ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 5, Issue 2, March 2016

Implementation and Analysis of Improved Apriori Algorithm

Sangita Chaudhari, Mayur Borkhatariya, Apurva Churi, Mohini Bhonsle

Abstract*— Association rule is the main technique to determine the frequent item set in data mining. Apriori algorithm is a classical algorithm of association rule mining. This classical algorithm is inefficient due to multiple scans over the database. And if the database is large then it takes too much time to scan the whole database. In this paper we have proposed Improved Apriori algorithm which will help in reducing multiple scans over the database by cutting down unwanted transaction records as well as redundant generation of sub-items while pruning the candidate item sets. The performance of this algorithm is analyzed against the FP Growth algorithm in which there is no generation of candidate set.

***Index Terms*— - Data mining, Apriori Algorithm, FP-Growth Algorithm, Association Rule, Candidate item set, Itemset.**

I. INTRODUCTION

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 [6]. This data may reach to more than terabytes. Data mining is also called (KDD) knowledge discovery in databases [3], and it includes an integration of techniques from many disciplines such as statistics, neural networks, database technology, machine learning and information retrieval, etc [7]. Interesting patterns are extracted at reasonable time by KDD's techniques [2]. KDD process has several steps, which are performed to extract patterns to user, such as data cleaning, data selection, data transformation, data preprocessing, data mining and pattern evaluation [4].

Association rule are the statements that find the relationship between data in any database. Association rule has two parts "Antecedent" and "Consequent". For example {bread} => {butter}. Here bread is the antecedent and butter is the consequent. Antecedent is that item which is found in the database, and consequent is the item that is found in combination with the first i.e. the antecedent [5].Association rule is used to abstract the data by picking the frequently used data in retail store for marketing, inventory control, etc.

Apriori is an algorithm for frequent item set mining and association rule learning over transactional databases. It proceeds by identifying the frequent individual items in the database and extending them to larger and larger item sets as long as those item sets appear sufficiently often in the database. The frequent item sets determined by Apriori can be used to determine association rules which highlight general trends in the database: this has applications in domains such as market basket analysis.[8] Apriori algorithm is breadth first search algorithm which generate (k+1) candidate item set based on k-frequent items. In the improved Apriori algorithm, we are reducing the scans over the main database to remove the insufficiency drawback of the Apriori algorithm.

The FP-Growth Algorithm, proposed by Han in [9], is an efficient and scalable method for mining the complete set of frequent patterns by pattern fragment growth, using an extended prefix-tree structure for storing compressed and crucial information about frequent patterns named frequent-pattern tree (FP-tree). The FP-Growth Algorithm is an alternative way to find frequent item sets without using candidate generations, thus improving performance. For so much it uses a divide-and-conquer strategy. The core of this method is the usage of a special data structure named frequent-pattern tree (FP-tree), which retains the item set association information. In simple words, this algorithm works as follows: first it compresses the input database creating an FP-tree instance to represent frequent items. After this first step it divides the compressed database into a set of conditional databases, each one associated with one frequent pattern. Finally, each such database is mined separately. Using this strategy, the FP-Growth reduces the search costs looking for short patterns recursively and then concatenating them in the long frequent patterns, offering good selectivity. In large databases, it's not possible to hold the FP-tree in the main memory. A strategy to

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 5, Issue 2, March 2016

cope with this problem is to firstly partition the database into a set of smaller databases (called projected databases), and then construct an FP-tree from each of these smaller databases.

II. REVIEW OF LITERATURE

Data mining (the analysis step of the "Knowledge Discovery in Databases" process, or KDD), an interdisciplinary subfield of computer science, is the computational process of discovering patterns in large data sets ("big data") involving methods at the intersection of artificial intelligence, machine learning, statistics, and database systems. The overall goal of the data mining process is to extract information from a data set and transform it into an understandable structure for further use. Aside from the raw analysis step, it involves database and data management aspects, data pre-processing, model and inference considerations, interestingness metrics, complexity considerations, post-processing of discovered structures, visualization, and online updating. [1]

The term is a misnomer, because the goal is the extraction of patterns and knowledge from large amount of data, not the extraction of data itself. It also is a buzzword and is frequently applied to any form of large-scale data or information processing (collection, extraction, warehousing, analysis, and statistics) as well as any application of computer decision support system, including artificial intelligence, machine learning, and business intelligence. The popular book "Data mining: Practical machine learning tools and techniques with Java" (which covers mostly machine learning material) was originally to be named just "Practical machine learning", and the term "data mining" was only added for marketing reasons. Often the more general terms "(large scale) data analysis", or "analytics" – or when referring to actual methods, artificial intelligence and machine learning – are more appropriate. [1]

The actual data mining task is the automatic or semi-automatic analysis of large quantities of data to extract previously unknown, interesting patterns such as groups of data records (cluster analysis), unusual records (anomaly detection), and dependencies (association rule mining). This usually involves using database techniques such as spatial indices. These patterns can then be seen as a kind of summary of the input data, and may be used in further analysis or, for example, in machine learning and predictive analytics. For example, the data mining step might identify multiple groups in the data, which can then be used to obtain more accurate prediction results by a decision support system. Neither the data collection, data preparation, nor result interpretation and reporting are part of the data mining step, but do belong to the overall KDD process as additional steps. [1]

The related terms data dredging, data fishing, and data snooping refer to the use of data mining methods to sample parts of a larger population data set that are (or may be) too small for reliable statistical inferences to be made about the validity of any patterns discovered. These methods can, however, be used in creating new hypotheses to test against the larger data populations. [1]

III. EXISTING SOLUTION

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 [10]. It is mining for association rules in database of sales transactions between items which is important field of the research in dataset [7]. The benefits of these rules are detecting unknown relationships, producing results which can perform basis for decision making and prediction [10].

Apriori algorithm is easy to execute and very simple, is used to mine all frequent item sets in database. The algorithm makes many searches in database to find frequent item sets where k item sets are used to generate k+1-itemsets. Each k-itemset must be greater than or equal to minimum support threshold to be frequency. Otherwise, it is called candidate itemsets. In the first, the algorithm scan database to find frequency of 1-itemsets that contains only one item by counting each item in database. The frequency of 1-itemsets is used to find the itemsets in 2- itemsets which in turn is used to find 3-itemsets and so on until there are not any more k-itemsets.

In FP Growth algorithm there is no generation of candidate set which saves a lot of our efforts. FP-Growth simplifies all the problems present in apriori by using a structure called an FP-Tree. In an FP-Tree each node

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 5, Issue 2, March 2016

represents an item and its current count, and each branch represents a different association. Following are the steps in which FP growth algorithm is carried out:

- Steps 1: FP-Growth Allows frequent itemset discovery without candidate itemset generation.
- Step 2 approach: (i). Step 1 Build a compact data structure called the FP-tree built using 2 passes over the data-set.
  (ii). Step 2 Extracts frequent itemsets directly from the FP-tree traversal through leaf node of the tree.

The frequent-pattern tree (FP-tree) is a compact structure that stores quantitative information about frequent patterns in a database.[11].Han defines the FP-tree as the tree structure defined below[9]:

1. One root labeled as "null" with a set of item-prefix sub trees as children, and a frequent-item-header table.
2. Each node in the item-prefix subtree consists of three fields:
    1. Item-name: registers which item is represented by the node;
    2. Count: the number of transactions represented by the portion of the path reaching the node;
    3. Node-link: links to the next node in the FP-tree carrying the same item-name, or null if there is none.
1. Each entry in the frequent-item-header table consists of two fields:
    1. Item-name: as the same to the node;
    2. Head of node-link: a pointer to the first node in the FP-tree carrying the item-name

IV. PROPOSED SYSTEM

The proposed system helps us to improve the efficiency of the Apriori algorithm by using the transaction reduction methodology. Here we are using FP growth algorithm to analyze the improved Apriori algorithm against it.

A. Principle of Improved Apriori Algorithm

Apriori Algorithm generates a large number of candidate sets if the database is very large. 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.

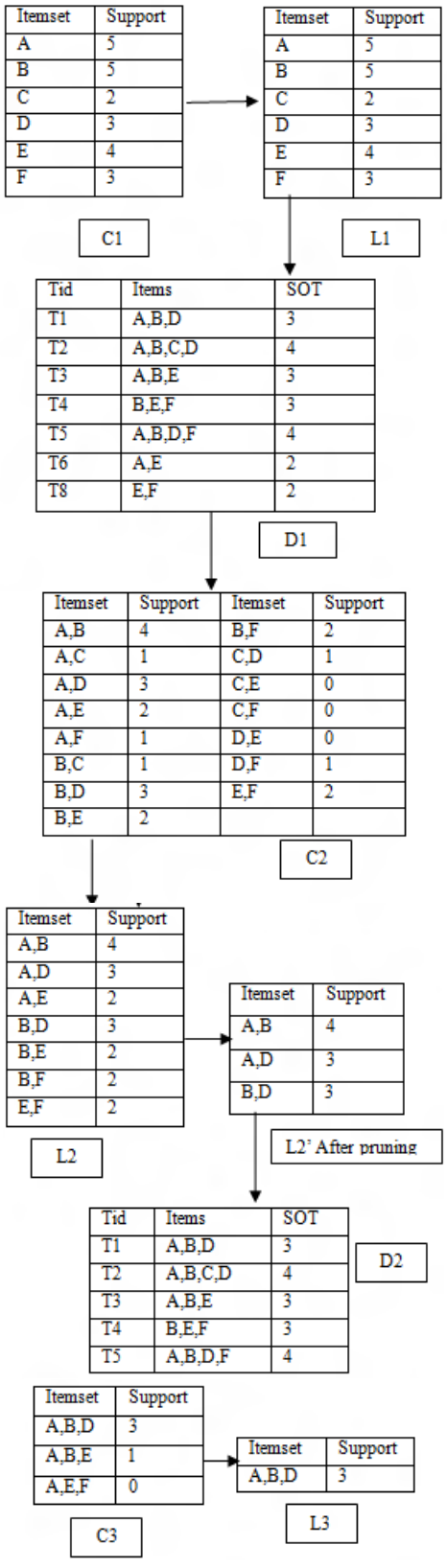
A. Example

Let us consider a transaction database as shown in table 1.

TID	ITEMS	SOT
T1	A,B,D	3
T2	A,B,C,D	4
T3	A,B,E	3
T4	B,E,F	3
T5	A,B,D,F	4
T6	A,E	2
T7	C	1
T8	E,F	2

Suppose the minimum support count min_sup=2. The algorithm proceeds as follows:

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 5, Issue 2, March 2016



Itemset	Support
A	5
B	5
C	2
D	3
E	4
F	3

C1

Itemset	Support
A	5
B	5
C	2
D	3
E	4
F	3

L1

Tid	Items	SOT
T1	A,B,D	3
T2	A,B,C,D	4
T3	A,B,E	3
T4	B,E,F	3
T5	A,B,D,F	4
T6	A,E	2
T8	E,F	2

D1

Itemset	Support	Itemset	Support
A,B	4	B,F	2
A,C	1	C,D	1
A,D	3	C,E	0
A,E	2	C,F	0
A,F	1	D,E	0
B,C	1	D,F	1
B,D	3	E,F	2
B,E	2		

C2

Itemset	Support
A,B	4
A,D	3
A,E	2
B,D	3
B,E	2
B,F	2
E,F	2

L2

Itemset	Support
A,B	4
A,D	3
B,D	3

L2' After pruning

Tid	Items	SOT
T1	A,B,D	3
T2	A,B,C,D	4
T3	A,B,E	3
T4	B,E,F	3
T5	A,B,D,F	4

D2

Itemset	Support
A,B,D	3
A,B,E	1
A,E,F	0

C3

Itemset	Support
A,B,D	3

L3

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 5, Issue 2, March 2016

Step 1: Firstly, the SOT column is added to the database.
Step 2: In the first iteration, each item is a member of candidate 1-itemsets, C1.The algorithm simply scans the database to count the occurrences of each item.
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=2, the set of frequent 1-itemset, L1 can be determined. It consists of the candidate 1-itemset, satisfying the minimum support threshold.
Step 5: As there are no items with support less than 2 nothing is deleted. 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 T7. Thus after deleting the transaction we 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 item-set in C2 is accumulated.
Step 8: L2, the set of frequent 2-itemsets is now then determined, consisting of those candidate 2-itemsets in C2 having minimum support greater or equal to the specified value.
Step 9: Now L2 is pruned by deleting the item sets with number of occurrences in Lk-1 less than k-1 to before candidate item sets occur i.e. itemsets {B,F}, {A,E}, {B,E}, {E,F} are deleted.
Step 10: After L2 is pruned, we find that the transactions T6 and T8 consists of only two items. Now, the value of k is2, delete the transactions 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 11: 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. There are a number of elements in C3. According to the property of Apriori algorithm, C3 needs is pruned.
Step 12: The transactions in D2 are scanned and the support count of each candidate itemset in C3 is accumulated. Use C3 to generate L3.
Step 13: L3 has only one 3-itemsets so that C4 =null. The algorithm will stop and give out all the frequent itemsets.
Step 14: Algorithm will be generated for Ck until Ck+1 becomes empty.

A. FP growth Algorithm

In an FP-Tree each node represents an item and its current count, and each branch represents a different association.

1.First we will calculate minimum support count using formula,

min_supcount=(min_sup in %)*no.of transaction

where,min_sup in % is given or asuumed.

2. Then we will create FP tree,

i. Firstly we will create table where each item will be prioritized according to its frequency in the whole database and according to that items list priority, each transaction's item will be arrange in descending order.

ii. For FP-tree generation root node is taken as null.

iii. Then add node with item-name along with the count, where count represent the number of repetition of item.

iv. If another transaction contain item which are already present then the count will be increased and the further item will be added in the tree.

3. Now the item are extracted from tree using each leaf node and frequent item set then combined to get final item set.

V. IMPLEMENTATION METHODOLOGY

Here we will study how the entire project idea was implemented into a run able code which produces a tangible output, i.e. a running version of software. Hence in this chapter we will learn implementation of the project. Implementation is vital part in software building and designing. Our project is implemented in Java. We have used NetBeans to build our Project.

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 5, Issue 2, March 2016

A. Improved Apriori 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 along with reducing the number of candidate item sets generated. In our proposed method, we introduced an attribute named 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. Depending on the value of K, algorithm searches the same value for SOT in database. If value of SOT matches with value of K then those transactions are deleted from the database. For reducing the size of the candidate item sets formed before the candidate item sets Ck are generated, further prune Lk-1, count the times of all items occurred in Lk-1, delete item sets with this number less than k-1 in Lk-1. In this way, the number of connecting items sets and the size of the database to be scanned will be decreased, so that the number of candidate items will decline.

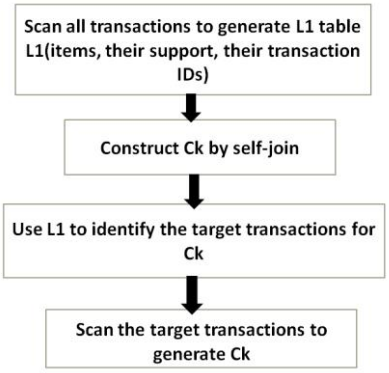


Fig. 1

B. FP Growth Algorithm

In an FP-Tree each node represents an item and its current count, and each branch represents a different association. It is a scalable technique for mining frequent patterns in the database.

FP-Growth: allows frequent itemset discovery without candidate itemset generation. Two step approach:

Step 1: Build a compact data structure called the FP-tree

- Built using 2 passes over the data-set.

Step 2: Extracts frequent itemsets directly from the FP-tree

A. Example

Given a transaction database DB and a minimum support threshold ξ, find all frequent patterns (item sets) with support no less than ξ.

Input:

TID	*Items bought*
100	{*f, a, c, d, g, i, m, p*}
200	{*a, b, c, f, l, m, o*}
300	{*b, f, h, j, o*}
400	{*b, c, k, s, p*}
500	{*a, f, c, e, l, p, m, n*}

Minimum support: ξ =3

Output: all frequent patterns, i.e., *f, a, …, fa, fac, fam, fm,am…*

Step 1: Scan database for the first time to generate L

TID	*Items bought*
100	*{f, a, c, d, g, i, m, p}*
200	*{a, b, c, f, l, m, o}*
300	*{b, f, h, j, o}*

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 5, Issue 2, March 2016

400	*{b, c, k, s, p}*
500	*{a, f, c, e, l, p, m, n}*

⇩

ITEM	*FREQUENCY*
f	4
c	4
a	3
b	3
m	3
p	3

Step 2: Scan the database for the second time, order frequent items in each transaction.

TID	*Items bought*	*(ordered) frequent items*
100	*{f, a, c, d, g, i, m, p}*	*{f, c, a, m, p}*
200	*{a, b, c, f, l, m, o}*	*{f, c, a, b, m}*
300	*{b, f, h, j, o}*	*{f, b}*
400	*{b, c, k, s, p}*	*{c, b, p}*
500	*{a, f, c, e, l, p, m, n}*	*{f, c, a, m, p}*

Step 3: Construct FP tree

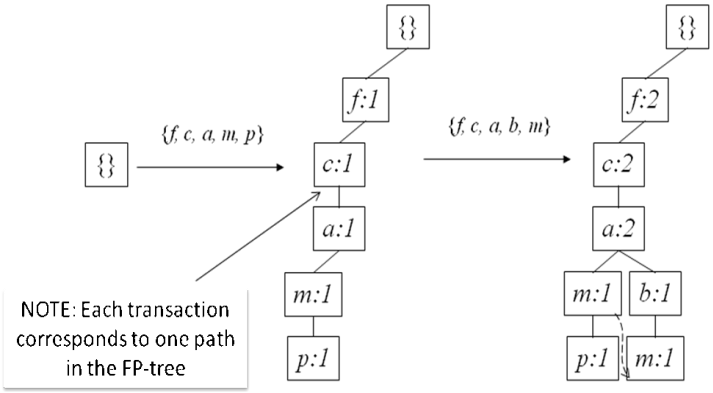


Fig. 2

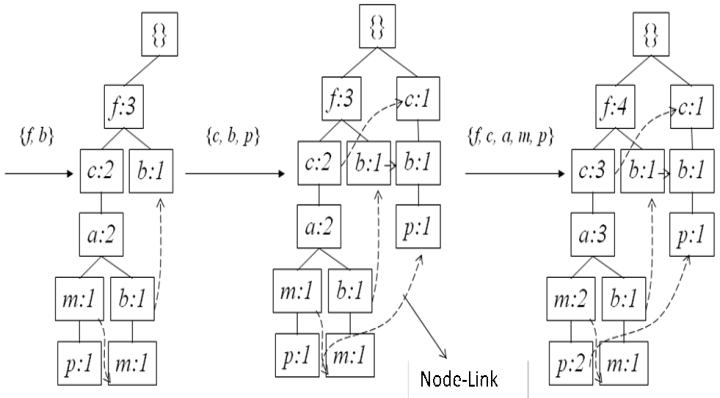


Fig. 3

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 5, Issue 2, March 2016

FINAL FP TREE:

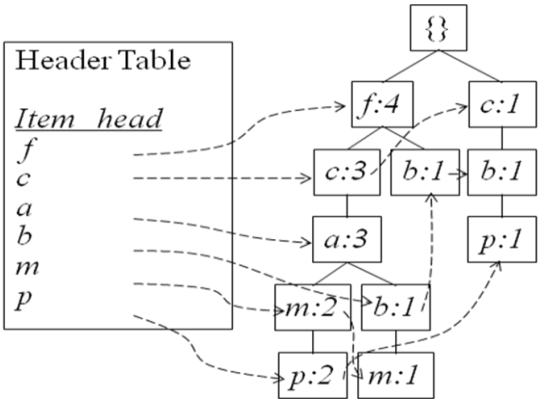


FIG. 4

C. Analysis

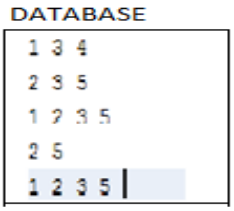
Here analysis is done between the Improved Apriori Algorithm and FP Growth Algorithm. Following are the various points on which the comparison is done.

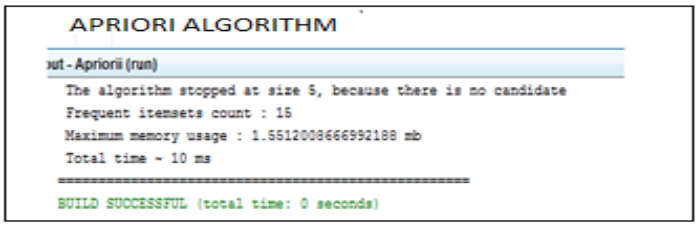
Table: Analysis

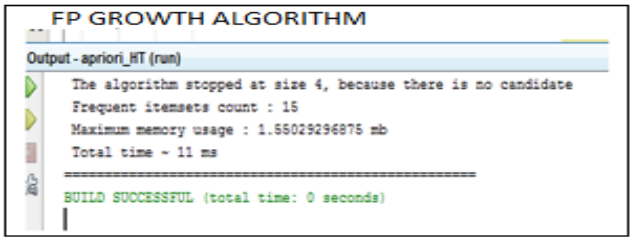
Properties	Advance Apriori	FP Growth
1.Technique	Use Apriori property and join and prune property	It constructs conditional frequent pattern tree which satisfy minimum support.
2.Memory Usage	Require less memory as compared to FP Growth	Requires more memory as compared to Advance Apriroi
3.Parallelizability	Candidate generation is parallelizable.	Data are very interdependent, each node needs the root.
4. No. of scans	Database is pruned after every scan.	Scans the DB only twice and twice.
5. Runtime	Runtime increases exponentially depending on the number of items	Runtime increases linerally, depending on the number of transactions

VI. RESULT

A. Screenshot







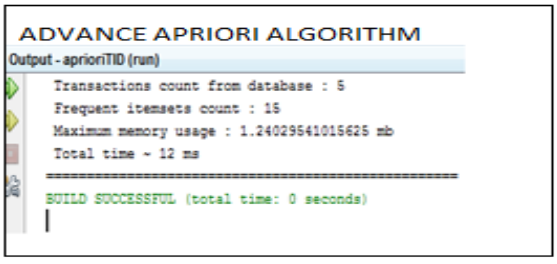


Fig.5

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 5, Issue 2, March 2016

VII. CONCLUSION

In this paper, Apriori algorithm is improved based on the properties of cutting database and pruning of the frequent item-set. The typical Apriori algorithm has performance bottleneck in the large data processing so that we need to optimize the existing algorithm with variety of methods. Thus the optimized algorithm prunes Lk-1 before we generate Ck. Thus decreasing the number of connecting item sets and removing the item-sets which does not satisfies the conditions. The improved algorithm proposed optimizes the algorithm by reducing the size of the candidate set Ck and also reducing the I/O spend by cutting down transaction records in the database. The proposed algorithm also reduces the time to repeat the process. For large database, this algorithm can save time cost and increase the efficiency of data mining. Thus 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 the existing algorithm and is more efficient but it has overhead to manage the new database after every generation of Lk. Thus, there should exist some approach which requires less number of scans of database. Another solution might be division of large database among processors. Also the analysis with Fp growth algorithm shows that the memory required for Fp growth algorithm is more as compared to improved Apriori algorithm. Thus Improved Apriori Algorithm proved to efficient in terms of memory requirements.

VIII. FUTURE ENCHANCEMENT

In future enhancement we would like to do the analysis on strings of characters that is we can take a market base analysis to do the same. This will help the distributors or manager know what exactly the requirement of the customer is based on the association between the items.

REFERENCES

[1] http://en.wikipedia.org/wiki/Data_mining.

[2] S. Rao, R. Gupta, “Implementing Improved Algorithm Over APRIORI Data Mining Association Rule Algorithm”, International Journal of Computer Science And Technology, pp. 489-493, Mar. 2012.

[3] H. H. O. Nasereddin, “Stream data mining,” International Journal of Web Applications, vol. 1, no. 4, pp. 183–190, 2009.

[4] F. Crespo and R. Weber, “A methodology for dynamic data mining based on fuzzy clustering,” Fuzzy Sets and Systems, vol. 150, no. 2, pp. 267–284, Mar. 2005.

[5] Sotiris Kotsiantis, Dimitris Kanellopoulas, “Association Rules Mining: A Recent Overview”, GESTS International Transactions on Computer Science and Engineering, Vol. 329(1), pp. 71-82, 2006.

[6] M. Halkidi, “Quality assessment and uncertainty handling in data mining process,” in Proc, EDBT Conference, Konstanz, Germany, 2000.

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

[8] https://en.wikipedia.org/wiki/Apriori_algorithm.

[9] J. Han, H. Pei, and Y. Yin. Mining Frequent Patterns without Candidate Generation. In: Proc. Conf. on the Management of Data (SIGMOD’00, Dallas, TX). ACM Press, New York, NY, USA 2000.

[10] F. H. AL-Zawaidah, Y. H. Jbara, and A. L. Marwan, “An Improved Algorithm for Mining Association Rules in Large Databases,” Vol. 1, No. 7, 311-316, 2011.

[11] B.Santhosh Kumar and K.V.Rukmani. Implementation of Web Usage Mining Using APRIORI and FP Growth Algorithms. Int. J. of Advanced Networking 1and Applications, Volume: 01, Issue: 06, Pages: 400-404 (2010).