Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 47 (2015) 52 – 61

Procedia Computer Science

An Efficient Count Based Transaction Reduction Approach For Mining Frequent Patterns

V.Vijayalakshmi[a], Dr.A.Pethalakshmi[b]

[a]Research Scholar, M.S University,Tirunelveli, T.N

[b]Head & 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 generating frequent item sets. This classical algorithm is inefficient due to so many scans of database. And if the database is large, it will take too much time to scan the database. To overcome these limitations, researchers have made a lot of improvements to the Apriori. This paper analyses the classical algorithm as well as some disadvantages of the improved Apriori and also proposed two new transaction reduction techniques for mining frequent patterns in large databases. In this approach, 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. It also adopts a new count-based transaction reduction and support count method for candidates. Appropriate operations are designed and performed on matrices to achieve efficiency. All the algorithms are executed in 5% to 25% support level and the results are compared. Efficiency is proved through performance analysis.

Peer-review under responsibility of organizing committee of the Graph Algorithms, High Performance Implementations and Applications (ICGHIA2014)

Keywords: Association Rule, Frequent Item Set, Support Count, Apriori, Transaction Reduction Technique

1. Introduction

With the progress of the technology of information and the need for extracting useful information of business people from dataset, data mining and its techniques is appeared to achieve the above goal. Data mining is the essential process of discovering hidden and interesting patterns from massive amount of data where data is stored in data warehouse, OLAP (on line analytical process), databases and other repositories of information.

* Corresponding author. Tel.: 9788170037 ; fax: +0-000-000-0000

E-mail address: v.viji08@yahoo.in

This data may reach to more than terabytes. Data mining is also called (KDD) knowledge discovery in databases, and it includes an integration of techniques from many disciplines such as statistics, neural networks, database technology, machine learning and information retrieval, etc [6].

Peer-review under responsibility of organizing committee of the Graph Algorithms, High Performance Implementations and Applications (ICGHIA2014)

doi:10.1016/j.procs.2015.03.183

Interesting patterns are extracted at reasonable time by KDD's techniques. KDD process has several steps, which are performed to extract patterns to user, such as data cleaning, data selection, data transformation, data pre-processing, data mining and pattern evaluation.

The architecture of data mining system has the following main components [6]: data warehouse, database or other repositories of information, a server that fetches the relevant data from repositories based on the user's request, knowledge base is used as guide of search according to defined constraint, data mining engine include set of essential modules, such as characterization, classification, clustering, association, regression and analysis of evolution. Pattern evaluation module that interacts with the modules of data mining to strive towards interested patterns. Finally, graphical user interfaces from through it the user can communicate with the data mining system and allow the user to interact.

Association Mining is one of the most important data mining's functionalities and it is the most popular technique has been studied by researchers. Extracting association rules is the core of data mining. It is mining for association rules in database of sales transactions between items which is important field of the research in dataset. The benefits of these rules are detecting unknown relationships, producing results which can perform basis for decision making and prediction.

In this regard, the first algorithm Apriori was proposed by Agarwal and Srikanth in the year 1994 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. The normally followed scheme for mining association rules consists of two stages [1]:

1. the discovery of frequent itemsets
2. the generation of association rules.

As the second step is rather straightforward and as the first step dominates the processing time, we explicitly focus the paper on the first step: the discovery of frequent itemsets.

The remaining part of this paper is organized as follows: Section 2 contains related works. In section 3, elaborates the proposed transaction reduction techniques. Section 4 discusses about the performance analysis of proposed algorithm compared with Apriori algorithm. Section 5 contains conclusion.

2. Related Works

Agrawal proposed an algorithm, called AIS algorithm [1], for generating frequent itemsets. In the AIS algorithm, frequent itemsets are generated through iterations on scanning the database. The iteration terminates when no new frequent item-set is derived. After reading a transaction in the k^{th} iteration, the AIS algorithm computes the candidate k − itemsets by first deriving a set of (k−1) −itemsets which contains itemsets that are both in the frequent (k−1) −itemsets and in the transaction. 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.

Ayres.J [2] introduced an effective pruning mechanism called depth first strategy to mine the sequential pattern in large database,. This strategy defines the database in vertical bitmap format with effective support counting. For each item in the dataset, a vertical bitmap is constructed by which each data set transaction is represented as a bit. The value for items is set based on the item present in the transaction. The efficient support counting and candidate generation is obtained by partitioning the bitmap.

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.

One of the novel algorithm presented by Chen.J [4] is a BISC (Binary Item set Support Counting), which is

responsible for efficient mining of frequent item set. According to this algorithm, support of all item sets in a database is derived with respect to direct supports. The context BISC transforms the database transaction into binary format. The memory consumption and the cost of support updating is minimized by integrating the two related techniques namely, one stage BISC (BISC1) and two stage BISC (BISC2). This technique is both time and space efficient because of BISC in conjunction with projection techniques that reduce the branching factor of database projection and maximum depth.

Dongme Sun, Sheohue Teng [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.

Hanbing Liu [7] presented a new association rule algorithm called ABBM (Association Rule Mining Based on Boolean Matrix algorithm) which transforms a transaction database into a Boolean matrix. It scanned the transaction database once, it does not produce candidate itemsets, and it adopted the Boolean vector "relational calculus" to discover frequent itemsets. In addition, it stores all transaction data in bits, so it needs less memory space and can be applied to mining large databases.

Jaisree Singh [8] developed a Transaction Reduction Algorithm which reduced 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. But it has overhead to manage the new database after every generation of L_k. So, there should be some approach which has very less number of scans of database.

Kavitha.K [9] proposed an efficient transaction reduction technique named TR-BC to mine the frequent pattern based on bitmap and class labels. The proposed approach reduces the rule generation by counting the item support and class support instead of only item support. Moreover, the database storage is compressed by using bitmap that significantly reduces the number of database scan. The rules are reduced by horizontal and vertical transaction and then finally combined rules are generated by eliminating the redundancy.

Ramaraj.E [11] proposed a novel frequency itemsets generation algorithm called TRApriori that maintained its performance even at relative low supports. The advantages of TRApriori include interactive mining with different supports; faster execution time and infrequently used item are not stored and hence improves the size of the query data.
Sixue Bai and Xinxi Dai [12] 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.

Wanjun Yu, Xiaochun Wang [13] 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.

Zhi Lin, Guoming Sang, Mingyu Lu [14] 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 a matrix is presented to find the frequent itemsets. In this approach, a bit array matrix is generated directly from the database. The frequent itemsets and support of each frequent itemset is generated directly from the 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.

3. Proposed Algorithm

Matrix Strategy

We generate a Bit Array Matrix which contains only 0 and 1 (where 0 and 1 represents the presence and absence of item respectively in transaction database). In the matrix, each row represents a transaction T_i, each column represents one item I_j, indicated the items occurred in transaction. The process of generating the matrix is as follows:

Let T= [T_1, T2 ,….., Tm] is the set of transactions and I=[I_1, I_2,….,I_n] be the set of items.

$$M = (Tij)mxn = \begin{cases} Tij = 1, \text{if } Ij \in Ti \\ Tij = 0, \text{if } Ij \text{ NOT } \in Ti \\ \text{where } i = 1,2,3,..,m; \quad j = 1,2,\dots,n \end{cases} \tag{1}$$

New Transaction Reduction Strategy

Property 1. Each value of RC column stores the corresponding number of similar rows. If the transaction doesn't repeat then repetition column for the transaction is set to 1.

$$RCi = \begin{cases} RCi + 1 \ , if\ Tsj = Ttj \quad (s \# t), \\ 1 \quad , \quad otherwise \end{cases} \quad \begin{matrix} where\ j = 1,2,\dots,n \\ i,s,t = 1,2,\dots,m \end{matrix} \tag{2}$$

Property 2. We use an attribute Size-of-Transaction (COUNT), containing number of nonzero elements in individual row of the Bit Array Matrix. The corresponding item set is extracted directly without moving transactions scanning. [8][9][10].

$$count = \sum_{j=1}^{n} Tij \qquad where\ i = 1,2,\dots,m\ \&\ Tij > 0 \tag{3}$$

New Support Count Strategy

Property 3. Support count of one itemset is sum of nonzero elements of each column.

$$sum = \sum_{i=1}^{m} Tij \qquad where\ j = 1,2,\dots,n;\ Tij > 0 \tag{4}$$

Property 4. Support count of k itemsets can be got by using count in the RC column and bitwise "&"operation. The "& operation" is to AND the rows according to the items in the matrix, then add the result of the "&" and the result is the support count.

$$\text{sup\_}count\ of\ k\ itemsets = \sum_{i=1}^{n} \sum_{j=1}^{k} (RCi\ x\ (\&\ Tij)) \tag{5}$$

3.1 Proposed Method 1: TR-RC (Transaction Reduction Based on Repetition Count)

In this approach, the number of transactions to be scanned is greatly reduced when comparing to the original Apriori algorithm by reducing the number of similar transactions in the database and this results in reduction of time. The process is started from a given transactional database as shown in Table 1.

Table 1 Transactional Database

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

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 along the row side and all the items occurring in the database along the column side. Don't repeat the transaction in the transactional matrix. In the Bit Array Matrix, if the transaction T_i includes I_j, then $T_{ij} = 1$, otherwise $T_{ij} = 0$. In the matrix, each row represents a transaction T_i each column represents one item I_j. 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. The Bit Array Matrix is as follows

Table 2

	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 by using property 3. 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. Move all those transactions from C_1 to L_1 whose sum value is not less than min_support (min_{sup}=2).

3. Each value of RC column stores the corresponding number of similar rows. If the transaction doesn't repeat then repetition column for the transaction is set to 1 by using property 1.

Table *3* Bit Array Matrix with RC

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

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

$Sup\_count(I_1,I_2)=4$ $Sup\_count(I_1,I_3)=4$ $Sup\_count(I_1,I_4)=1$ $Sup\_count(I_1,I_5)=2$
$Sup\_count(I_2,I_3)=4$ $Sup\_count(I_2,I_4)=2$ $Sup\_count(I_2,I_5)=2$ $Sup\_count(I_3,I_5)=2$

Then move only those item sets from C_2 to L_2 whose support count value is not less than minimum support. $\{I_1,I_2\}$, $\{I_1, I_3\}$, $\{I_1, I_5\}$, $\{I_2, I_3\}$, $\{I_2, I_4\}$, $\{I_2, I_5\}$, $\{I_3, I_5\}$ will be frequent 2 itemsets.

5. Next, Consider all the 3-itemsets combinations of the items. The various combinations possible are $\{I_1, I_2, I_5\}$, $\{I_1, I_2, I_4\}$, $\{I_1, I_2, I_3\}$, $\{I_2, I_3, I_5\}$, $\{I_1, I_3, I_5\}$. Now, by the property 4, L_3 will contain $\{I_1, I_2, I3\}$ and $\{I_1, I_2, I_5\}$.

6. Now for the generation of C_3, consider the Bit Array Marix again. Calculate the support count of 3 item sets.

$Sup\_count\{I_1, I_2, I_3\}=2$ $Sup\_count\{I_1, I_2, I_5\}=2$

$\{I_1, I_2, I_3\}$ & $\{I_1, I_2, I_5\}$ will be the collection of 3 itemsets. The frequent 4 itemsets does not exist.

Algorithm 1 - TR-RC for FIM

Min_sup. : Minimum support count
Step 1: Begin
Step 2: Read BAM
Step 3:
Generate the set of frequent 1 itemset
Add RC column //
k:=2;
while ($L_{k-1} \neq ø$) do
begin
for each k itemset
compute sup_count
if sup_count >= min_{sup} then
L_k := All candidates in C_k with minimum support ;
end if
end for
k := k + 1;
end Answer := $U_k L_k$
Step 4: End.

3.2 Proposed Method 2: CBTRA (Count Based on Repetition Count & Size of Transaction)

In this approach, we have combined the property 1 and 2. The major advantage of this approach is that, the number of transactions to be scanned is greatly reduced, by reducing the number of similar rows as well as by cutting down the unnecessary transaction rows. So, the corresponding item set is extracted directly without moving for entire database.

The steps of proposed algorithm are as follows:

1. The transaction database (Table 1) is transformed into the Bit Array Matrix.

2. Add COUNT and RC column on the right side of the matrix. The proposed methodology exploits horizontal transaction of the data set that automatically reduces the entire database scanning. Table 4 represents the horizontal transaction for the given data set.

Table 4 Horizontal Transaction for the given data set

	I1	I2	I3	I4	I5	COUNT	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		

3. Set the minimum support count as min_ sup=2, move all those transactions from C_1 to L_1 whose sum value is not less than min_support (min_sup=2). So, the set of frequent 1-itemset is: L_1= {{I_1}, {I_2}, {I3}, {I_4}, {I_5}} by using property 3.

4. By using the property 1 and 2, set RC and COUNT column of a matrix respectively.

Table 5 BAM after Generation of One Itemsets

I1	I2	I3	I4	I5	COUNT	RC
1	1	0	0	1	3	1
0	1	0	1	0	2	1
0	1	1	0	0	2	2
1	1	0	1	0	3	1
1	0	1	0	0	2	2
1	1	1	0	1	4	1
1	1	1	0	0	3	1

5. For generation of L_2, scan every row of the above matrix and consider the transactions which have the COUNT value greater than one by using property 2. Therefore {{I_1,I_2}, {I_1,I_3}{I_1,I_5}, {I_2,I_3}, {I_2,I_4}, {I_2,I_5}, {I_3,I_5}} will be frequent 2 itemsets.

6. Next, for generation of 3-itemsets combinations from above matrix, consider the transactions which have the COUNT value greater than two. So, L_3 will contain {I_1, I_2, I_3} and {I_1, I_2, I_5}.

7. Similarly, 4-itemsets possible combinations are considered. i.e. {I_1, I_2, I_3, I_5}. The support count of this itemset is less than min_sup. Therefore C_4=NULL and L_4= NULL. Hence all the frequent itemsets are generated.

Algorithm 2 – CBTRA for FIM

Min_sup. : Minimum support count
Step 1:
Begin
Step 2: Read BAM
Step 3: Generate the set of frequent 1 temset
Add RC & COUNT column .
k:=2;
while ($L_{k-1} \neq \emptyset$) do
begin
calculate the sup_count for each k itemset
for each k itemset
if COUNT is greater than or equal to k.
compute support count for k itemsets (Ii,Ij…….,I_k)
if sup_count >= min_{sup} then
L_k := All candidates in C_k with min_sup
end if
end if
end for
k := k + 1;
end
Answer := U_k L_k
Step4:
End.

4. Experimental Results

In order to appraise the performance of the proposed algorithms, we conducted an experiment using the Apriori algorithm, TR-RC and the CBTRA algorithm.

4.1. Experiment 1

For this purpose, we select the supermarket data to study the object. The supermarket database contains 958 transactions, apply algorithms on same number of transaction and compare the execution time with support count 5, 10,15,20,25 shown in Fig. 1.

Fig. 1 shows performance of Apriori with TR-RC and CBTRA respectively, here proposed algorithm outperforms.

Table 6 The Time Reducing Rate of TR-RC on the Apriori of according to the value of minimum support; The average of reducing time rate in the TR-RC is 66.47%

MIN_SUP	APRIORI(S)	TR-RC(S)	TRR(%)
5	0.063	0.016	74.60
10	0.062	0.016	74.19
15	0.047	0.016	65.95
20	0.047	0.016	65.95
25	0.031	0.015	51.61

Table 7 The Time Reducing Rate of CBTRA on the original Apriori according to the value of minimum support; The average of reducing time rate in the CBTRA is 67.312%.

MIN_SUP	APRIORI(S)	CBTRA(S)	TRR(%)
5	0.063	0.016	74.60
10	0.062	0.016	74.19
15	0.047	0.015	68.08
20	0.047	0.015	68.08
25	0.031	0.015	51.61

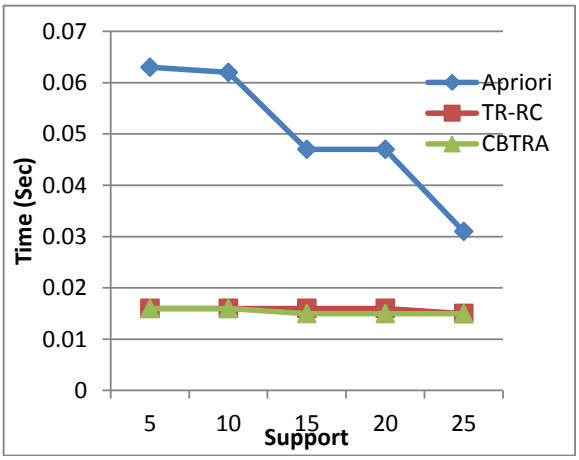

Fig. 1. Time consuming comparison for different values of minimum support

As we observe in Fig. 1, that the time consuming in proposed approach in each value of minimum support is less than it in the original Apriori, and the difference increases more and more as the value of minimum support decreases.

4.2 Experiment 2

The second experiment compares the time consumed of original Apriori, and our proposed algorithm by applying the three groups of transactions in the implementation. The result is shown in Fig. 2.

- T1 : 999 Transactions
- T2 : 1200 Transactions
- T3 : 1750 Transactions

Table 8 The Time Reducing Rate of TR-RC on the Apriori minimum support ; The average of reducing time rate in the TR-RC is 63.75%

MIN_SUP	APRIORI(S)	TR-RC(S)	TRR(%)
999	0.078	0.032	58.97
200	0.079	0.031	60.75
1750	0.109	0.031	71.55

Table 9 The Time Reducing Rate of CBTRA on the original Apriori according to the value of minimum support, The average of reducing time rate in the CBTRA is 75.24%.

MIN_SUP	APRIORI(S)	CBTRA(S)	TRR(%)
999	0.078	0.031	60.25
1200	0.079	0.016	79.74
1750	0.109	0.015	86.23

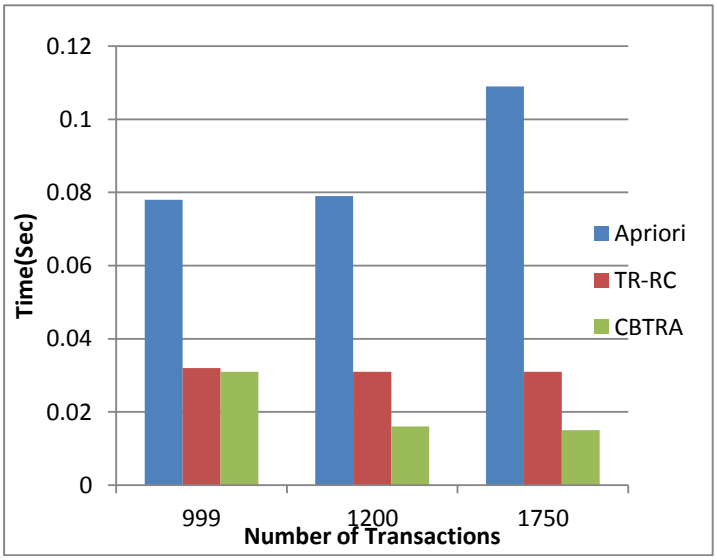

Fig. 2. Time consuming comparison for different groups of transactions

As we observe in Fig. 2, that the time consuming in proposed approach in each group of transactions is less than it in the original Apriori, and the difference increases more and more as the number of transactions increases.

All experiments are performed on Intel core i3, 3.07GHz processor and 2GB of RAM, the algorithms were implemented in Java and tested on a Windows XP platform.

5. Conclusion

In this research paper, a new transaction reduction algorithm is proposed through reducing the time consumed by reducing the number of coequal transactions to be scanned. Whenever the value of minimum support increases, the gap between our proposed and original Apriori algorithm decreases in view of time consumed. The time consumed to generate frequent item set in our proposed algorithm is less than the original Apriori; our algorithms reduces consuming time by 63.75% and 75.24%, as this is proved and validated by the experiments and observed in Fig. 1, Fig. 2, Table 8 and Table 9.

References

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

[2] Ayres.J, Flannick.J, Gehrke.J, and Yiu.T, "Sequential pattern mining using a bitmap representation," in Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining, 2002, pp. 429-435.

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

[4] Chen.J and Xiao.K, "BISC: A bitmap itemset support counting approach for efficient frequent itemset mining," ACM Transactions on Knowledge Discovery from Data (TKDD), 2010. vol. 4, p. 12.

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

[6] Han.J, Kamber.M, "Data Mining: Concepts and Techniques", Morgan Kaufmann Publishers, Book, 2000.

[7] Hanbing Liu and Baisheng Wang "An Association Rule Mining Algorithm Based on a Boolean Matrix", Data Science Journal, 2010, Vol(6), supplement 9.

[8] Jaisree singh, Hari Ram, Dr.J.S.Sodhi "Improving efficiency of Apriori Algorithm using Transaction Reduction" IJSRP, Vol(3), 2013. ISSN 2250-3153.

[9] Kavitha.K, Dr.E.Ramaraj,"Efficient Transaction Reduction in Actionable Pattern Mining for High Voluminous Datasets based on Bitmap and Class Labels", IJCSE, Vol.5 No.07 Jul 2013. ISSN: 0975-3397.

[10] Logeswari.T, Valarmathi.N, Sangeetha.A, Masilamani.M, "Analysis of Traditional and Enhanced Apriori Algorithm in Association Rule Mining", International Journal of Computer Applications. 2014, vol(87).

[11] Ramaraj.E, K.RameshKumar, N.Venkatesan, "A Better Performed Transaction Reduction Algorithm for Mining Frequent Itemsets from Large Voluminuous Database", Proceeding of the 2nd National Conference, Computing for Nation Development, February 08-09, 2008.

[12] Sixue Bai, Xinxi Dai, "An efficiency Apriori algorithm: P_matrix algorithm," First International Symposium on Data, Privacy and ECommerce, 2007, pp.101-103.

[13] Wanjun Yu, Xiaochun Wang., "The Research of Improved Apriori Algorithm for Mining Association Rules," in Proc. of 11th IEEE International Conference on Communication Technology Proceedings, 2004, pp. 513-516.

[14] Zhi Lin, Guoming Sang, Mingyu Lu "A Vector Operation Based Fast Association Rules Mining Algorithm," in Proc. of Int. Joint Conf. on Bioinformatics, System Biology and Intelligent Computing, 2009, pp. 561- 564.

‘