used find the set of frequent 2-itemsets $L2$. Then $L2$ is used to find the set of frequent 3-itemsets $L3$.The method iterates like this till no more frequent k-itemsets are found.

Apriori algorithm finds the frequent itemsets from candidate itemsets. It is executed in two steps: first, it

Manuscript received September 9, 2012; revised October 12, 2012.

The authors are with the Department of Computer Science and Engineering, National Institute of Technology, Jalandhar, India (e-mail: harpreet99.nitj@gmail.com; dhirr@nitj.ac.in).

DOI: 10.7763/IJFCC.2013.V2.184

retrieves all the frequent itemsets from the database by considering those itemsets whose support is not smaller than the minimum support (min_sup). Secondly, it generates the association rules satisfying the minimum confidence (min_conf) from the frequent itemsets generated in first step. The first step consists of join and pruning action.While joining, the candidate set *Ck* is produced by joining *Lk*-1 with itself and pruning the candidate sets by applying the Apriori property i.e. all the non-empty subsets of a frequent itemset must also be frequent.

The pseudo code for generation of frequent itemsets is given below.

```
Ck: Candidate itemset of size k
Lk: Frequent itemset of size k
{
L1= frequent 1-itemset
For (k=1; k! =NULL; k++)
{
Ck+1=Join Lk with Lk to generate Ck+1;
Lk+1= Candidate in Ck+1 with support greater than
or equal to min support;
}
End;
Return Lk;
}
```

A transactional database is presented below in Table I to specify the process of Apriori Algorithm. Let minimum support value be 2 (min_sup=2) and minimum confidence (min_conf) be 50% (min_conf=50%).

The process of generating frequent itemsets by Apriori algorithm is shown below in Fig. 1.

TABLE I: AN EXAMPLE DATABASE

Transaction ID	Itemsets
T1	I1,I2,I3,I5
T2	I2,I4
T3	I2,I3
T4	I1,I4,I6
T5	I1,I3
T6	I3,I6
T7	I1,I3
T8	I1,I2,I3,I5
T9	I1,I2,I3
T10	I4,I5

In Apriori algorithm it is found that in the join step, *Lk*+1 is produced from *Ck*+1, which is produced by joining of *Lk* with itself. So it may produce large number of candidate itemsets. For example if there are 10^4 frequent 1-itemsets Apriori algorithm may produce 10^7 candidate2-itemsets while performing join operation [1]. So with large database the number of candidate itemsets generated by Apriori algorithm is too large and it scans the database too many times.

III. PROPOSED MBAT ALGORITHM

In this method the frequent itemsets are generated directly from the matrix which is generated from the transactional database. The matrix formed in this method is called as Transactional Matrix *T* shown below in Fig. 2 where *T*1, *T*2, *T*3, *T*i, *Tn* represent the various transactions; *I*1, *I*2, *I*3, *Ik*, *Im* represent the various items occurring in the transactional database. The entries y_{11}, y_{12}, $y13$… and so on represent either 1 or 0. If a transaction contains the item mentioned in the column then the value of that item in the corresponding row is written as 1 otherwise it is 0. Tag 1 and Tag 2 are columns representing smallest item serial number and largest item serial number respectively in the corresponding transaction.

C1		L1	
Itemsets	Support	Itemsets	Support
I1	6	I1	6
I2	5	I2	5
I3	7	I3	7
I4	3	I4	3
I5	3	I5	3
I6	2	I6	2

C2				L2	
Itemsets	Support	Itemsets	Support	Itemsets	Support
I1,I2	3	I3,I6	1	I1,I2	3
I1,I3	5	I4,I5	1	I1,I3	5
I1,I4	1	I4,I6	1	I1,I5	2
I1,I5	2	I5,I6	0	I2,I3	4
I1,I6	1			I2,I5	2
I2,I3	4			I3,I5	2
I2,I4	1				
I2,I5	2				
I2,I6	0				
I3,I4	0				
I3,I5	2				

C3		L3	
Itemsets	Support	Itemsets	Support
I1,I2,I3	3	I1,I2,I3	3
I1,I2,I5	2	I1,I2,I5	2
I1,I3,I5	2	I1,I3,I5	2
I2,I3,I5	2	I2,I3,I5	2

C4		L4	
Itemsets	Support	Itemsets	Support
I1,I2,I3,I5	2	I1,I2,I3,I5	2

Fig. 1. Generation of frequent itemsets by applying apriori algorithm

	I_1	I_2	I_3	I_k	.....	I_m	Tag1	Tag2
T1	Y_{11}	Y_{12}	Y_{13}	Y_{1k}	.....	Y_{1m}	T_{11}	T_{12}
T2	Y_{21}	Y_{22}	Y_{23}	Y_{2k}	.....	Y_{2m}	T_{21}	T_{22}
....	....	....	...	....	.....	....	....	.....
Ti	Y_{i1}	Y_{i2}	Y_{i3}	Y_{ik}	.....	Y_{im}	T_{i1}	T_{i2}
....	...	....	....	....	.....	...	....	....
Tn	Y_{n1}	Y_{n2}	Y_{n3}	Y_{nk}	.....	Y_{nm}	T_{n1}	T_{n2}

Fig. 2. Generation of Transactional matrix *T*

The proposed algorithm uses two properties:

1) Transaction reduction-It can be done by two ways described below:
   - A transaction that does not contain any frequent *k*-itemsets cannot contain any frequent (*k*+1)-itemsets, Therefore, such a transaction can be marked or removed from further consideration because subsequent scans of the transactional matrix will not require it.
   - A transaction which contains only *k* items cannot be useful in finding (k+1)-itemsets, Therefore, such a transaction can be marked or removed from the matrix.
2) All the non empty subsets of a frequent itemset must also be frequent.
   - So, there is no need to consider those frequent itemsets whose subsets are not frequent.
3) Two tag columns named Tag1 and Tag2 are used in the matrix where Tag1 represents smallest item serial number and Tag2 represents largest item serial number in the corresponding transaction.

The steps of the proposed algorithm are as follows:

4) First scan the database to find the different items occurring in the database and then make the transactional matrix by writing all the transactions along the row side and all the items occurring in the database along the column side.
5) Now complete the Transactional matrix, if the transaction contains the item mentioned in the column then write 1 otherwise 0 in the row corresponding to that transaction. Write the smallest item serial number under the column Tag 1 and largest item serial number under the column Tag 2 for the corresponding transaction.
6) The candidate set $C1$ is generated directly from the transactional matrix and the support count value is counted by counting the occurrence of particular item along the column in the transactional matrix.
7) For generation of $L1$, use $C1$. Move all those transactions from $C1$ to $L1$ whose support count value is not less than minimum support. After the generation of $L1$ use the transaction reduction properties mentioned above to remove rows from the matrix.
8) Now for the generation of $C2$, consider the transactional matrix again. Scan each row of the transactional matrix in such a way so as to generate 2-itemsets by considering the combinations of two items out of those items of the row which have value of 1. Write all those 2-itemsets in the candidate itemset table $C2$. Then find the support count of each 2-itemset generated by counting along the columns of the transactional matrix. Before scanning the columns of the every row for counting, first, Tag 1 is checked to see if the smallest item serial number of the itemset is less than the value of Tag 1 in the corresponding row. If the smallest item serial number is less than the value of the Tag 1 then there is no need to scan columns of that row for counting. If the smallest item serial number is not less than the value of the Tag 1 then Tag 2 is checked to see if the largest item serial number is greater than Tag 2. If the largest item serial number is greater than Tag 2 then we move to the next row for counting. Hence during the counting of support for candidate itemsets the tags help to reduce counting effort. Then move only those itemsets from $C2$ to $L2$ whose support count value is not less than min_support. After the generation of $L2$, use the transaction reduction properties mentioned above to remove the rows from the matrix.
9) Similarly generate $L3$, L4….and so on till all the rows of the matrix are removed.

NOTE 1: While generating itemsets, it is also considered not to count the support of those itemsets which are having non frequent subsets and exclude them from candidate set straight way.

IV. Comparison of the Proposed Algorithm

Comparison between the Apriori algorithm and new proposed method is presented with the help of an example database assumed above. The formation of the transactional matrix T and generation of frequent itemsets by the proposed method is shown below in Fig. 3 to 7.

	I1	I2	I3	I4	I5	I6	Tag1	Tag2
T1: I1 I2 I3 I5	1	1	1	0	1	0	1	5
T2 : I2 I4	0	1	0	1	0	0	2	4
T3: I2 I3	0	1	1	0	0	0	2	3
T4: I1 I4 I6	1	0	0	1	0	1	1	6
T5: I1 I3	1	0	1	0	0	0	1	3
T6: I3 I6	0	0	1	0	0	1	3	6
T7: I1 I3	1	0	1	0	0	0	1	3
T8: I1 I2 I3 I5	1	1	1	0	1	0	1	5
T9: I1 I2 I3	1	1	1	0	0	0	1	3
T10: I4 I5	0	0	0	1	1	0	4	5

Fig. 3. Generation of transactional matrix T

	I1	I2	I3	I4	I5	I6	Tag1	Tag2
T1: I1 I2 I3 I5	1	1	1	0	1	0	1	5
T8: I1 I2 I3 I5	1	1	1	0	1	0	1	5
T9: I1 I2 I3	1	1	1	0	0	0	1	3

Fig. 4. Transactional matrix T after generation of frequent 2-itemsets

	I1	I2	I3	I4	I5	I6	Tag1	Tag2
T1: I1 I2 I3 I5	1	1	1	0	1	0	1	5
T8: I1 I2 I3 I5	1	1	1	0	1	0	1	5

Fig. 5. transactional matrix T after generation of frequent 3-itemsets

	I1	I2	I3	I4	I5	I6	Tag1	Tag2

Fig. 6. Transactional matrix T after generation of frequent 4-itemsets

Items combination	C1		L1	
Itemsets	Itemsets	Support	Itemsets	Support
I1	I1	6	I1	6
I2	I2	5	I2	5
I3	I3	7	I3	7
I4	I4	3	I4	3
I5	I5	3	I5	3
I6	I6	2	I6	2
Items combination	**C2**		**L2**	
Itemsets	Itemsets	Support	Itemsets	Support
I1,I2	I1,I2	3	I1,I2	3
I3,I6	I3,I6	1	I1,I3	5
I1,I3	I1,I3	5	I1,I5	2
I4,I5	I4,I5	1	I2,I3	4
I1,I4	I1,I4	1	I2,I5	2
I4,I6	I4,I6	1	I3,I5	2
I1,I5	I1,I5	2		
I1,I6	I1,I6	1		
I2,I3	I2,I3	4		
I2,I4	I2,I4	1		
I2,I5	I2,I5	2		
I3,I5	I3,I5	2		
Items combination	**C3**		**L3**	
Itemsets	Itemsets	Support	Itemsets	Support
I1,I2,I3	I1,I2,I3	3	I1,I2,I3	3
I1,I2,I5	I1,I2,I5	2	I1,I2,I5	2
I1,I3,I5	I1,I3,I5	2	I1,I3,I5	2
I2,I3,I5	I2,I3,I5	2	I2,I3,I5	2
Items combination	**C4**		**L4**	
Itemsets	Itemsets	Support	Itemsets	Support
I1,I2,I3,I5	I1,I2,I3,I5	2	I1,I2,I3,I5	2

Fig. 7. Generation of frequent itemsets by applying new method

In this method first transactional matrix T is generated as shown above in Fig. 3.The generation of frequent itemsets is shown in Fig. 7. For $C1$ the items combinations {I1}, {I2}, {I3}, {I4}, {I5} and {I6} are considered and then their respective support count is counted by using transactional matrix T. It is found that all the items in $C1$ have the support count more than minimum support. Therefore, move all the items of $C1$ to $L1$. For generation of $L2$, scan every row of the transactional matrix and consider all the 2-itemsets combinations of the elements which have value 1 in the rows. Then count the support for each itemset and move only those itemsets from *C*2 to *L*2 whose support is not less than minimum support.

Therefore, itemsets {I2, I4}, {I1, I4}, {I1, I6}, {I4, I6}, {I3, I6}, {I4, I5} are not moved to L2.Now let us take the case of itemset {I2, I4}. For support count of this itemset the counting is started from the first row by counting along column numbers 2 and 4. When the 6^{th} row is reached, its Tag1 is checked and it is found that the lowest serial item number i.e. 2 is less than the value of in Tag1 column in the corresponding row and hence the counting is not performed in that row and next row's columns are scanned for counting. Similar would be the case at 10^{th} row. After generation of *L*2, we use the transaction reduction properties. *T*4 and T10 do not contain any frequent 2-itemset therefore *T*4, T10 rows are removed using the first way of transaction reduction and *T*2, *T*3, *T*5, *T*6, *T*7 rows are deleted using second way of transaction reduction because these transactions contain only two items and hence these will not be useful in the calculation of frequent 3-itemsets.After the use of transaction reduction property transactional matrix is reduced to only three rows as shown in the Fig. 4. Next, consider all the 3-itemsets combinations of the items having value 1 in the rows. Then count the support of these itemsets by using transactional matrix *T*. Tags are checked continuously to check if scanning of columns during counting of support can be avoided. After generation of *L*3 the transaction reduction property is used and the matrix is reduced to two rows shown in Fig. 5. Similarly frequent 4-itemsets are generated and after using property of transaction reduction all the rows of the matrix are removed as shown in Fig. 6 and the algorithm is stopped. Hence all the frequent itemsets are generated.

Following table shows the general comparison of classical apriori algorithm and the new proposed method.

TABLE II: COMPARISON OF APRIORI AND NEW METHOD

Method	Number of times database scanned	Number of candidate itemsets generated	Computational time
Apriori algorithm	Large	Large	Large
Proposed method	Only once	Very less as compared to apriori algorithm	Very less

Advantages of the proposed MBAT algorithm over the classical Apriori algorithm are:

1) In this algorithm the database is scanned only once and that is only to generate the transactional matrix.
2) This method greatly reduces the problem of generation of a large number of candidate itemsets because this method considers only those items in the row of the matrix which are having the value of 1.
3) The tag columns are very helpful in reducing the effort in counting support for itemsets.
4) The combination of above properties and the transaction reduction property provides another advantage of less computational time.
5) Method is much easier to implement than apriori and other popular algorithms for association rule mining.

V. CONCLUSION

The frequent itemset mining is the process of finding out frequent itemsets from a large existing database. Apriori algorithm suffers from two limitations of large number of candidate itemsets generation and database is scanned too many times. The proposed new method based on transactional matrix and transaction reduction provided in this paper solves both these problems of Apriori algorithm. It helps in mining frequent itemsets efficiently. In this method once the transactional matrix is generated it is easy to generate the frequent itemsets directly from the transactional matrix.

REFERENCES

[1] J. Han and M. Kamber, *Data mining Concepts and Techniques*, Morgan Kaufman Academic Press, 2001

[2] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules," in *Proc. of Int. Conf. Very Large Data Bases (VLDB'94)*, pp. 487–499, Santiago, Chile, Sept. 1994

[3] C. Zhang and J. Raun, "A Modified Apriori Algorithm with its application in Instituting Cross-Selling strategies of the Retail Industry," in *Proc. of 2009 International Conference on Electronic Commerce and Business Intelligence*, pp. 515-518

[4] W. Yu, X. Wang *et al*., "The Research of Improved Apriori Algorithm for Mining Association Rules," in *Proc. of 11th IEEE International Conference on Communication Technology Proceedings*, pp. 513-516

[5] D. Sun *et al*., "An algorithm to improve the effectiveness of Apriori Algorithm," in *Proc. of 6th ICE Int. Conf. on Cognitive Informatics*, 2007, pp. 385-390.

[6] S. Bai and X. Dai, "An efficiency Apriori algorithm: P_matrix algorithm," *First International Symposium on Data, Privacy and E-Commerce*, pp.101-103, 2007.

[7] Z. Liu, G. Sang, and M. Lu, "A Vector Operation Based Fast Association Rules Mining Algorithm," in *Proc. of Int. Joint Conf. on Bioinformatics, System Biology and Intelligent Computing*, pp. 561-564, 2009.

Renu Dhir has completed her B.Sc. Engineering Electrical from Panjab University in 1983 with Honours (1st division). She has completed her M. Tech. in Computer Science and Engineering from TIET Patiala in 1997, with 8.72 CGPA on 10 point scale. She has completed her Doctor of Philosophy (Ph.D.) in Computer Science and Engineering from Punjabi University, Patiala in March 2008 in the area of character recognotion. Her research area is Pattern Recognition, Image Processing, Information security and Software Engineering. She is presently working at Dr. B R Ambedkar National Institute of Technology Jalandhar (Punjab) - 144011 in the Department of Computer Science & Engineering as Associate Professor. She is teaching courses to both UG and PG students since more than 25 years. She has more than 20 publications in various International Journals and more than 40 publications in various International and National conferences.