ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 1, January 2017

Survey on Improvement of Apriori Algorithm

Dixita Tandel[#1], Prof. Neha Soni[#2]

M.E. Student, Dept. of Computer Engineering, Sardar Vallabhbhai Patel Institute of Technology, Vasad, Gujarat, India

Assistant Professor, Dept. of Computer Engineering, Sardar Vallabhbhai Patel Institute of Technology, Vasad, Gujarat, India

ABSTRACT: Data mining is the process of analyzing the data from large database and summarizing that data into useful information. Association rule mining is one of the most important data mining's functionalities for finding frequent itemset or association between items or attribute. It can be consider as two step process, first it find frequent items and from that it generate association rule. Apriori algorithm is the most popular algorithm which is used to extract frequent itemsets from large data sets where these frequent itemsets can be used to generate association rules. These rules are used for discovering knowledge such as detecting unknown relationships, decision making, prediction, etc. It finds frequent itemset based on join and prune step. It's simple but having some disadvantages such as generate more candidate itemset, multiple scanning of database, etc. Large number of variations has been proposed in the literature to overcome the shortcomings of Apriori algorithm. So this paper presents a survey on Apriori algorithm of recent research work carried by different researchers with example.

KEYWORDS: Data mining, association rule, apriori, support, frequent itemset.

I. INTRODUCTION

Data mining is the process of examining large pre-existing databases in order to generate new information. Data mining also known as knowledge discovery is used to discover useful patterns from the huge database. Many techniques have been developed in data mining amongst which association rule mining is important for finding frequent itemset. Apriori is one of the best algorithms for the association rule mining. The Apriori algorithm generate frequent patterns from database whose support must satisfy the minimum support criteria where these frequent itemsets are used to generate association rule whose confidence must satisfy the minimum confidence criteria.

The main contribution of our work is to survey in the direction of optimized apriori algorithm. In all previous survey papers they simply give description about some improved algorithm. In our survey paper we are given summary of algorithm theory as well as an example. All examples are calculated using same sample transaction database. So, from that example we can easily understand the differences in different algorithm or difference in frequent itemset generation process. We can also get the idea about which algorithm produce more candidate itemsets than other.

II. RELATED WORK

This section includes literatures carried out on Improvement of apriori algorithm. Following are the improvement proposed over Apriori algorithm:

In [17] authors improved this algorithm, Pick a random sample S of the given data D, and then search for frequent itemsets in S instead of D. So, uses a lower support threshold than minimum support to find the frequent itemsets local to S. In [13] authors present the DHP algorithm which is hash-based. The candidate generated by this algorithm is much smaller than previous methods, especially generated candidate 2-itemsets. And the generation of smaller candidate sets enables us to effectively trim the transaction database size at a much earlier stage of the iterations. In [15] authors presented the partition-based algorithm, which scans the database only twice, reducing the I/O operation observably and thus improved the efficiency of the algorithm. Large capacity databases will be logically divided into several disjoint blocks which are used to generate locally frequent item sets, and then these local frequent item sets to get the final global frequent items sets through testing their support. In [8] authors presented filtration approach instead

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 1, January 2017

of pruning, in which frequent and infrequent itemsets are generated. They also improved the filtration approach [9] by checking only (k-1) infrequent itemset as filters. In [5] Authors proposed Improved DC_Apriori algorithm, which restructured the storage structure of the database, improved connection of frequent item sets, improved join step, and greatly reduced the number of connections. In order to improve the efficiency of Apriori authors proposed, a novel algorithm, named BitApriori [6], for mining frequent itemsets. In BitApriori, the data structure binary string is used to describe the database. The support count can be implemented by performing the Bitwise "And" operation on the binary strings. Another technique presented in this paper is a special equal-support pruning. The Enhanced BitApriori algorithm [20] is same like a BitApriori algorithm, but the main difference is that in BitApriori used the Bitwise "And" operation on binary strings and in Enhanced BitApriori Bitwise "XOR" operation is used on binary strings. Ordering is not based on decreasing support count [20]. In [2] authors used adjacency matrix as optimized data structure. This algorithm reduces the size of candidate-K itemset in successive iteration. Pruning is also done at two stages which reduces the memory space. The algorithm works as follow: **(**Frequent item set generation:) First select max item from sorted table as frequent Itemset name given as A. Items of adjacency matrix which are associated with A are consider as candidate itemsets. This procedure is continuing until no more itemsets are generated. In [7] the data mapping technique is optimized, so the database is scanned only once. After generating frequent 1-itemset following frequent item sets can be obtained by scanning C_k and L_k. Authors give the name of the algorithm as OApriori [7]. Algorithm [4] is based on transaction reduction in which the attribute SOT (size of transaction) is used. As compared to apriori algorithm authors [18] has improved join and pruned step. Join step: For two k-1 frequent itemset named l_x and l_y, if these two itemset can't be connected, then we can say that all the itemsets after l_x and l_y can't satisfy the connection condition. As a result, there is no need to determine whether the itemsets which are behind the itemset l_x and l_y satisfy the connection conditions or not. Prune step: The necessary condition for a k-dimensional becomes a frequent itemset is that all the (k-1) dimensional subsets are also frequent itemset. Let k-dimensional itemset {i1 i2,….,i_k }, if $|L_{k-1}(j)| <$k-1 is true then delete j from L_{k-1}. Here $|L_{k-1}(j)|$ means the number of j in$L_{k-1}(j)$. In [14] Authors give the name of the algorithm as APRIORI-IMPROVE algorithm which generates L_2 directly from one scan over the database without generating C_1, $L_1$and C_2. And it replaces the hash tree by a hash table to reduce searching cost. The L_2 is directly generated using a hash function. Hash function: $H(x,y) = (P(x) - 1) * (N - 1) + P(y) - P(x)$**,** Where x and y are the items in database, N is the number of transactions and P denotes the position. In [19] authors present an improved algorithm which is based on the classic Apriori algorithm, in which Pruned optimization strategy adapted to optimize the process of pruning in order to reduce the generation of frequent itemsets, and take Transaction Reduction strategy to reduce the size of the transaction database to be scanned.

III. METHODOLOGY

Apriori Algorithm is a traditional algorithm for frequent item generation, which is proposed by R.Agrawal and R.Srikant in 1994 [1].

The general block diagram of Apriori Algorithm is shown below:

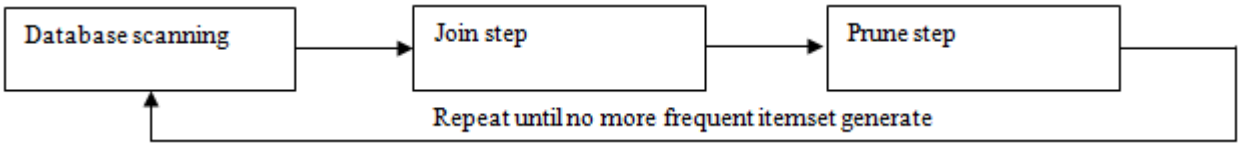


Fig.1 Block diagram of Apriori Algorithm

The algorithm simply scans all of the transactions in order to count the number of occurrences of each itemset. The item which satisfies the minimum support criteria is taken as frequent k-itemset which is known as L_k.

Join step: To find L_k a set of candidate k-itemsets is generated by joining L_{k-1} with itself. This set of candidates is denoted C_k.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 1, January 2017

Prune step: To reduce the size of C_k, the Apriori property is used as follows: if any (k-1)-subset of a candidate k-itemset is not in L_{k-1}, then the candidate cannot be frequent either, so it can be removed from C_k.

IV. RESULT

This section includes the example of all improved algorithms using sample transaction database (Table.1).
Table 1 is the sample transaction database which we used to solved the all Algorithms.

TID	List of item IDs
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

Table.1 Sample transaction database [21]

Fig. 2 is example of Apriori Algorithm which is solved (Table.1) based on Algorithm [1].

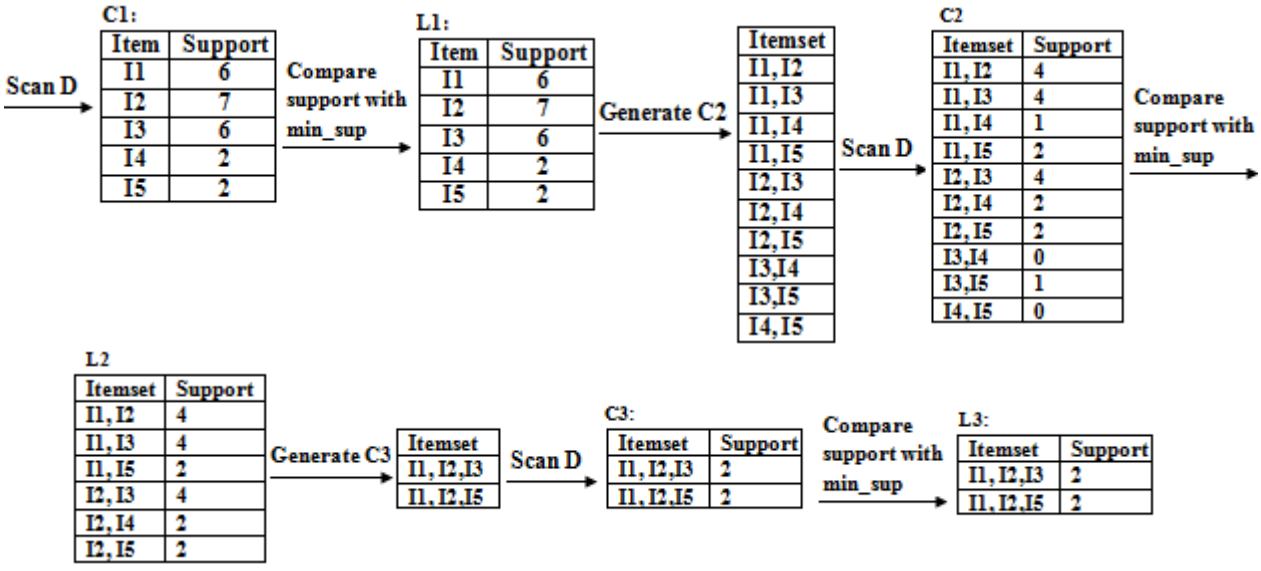


Fig.2 Example of Apriori Algorithm

Fig. 3 is example of research and improvement of apriori algorithm which based on Algorithm given in paper [5]. In this algorithm improved Join step is: The generation of k-frequent itemsets is only need to join the 1-frequent item sets with frequent k-1-itemsets (L_{k-1}). For that it needs to compare last element of frequent k-1-itemset (l) with L1 items. If $l<l_1$, then join the itemset of L_{k-1} with L_1. Based on that frequent itemset generated is as Fig. 3:

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 1, January 2017

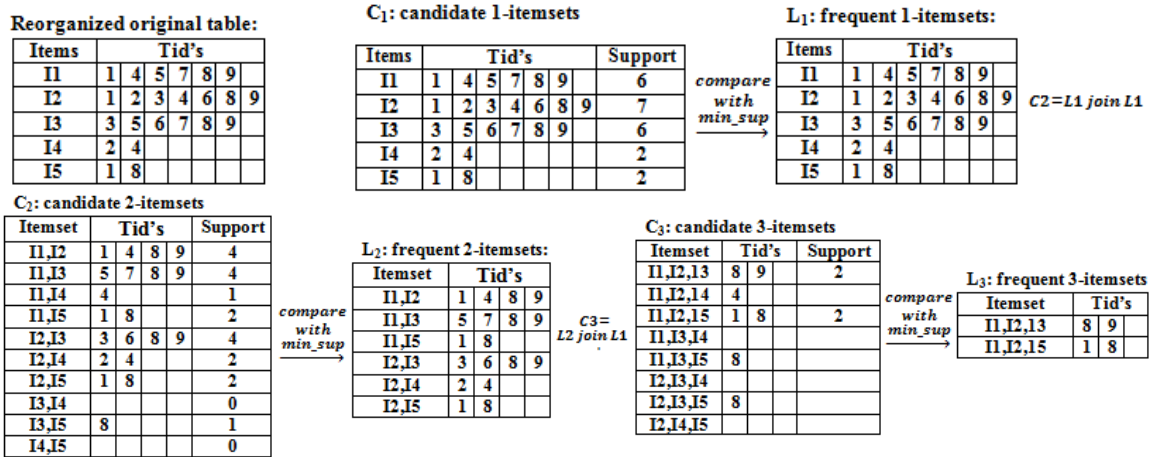


Fig.3 Example of research and improvement of apriori algorithm

Fig. 4 is example of BitApriori based on Algorithm given in paper [6].

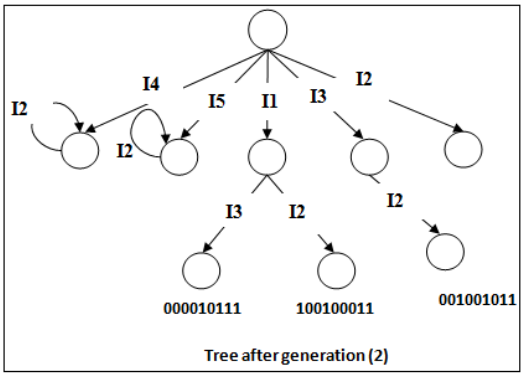
First scan of database we get
I1=6, I2=7, I3=6, I4=2, I5=2.
Compare with min_sup,

Frequent items are:

Items	I4	I5	I1	I3	I2
Support	2	2	6	6	7

Frequent 2-itemset:

Tid	Items	Ordered	{I5,I1}	{I5,I2}	{I1,I2}	{I4,I2}	{I3, I2}	{I1, I3}
1	I1, I2, I5	I5,I1, I2	1	1	1	0	0	0
2	I2, I4	I4,I2	0	0	0	1	0	0
3	I2, I3	I3,I2	0	0	0	0	1	0
4	I1, I2, I4	I4,I1,I2	0	0	1	1	0	0
5	I1, I3	I1,I3	0	0	0	0	0	1
6	I2, I3	I3,I2	0	0	0	0	1	0
7	I1, I3	I1,I3	0	0	0	0	0	1
8	I1, I2,I3,I5	I5,I1,I3,I2	1	1	1	0	1	1
9	I1, I2, I3	I1,I3,I2	0	0	1	0	1	1
Support			2	2	4	2	4	4



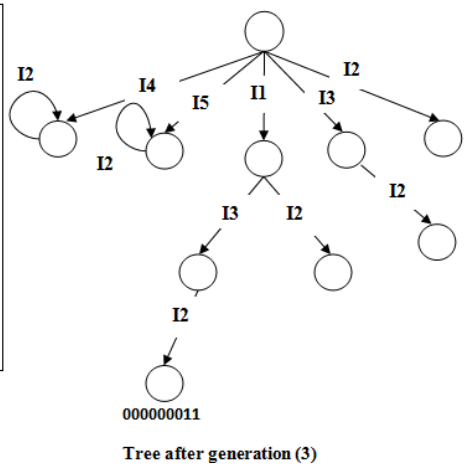


Fig. 4 Example of BitApriori

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 1, January 2017

Fig. 5 is example of Enhanced BitApriori Algorithm based on Algorithm given in paper [20].

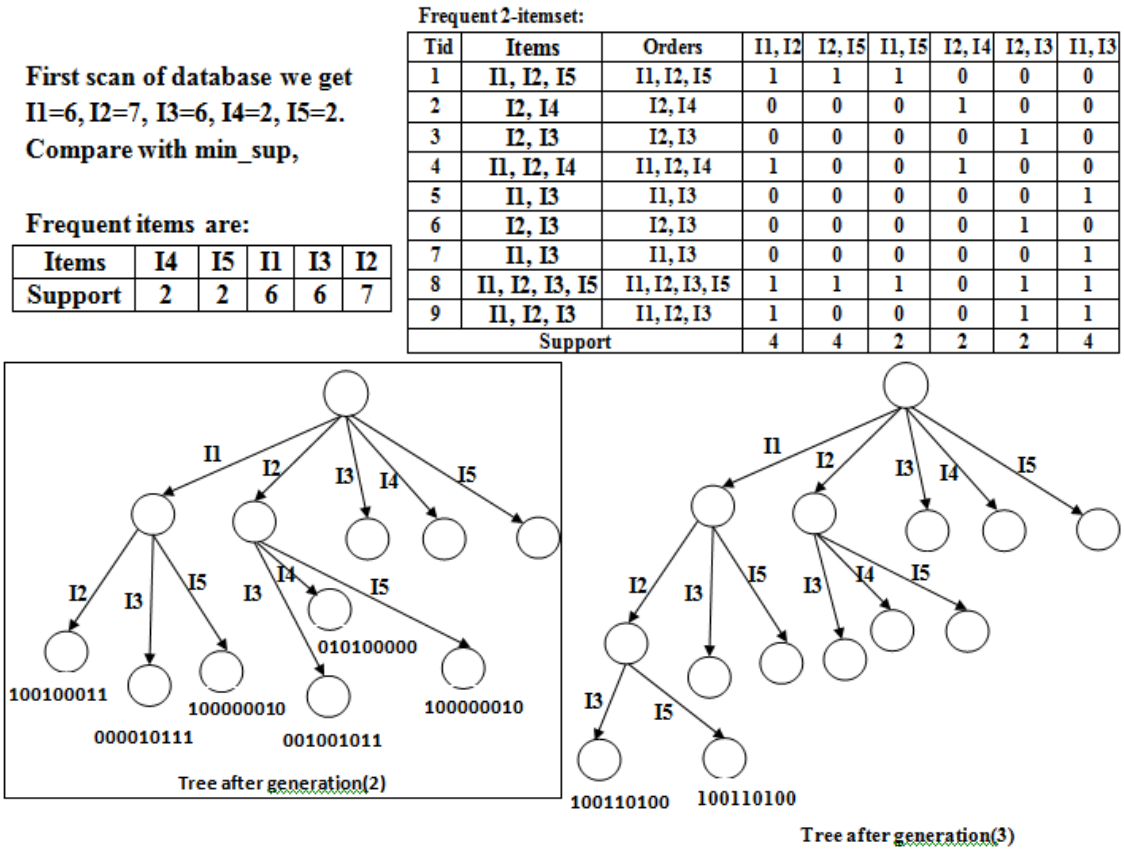


Fig. 5 Example of Enhanced BitApriori Algorithm

Fig. 6 is example of Proposed algorithm for frequent itemset generation using Algorithm given in paper [2].

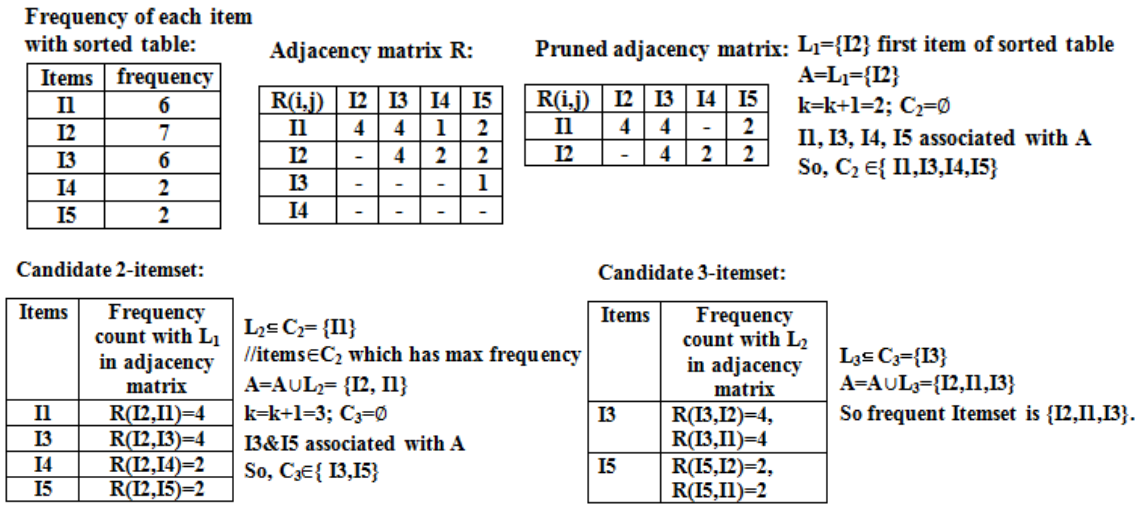


Fig. 6 Example of Proposed algorithm for frequent itemset generation

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

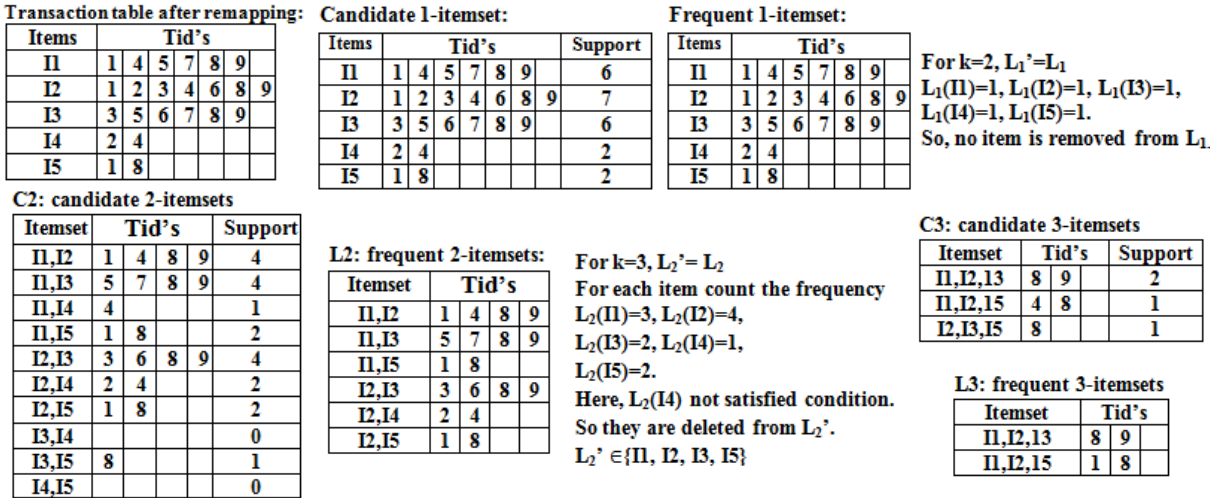
International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 1, January 2017

Fig. 7 is example of OApriori Algorithm based on Algorithm given in paper [7].



Transaction table after remapping:

Items	Tid's
I1	1, 4, 5, 7, 8, 9
I2	1, 2, 3, 4, 6, 8, 9
I3	3, 5, 6, 7, 8, 9
I4	2, 4
I5	1, 8

Candidate 1-itemset:

Items	Tid's	Support
I1	1, 4, 5, 7, 8, 9	6
I2	1, 2, 3, 4, 6, 8, 9	7
I3	3, 5, 6, 7, 8, 9	6
I4	2, 4	2
I5	1, 8	2

Frequent 1-itemset:

Items	Tid's
I1	1, 4, 5, 7, 8, 9
I2	1, 2, 3, 4, 6, 8, 9
I3	3, 5, 6, 7, 8, 9
I4	2, 4
I5	1, 8

For k=2, L_1'=L_1
L_1(I1)=1, L_1(I2)=1, L_1(I3)=1,
L_1(I4)=1, L_1(I5)=1.
So, no item is removed from L_1.

C2: candidate 2-itemsets

Itemset	Tid's	Support
I1,I2	1, 4, 8, 9	4
I1,I3	5, 7, 8, 9	4
I1,I4	4	1
I1,I5	1, 8	2
I2,I3	3, 6, 8, 9	4
I2,I4	2, 4	2
I2,I5	1, 8	2
I3,I4		0
I3,I5	8	1
I4,I5		0

L2: frequent 2-itemsets:

Itemset	Tid's
I1,I2	1, 4, 8, 9
I1,I3	5, 7, 8, 9
I1,I5	1, 8
I2,I3	3, 6, 8, 9
I2,I4	2, 4
I2,I5	1, 8

For k=3, L_2'= L_2
For each item count the frequency
L_2(I1)=3, L_2(I2)=4,
L_2(I3)=2, L_2(I4)=1,
L_2(I5)=2.
Here, L_2(I4) not satisfied condition.
So they are deleted from L_2'.
L_2' ∈{I1, I2, I3, I5}

C3: candidate 3-itemsets

Itemset	Tid's	Support
I1,I2,I3	8, 9	2
I1,I2,I5	4, 8	1
I2,I3,I5	8	1

L3: frequent 3-itemsets

Itemset	Tid's
I1,I2,I3	8, 9
I1,I2,I5	1, 8

Fig. 7 Example of OApriori

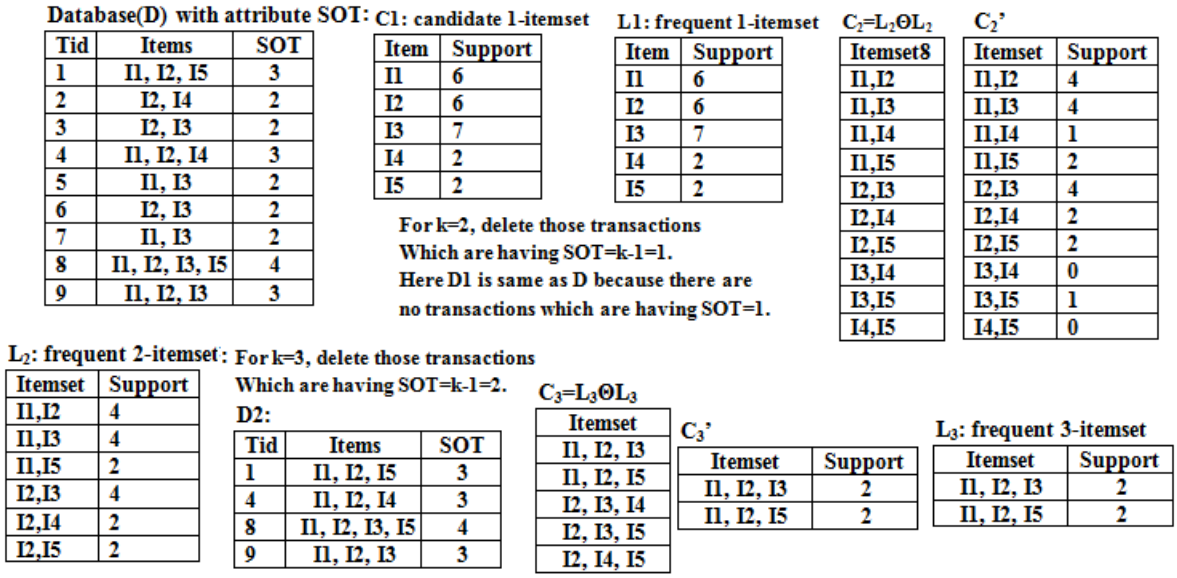
Fig. 8 is example of Improving Efficiency of Apriori Algorithm Using Transaction Reduction based on Algorithm given in paper [4] which is solved based on following steps:

Step 1: First convert database into the desired database that is with SOT column. In the first iteration of the algorithm, each item is a member of the set of candidate 1-itemsets, C_1. The algorithm simply scans all of the transactions in order to count the number of occurrences of each item. This algorithm will then generate number of items in each transaction (SOT). Based on the min_sup the set of frequent 1-itemset, L_1 can be determined.

Step 2: The deletion process of transaction in database will made according to the value of k. If value of k is match with value of SOT then delete those transactions from database.

Step 3: L_k is generated by join and prune operation.

Step 4: repeat step 2 and step 3 until C_{k+1}=Ø.



Database(D) with attribute SOT:

Tid	Items	SOT
1	I1, I2, I5	3
2	I2, I4	2
3	I2, I3	2
4	I1, I2, I4	3
5	I1, I3	2
6	I2, I3	2
7	I1, I3	2
8	I1, I2, I3, I5	4
9	I1, I2, I3	3

C1: candidate 1-itemset

Item	Support
I1	6
I2	6
I3	7
I4	2
I5	2

L1: frequent 1-itemset

Item	Support
I1	6
I2	6
I3	7
I4	2
I5	2

For k=2, delete those transactions
Which are having SOT=k-1=1.
Here D1 is same as D because there are
no transactions which are having SOT=1.

C_2=L_2ΘL_2

Itemset8
I1,I2
I1,I3
I1,I4
I1,I5
I2,I3
I2,I4
I2,I5
I3,I4
I3,I5
I4,I5

C_2'

Itemset	Support
I1,I2	4
I1,I3	4
I1,I4	1
I1,I5	2
I2,I3	4
I2,I4	2
I2,I5	2
I3,I4	0
I3,I5	1
I4,I5	0

L_2: frequent 2-itemset:

Itemset	Support
I1,I2	4
I1,I3	4
I1,I5	2
I2,I3	4
I2,I4	2
I2,I5	2

For k=3, delete those transactions
Which are having SOT=k-1=2.

D2:

Tid	Items	SOT
1	I1, I2, I5	3
4	I1, I2, I4	3
8	I1, I2, I3, I5	4
9	I1, I2, I3	3

C_3=L_3ΘL_3

Itemset
I1, I2, I3
I1, I2, I5
I2, I3, I4
I2, I3, I5
I2, I4, I5

C_3'

Itemset	Support
I1, I2, I3	2
I1, I2, I5	2

L_3: frequent 3-itemset

Itemset	Support
I1, I2, I3	2
I1, I2, I5	2

Fig. 8 Example of Improving Efficiency of Apriori Algorithm Using Transaction Reduction

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

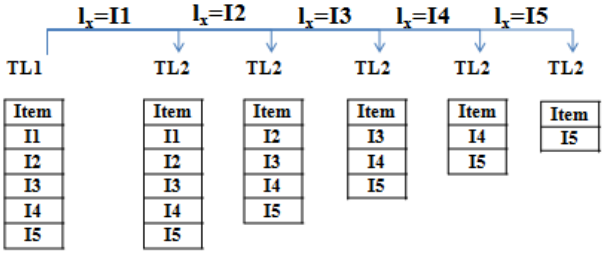
International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 1, January 2017

Fig. 9 is example of Example of Mining Association Rules Based on an Improved Apriori Algorithm based on Algorithm given in paper [18].



Here the algorithm is same as 5th literature but it require less comparison time when joining (see in Fig. 9 for join procedure)

Fig. 9 Example of Mining Association Rules Based on an Improved Apriori Algorithm

Fig. 10 is example of APRIORI-IMPROVE algorithm based on Algorithm given in paper [14].

The first step is to read the database once and make a hash table which includes each itemset and its support. Before going to next step we will discard the items which does not satisfies the minimum support criteria. The next step is to make combination of 2-itemsets for each transaction (i.e.) if transaction 1 contains items like I1,I2,I3 then possible combination would be {I1,I2} {I1,I3} {I2,I3}. Now by using the Hash function we store the support of each combination. We get L2 after pruning the data we can get the 2 Itemset. Similarly for the further candidate set generation we use the same procedure. From the hash structure table we get L_2 (after pruning the above table data we can get the 2 Item set). Similarly for the further candidate set generation we use the same procedure.

C_1:

Item	Support	P(X)
I1	6	1
I2	7	2
I3	6	3
I4	2	4
I5	2	5

Hash table:

TID	Item
T1	{I1, I2}, {I1,I5}, {I2,I5}
T2	{I2, I4}
T3	{I2, I3}
T4	{I1, I2},{I1,I4}, {I2,I4}
T5	{I1, I3}
T6	{I2, I3}
T7	{I1, I3}
T8	{I1, I2}, {I1,I3}, {I1,I5}, {I2,I3}, {I2,I5}, {I3,I5}
T9	{I1, I2}, {I1,I3},{I2,I3}

Hash Function: H(x, y) = (P(x)-1)*(N-1)+P(y)-P(x).

{I1, I2}	{I1, I3}	{I1, I4}	{I1, I5}	{I2, I3}	{I2, I4}	{I2, I5}	{I3, I4}	{I3, I5}	{I4, I5}
4	4	1	2	4	2	2	0	1	0

Fig. 10 APRIORI-IMPROVE algorithm

Fig. 11 is example of An Improved Apriori Algorithm Based on Pruning Optimization and Transaction Reduction based on Algorithm given in paper [19] which is solved based on following steps:

Step 1: Generate L_1 as same as classic Apriori.

Step 2: Generate k itemsets: if $k \neq 2$, count the frequency of $|L_{k-1}(a)|$ of every item exists in L_{k-1}. If ($|L_{k-1}(a)| < k-1$) then delete those itemsets from $L_{k-1}$and update L_{k-1}.

Step 3: $C_k = L_{k-1} \Theta L_{k-1}$. For each transaction count the length of transaction, delete transaction whose length is less than k.

Step 4: $L_k = \{c \in C_k \mid c.count \geq min\_sup\}$

Step 5: repeat step 2 to step 4 until $L_{k-1} \neq \emptyset$.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 1, January 2017

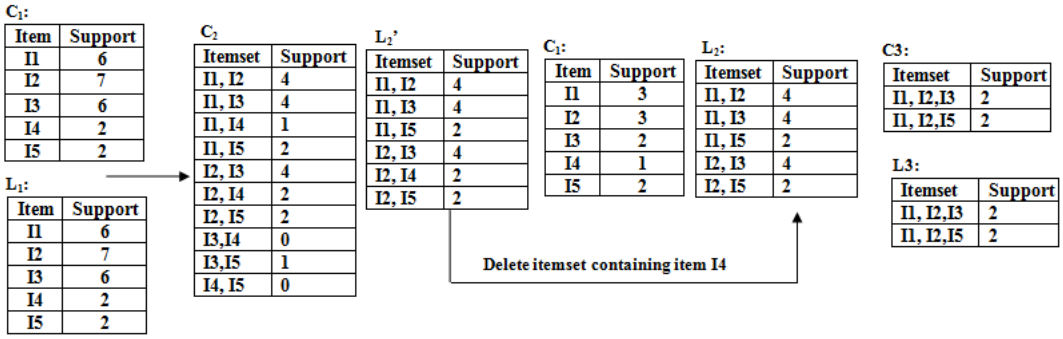


Fig.11 Example of An Improved Apriori Algorithm Based on Pruning Optimization and Transaction Reduction

V. COMPARISON

A table given below summarizes the methods, advantages and limitations of improved apriori algorithm.

Table.2 Comparative study of improvement of Apriori algorithms

Sr. No	Title	Method	Advantages	Disadvantages
1	Research and Improvement of Apriori Algorithm[5]	Database: Item-Tid form. Storage structure is based on dynamic array vector and map set. Join: L_{k-1} with L_1. Support: use intersection.	Avoid repeated comparison of (k-2) items. Avoid scanning of db. Only 1 traversal can get the intersection because Tid list is in ascending order.	More candidate sets are generated.
2	Enhanced BitApriori Algorithm: An Intelligent Approach for Mining Frequent Itemset[20]	BitApriori, Equal Support Pruning (ESP), Binary string, XOR operation	Need to scan the db twice. No candidate generated. Reducing the height of the tree because of ESP Easy to calculate the support count. All frequent itemset are generated.	When the database is large, may suffer the problem of memory scarcity. Traversing the tree recursively (because of no ordering).
3	Proposed algorithm for frequent item set generation[2]	Database: Adjacency matrix. Pruning: Sorted table and adjacency matrix are pruned. Candidate: consist itemset which have relation with L_{k-1}.	Reduced memory utilization Reduced number of item to be scanned	Not all frequent itemsets are generated.
4	A Method to Optimize Apriori Algorithm for Frequent Items Mining[7]	Database: Item-Tid form. Connection steps: delete the transaction from L_k which not satisfies the min_sup. Support: use intersection.	Scan database only once. Avoid comparison of checking (k-1) subset frequent or not.	
5	An Efficient Filtration Approach for Mining Association	Instead of pruning filtration approach is used in which frequent and infrequent itemsets are generated.	Optimum numbers of candidate k-frequent itemsets are generated.	Extra space is required to store the all infrequent k-itemset.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 1, January 2017

	Rules[8]			
6	Improved[9] Filtration step for Mining AssociationRules	Filtration steps: improved by checking only (k-1) infrequent itemset as filters.	Optimum k-candidate itemsets are generated. Reduces the extra effort in filtration process.	Extra space is required to store the all infrequent k-itemset.
7	Improving Efficiency of AprioriAlgorithm UsingTransaction Reduction[4]	Traction reduction strategy: number of items in each transaction <k delete those transaction.	Reduce the number of transaction scanning.	Manage new db after every generation of L_k.
8	An efficient algorithm for frequent itemsets in data mining[6]	BitApriori, Equal support pruning (ESP) Binary string Bitwise AND operator Sorting of frequent 1-itemset into non decreasing order of support	Need to scan the db twice. No candidate generation. Reducing the height of the tree using ESP. Easy to calculate the support count. All frequent itemset are generated.	When the database is large, the BitApriori may suffer the problem of memory scarcity. Not all frequent itemsets are generated.
9	Mining AssociationRules Based on an Improved Apriori Algorithm[18]	Pruning: $\|L_{k-1}(j)\| <k-1$ then it if not frequent. Database: Delete transaction of itemset which not contains the C_k during the calculation of C_k.	Reduce the number of candidate itemsets. Optimize the database so efficiency can be more and more with the growth of value k.	
10	An Improved Apriori Algorithm Based on Pruning Optimization and Transaction Reduction[19]	Prune step: temporary table used to store the frequency of items of L_{k-1}, if freq(item)<k-1, delete all the frequent itemsets containing the item in L_{k-1}. Transaction Reduction Strategy: delete transaction whose length is less than k+1.	Avoid a lot of scanning the database. Eliminate non-frequent candidate itemsets.	Extra space is required to temporary store table of each items frequency when generating k>2 itemset.
11	An Effective Hash-Based Algorithm for Mining Association Rules[13]	The proposed algorithm utilizes a hash method for candidate itemset generation during the initial iterations and employs pruning to reduce the transaction database size.	Effective reduction on transaction db size. Scans the full db for the first 2 passes and then scans D_k thereafter. Smaller transaction scan	The candidate itemsets generation and support count process is still the time consuming. Need extra time for hashing the transaction.
12	Sampling Large Database for Association Rule[17]	Pick a random sample S of the given data D. Search frequent itemsets in S instead of D. Uses a lower support threshold.	Scan only sample S, it saves time. Beneficial when efficiency is of utmost importance.	Sacrifices some accuracy. May lose some global frequent items sets.
13	An Efficient Algorithm for Mining Association Rule in Large Database[15]	Partition: Database will be logically divided into several disjoint blocks. Local frequent itemsets used to get final global frequent items sets for testing their support.	Requires only two pass of the db. Improves the performance of apriori algorithm as partitioned data sets can be executed parallel.	Some items may get missed due to partitioning.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 1, January 2017

VI. CONCLUSION

Association rule mining is one of the most important data mining techniques for finding association between items or attribute. Apriori is the classical and mostly used algorithm which is used to generate frequent itemsets from large data sets. Though it has been good algorithm, but have some disadvantage such as generate more candidate itemset, require multiple scanning of database. To overcome such issues number of algorithms has been proposed in the literature. After literature study it has been noted that the variations proposed may solve one or more issues of Apriori but results into some new problems or leaving the other issues untouched.

REFERENCES

1. Agrawal R and Srikant R., "Fast algorithms for mining association rules", In Proceedings of the international conference on very large data bases (VLDB'94), Santiago, Chile, 1994.
2. Archana Singh and Kyoti Agarwal, "Proposed algorithm for frequent item set generation", IEEE, 2014.
3. Feng WANG and Yong-hua LI, "An improved Apriori algorithm based on the matrix", IEEE, International Seminar on Future BioMedical Information Engineering, pp. 152-155, 2008.
4. Jaishree Singh, Hari Ram and Dr. J.S. Sodhi, "Improving Efficiency of Apriori Algorithm Using Transaction Reduction", International Journal of Scientific and Research Publications, Vol.3, Issue 1, pp.1-4, 2013.
5. Jiaoling Du, Xiangli Zhang, Hongmei Zhang and Lei Chen, "Research and Improvement of Apriori Algorithm", IEEE sixth International Conference on Science and Technology, pp.117-121, 2016.
6. Jiemin Zheng, Defu Zhang, Stephen C. H. Leung, and Xiyue Zhou, "An efficient algorithm for frequent itemsets in data mining", IEEE, 2010.
7. Ke Zhang, Jianhuan Liu, Yi Chai, Jiayi Zhou and Yi Li, "A Method to Optimize Apriori Algorithm for Frequent Items Mining", IEEE Seventh International Symposium on Computational Intelligence and Design, pp.71-75, 2014.
8. Lalit Mohan Goyal and M. M. Sufyan Beg, "An Efficient Filtration Approach for Mining Association Rules" IEEE, International Conference on Computing for Sustainable Global Development, pp.178-185, 2014.
9. Lalit Mohan Goyal and M. M. Sufyan Beg, "Improved Filtration Step for Mining Association Rules", IEEE, 2014.
10. Mannila H., Toivonen H., Verkamo A., "Efficient algorithm for discovering association rules", In: AAAI Workshop on Knowledge Discovery in Databases, pp.181-192, 1994.
11. Mingzhu Zhang and Chang zheng He, "Survey on Association Rules Mining Algorithms", Springer, Advancing Computing, Communi., Control and Management, LNEE 56, pp.111-118, 2010.
12. O.Jamsheela and Raju.G, "Frequent Itemset Mining Algorithms: A Literature Survey", IEEE International Advance Computing Conference (IACC), pp.1099-1104, 2015.
13. Park J.S., M.S. Chen and P.S. Yu, "An effective hash based algorithm for mining association rules", ACM SIGMOD, pp.175-186, 1995.
14. Rui Chang and Zhiyi Liu "An Improved Apriori Algorithm" IEEE, International Conference on Electronics and Optoelectronics (ICEOE), vol.1, pp.476-478, 2011.
15. Savasere A., E. Omiecinski, S.B. Navathe, "An efficient algorithm for mining association rules in large databases", in Proceedings of 21th International Conference on Very Large Data Bases (VLDB'95), 1995.
16. Surbhi K. Solanki and Jalpa T. Patel, "A Survey on Association Rule Mining", Fifth International Conference on Advanced Computing & Communication Technologies, pp.212-216, 2015.
17. Toivonen H., "Sampling large databases for association rules", In Proceedings of 22nd VLDB Conference, India, pp.1-12, 1996.
18. Yanfei Zhou, Wanggen Wan, Junwei Liu, Long Cai**,** "Mining Association Rules Based on an Improved Apriori Algorithm", IEEE, pp.414-418, 2010.
19. Zhuang Chen and Shibang Cai, Qiulin Song and Chonglai Zhu, "An Improved Apriori Algorithm Based on Pruning Optimization and Transaction Reduction" IEEE, pp.1908-1911, 2011.
20. Zubair Khan, and Faisal Haseen, "Enhanced BitApriori Algorithm: An Intelligent Approach for Mining Frequent Itemset", SPRINGER, Vol.1, pp.343-350, 2015.
21. Jiawei Han and Micheline Kamber, "Data Mining: Concepts and Techniques" 2nd Edn; Morgan Kaufmann Publishers, ELSEVEIR 2006.

BIOGRAPHY

Dixita Pankajbhai Tandel is a M.E. student in the Computer Engineering Department, SardarVallbhbhai Patel Institute of Technolgy, Vasad. Her research interest is Data Mining.