IOSR Journal of Engineering (IOSRJEN) **www.iosrjen.org**
ISSN (e): 2250-3021, ISSN (p): 2278-8719
Vol. 05, Issue 03 (March. 2015), ||V3|| PP 30-34

An Improved Technique for Frequent Itemset Mining

Patel Atul R.[#], Patel Tushar S.[$]
[#]*ME (CSE Student), S. P. B. Patel Engineering College, Mehsana, Gujarat, India*
[$]*IT Department, S. P. B. Patel Engineering College, Mehsana, Gujarat, India*

Abstract: *An association rule mining is important in data mining. Two Steps important in association rule mining. First, find the frequent itemset from dataset and Second, find the association rule from frequent itemsets. A frequent itemsets mining is crucial and most expensive step in association rule mining. In Apriori and Apriori-like principle it's known that the algorithms cannot perform efficiently due to high and repeatedly database passes. In this paper we proposed a improved technique for frequent itemset mining. This technique scan the database only once and reduces the number of transaction.*

Keywords: *Frequent, Support, Confidence*

I. INTRODUCTION

Data mining is applicable to real data like industry, textile showroom, super market etc. Association rule is one of the data mining technique is used to generate association rules. The association rule is used to find the frequent item sets from the large data. Frequent patterns are patterns (i.e. itemsets) that appears in a dataset frequently. A set of items, i.e. computer and antivirus that appears frequently together in a transaction dataset is a frequent itemsets. Frequent patterns mining like Frequent itemsets find frequent itemsets from the small database and/or large database, where the database are either transactional or relational. The frequent itemset mining is the process of finding out frequent itemsets from the DB.

Apriori and FP-Growth are known to be the two important algorithms each having different approaches in finding frequent itemsets[1][2]. The Apriori Algorithm uses Apriori Property in order to improve the efficiency of the level-wise generation of frequent itemsets. On the other hand, the drawbacks of the algorithm are candidate generation and multiple database scans. FP-Growth comes with an approach that creates signatures of transactions on a tree structure to eliminate the need of database scans and outperforms compared to Apriori[2].

In this paper, section 2 discuss the review of our work; section 3 we proposed an improved technique for frequent itemset mining; section 4 discussion about improved technique; Finally section 5 concludes the paper.

II. RELATED WORK

2.1 Support

Support is the ratio of the number of transactions that include all items in the antecedent and consequent parts of the rule to the total number of transactions. Support is an association rule interestingness measure.

$$\text{Support} = P(A \vee B) = \left(\frac{\text{Number of transactions containing both A and B}}{\text{Total no of transactions}} \right)$$

A and B represents a itemsets in a Database D.

2.2 Confidence

Confidence is the ratio of the number of transactions that include all items in the consequent as well as antecedent to the number of transactions that include all items in antecedent. Confidence is an association rule interestingness measure.

$$\text{Confidence} = P(B|A) = \frac{P(A \vee B)}{P(A)} = \left(\frac{\text{Number of transactions containing both A and B}}{\text{Number of transaction containing A}} \right)$$

Apriori algorithm introduced by Agrawal 1994[1], is a first frequent pattern and association rule mining algorithm. Apriori algorithm used to find the all frequent itemset and its association rules, the algorithm use breadth-first (level-wise) search method which is known as iterative approach. The key feature of Apriori algorithm is to make multiple database passes. Two main disadvantages of the Apriori algorithm, more time scan the database and huge candidate itemset generate.

Studies various frequent itemset mining algorithm has been introduced to solve the drawback of Apriori algorithm. First, More times scan the database. Second, Generate huge candidate itemset. Use a different data structure and other techniques, solve the problem to has been introduced various FIM alorithm. CBT-fi for Mining Frequent Itemsets[9] this technique Reduce the transaction, scan database only once, use less amount of memory; Index-BitTableFI: An improved algorithm for mining frequent itemsets[4] this technique similar transaction greatly, search space is reduced greatly; Mining frequent itemsets in large databases: The hierarchical partitioning approach[7] this technique there is no extra cost of re-scanning the original database and memory based algorithm for large database; An Improved Apriori Algorithm based on Matrix Data Structure[10] this technique scan database only once, works Top-bottom approach, reduce input/output cost; A Semi-Apriori Algorithm for Discovering the Frequent Itemsets[11] technique reduce candidate itemset and reduce total number of database pass;, An Association Rule Mining Algorithm Based On A Boolean Matrix[3] which technique database only once and not produce the candidate itemset; Improving the efficiency of Apriori Algorithm in Data Mining[8] which technique reduce candidate itemset and reduce the input- output cost;

III. AN IMPROVED TECHNIQUE

Following steps improved technique for frequent itemset mining.

Steps:

Step 1: Given transaction DB and minimum support.

Step 2: Add Count Column in M b_matrix. i.e Count represent the size of the row. And CC represent the count the number of 1 in every column.

Step 3: Delete infrequent items based on min_supp. (if CC< min_supp then remove the items column); and Rearrange b_matrix in descending order based on Count.

Step 4: Count distinct row and store the count value in TC in M b_matrix

Step 5: For each transaction T in M b_matrix, If (TC>=min_supp) Extract itemset its frequent move items with subset into FAL; then Remove the T; store the count value in TC in M' b_matrix.

Step 6: For each transaction T in M' b_matrix; Extract itemset and check into FAL if its present in FAL do not need AND operation else do AND operation; If (other respectively row count value grater then or equal to own count value) do AND operation; Results in same itemset structure as processing row's itemset structure then increase its support count value. Check support count value grater then or equal to minimum support then extract items its frequent and store into FAL.

M=Matrix ,TC= Transaction Count, FAL= Frequent Array List, CC= column count;

Pseudo Code:

```
Scan DB and convert into b _matrix
 For each column Mi of M
      if sum(Mi) < min_supp
        delete Mi from M;
 For each column Mr of M
      Count <— sum(Mr)
 Sort count in descending order
 if disctingrows in M
      rearrange M <— count distingrows in TC
 For each T in M do begin
    count <— sum(Mr)
 end
    write M, count, TC
 For each T in M
     if(TC>=min_supp)
       Extract itemset its frequent move itmes with subset
into FAL;
    then  Remove the T;
    else
    write M'
 end
     write update M', extrract itemset with subset in to
FAL;
 For each T in M'
     Extract itemset  and check into FAL
         if its present in FAL do not need  AND operation
             else do AND operation;
                  If (other respectively row count >= own count
value)
                  do AND operation
                  Results in same itemset structure as processing
                  row's itemset structure then increase its
support
                  count value;
 end
```

```
Rearrange b_matrix in descending order based on
Count.
 Count distinct row store the count value in TC in M
b_matrix.
 For each transaction T in M b_matrix do begin
        count number of 1 in each transaction store in
       count column
 end
 For each transaction T in M b_matrix
       If (TC>=min_supp)
            Extract itemset its frequent move itmes with
subset into FAL;
      then  Remove the T;
      else
      write M' b_matrix
 end
      write update  M' b_matrix, extrract itemset with
subset in
      to FAL
 For each transaction T in M' b_matrix
            Extract itemset  and check into FAL
           if its present in FAL do not need  AND
operation
           else do AND operation;
           If (other respectively row count >= own count
value)
           do AND operation
           Results in same itemset structure as
processing
           row's itemset structure then increase its
support
           count value;
           if ( support count value >= min_supp)
           extract items its frequent move items with
subset into FAL;
 end
```

Example:

Step 1: Above show the steps procedure step by step description improved technique. Here min_supp = 3;

Table 1

TID	Items
T1	I1,I2,I3,I5
T2	I2,I3
T3	I2,I3,I4
T4	I1,I3,I5
T5	I1,I2,I3
T6	I1,I3,I5

Step 2: Add count column in Table 1.

Table 2

TID/I	I1	I2	I3	I4	I5	count
T1	1	1	1	0	1	4
T2	0	1	1	0	0	2
T3	0	1	1	1	0	3
T4	1	0	1	0	1	3
T5	1	1	1	0	0	3
T6	1	0	1	0	1	3
CC	4	4	6	1	3	

Step 3: Remove the Items I4 based on min_supp from Tab 2 and rearrange M b_matrix based on descending-order from Tab 3;

Table 3

TID/I	I1	I2	I3	I5	count
T1	1	1	1	1	4
T2	0	1	1	0	2
T3	0	1	1	0	2
T4	1	0	1	1	3
T5	1	1	1	0	3
T6	1	0	1	1	3

Table 4

TID/I	I1	I2	I3	I5	count
T1	1	1	1	1	4
T4	1	0	1	1	3
T5	1	1	1	0	3
T6	1	0	1	1	3
T2	0	1	1	0	2
T3	0	1	1	0	2

Step 4: In M b_matrix; If (TC>=min_supp) Extract itemset its frequent move items with subset into FAL from Table 4; FAL={(I1,I3,I5),(I2,I3),(I1,I3),(I1,I5),(I3,I5),(I1),(I2), (I3),(I5)}

Table 5

TID/I	I1	I2	I3	I5	count	TC
T1	1	1	1	1	4	1
T4	1	0	1	1	3	2
T5	1	1	1	0	3	1
T2	0	1	1	0	2	2

Step 5: Generate M' b_matrix from Table 5 and used the M' b_matrix find the frequent itemset.

Table 6

TID/I	I1	I2	I3	I5	count	TC
T1	1	1	1	1	4	1
T5	1	1	1	0	3	1

Step 6:

Select First Row T1 and extract items{I1,I2,I3,I5}, check FAL if items is not present in FAL do need AND operation. Results in same itemset structure as processing row's itemset structure then increase its support count value. Here result in not same itemset structure as processing row's itemset structure. And move to next row.

Select First Row T5 and extract items{I1,I2,I3}, check FAL if items is not present in FAL do need AND operation and calculate support count value is equal to 4. If support count value is greater than or equal to min_supp. Then extract item its frequent and move items with its subset into FAL. Move items (I1,I2,I3) with subset into FAL.

FAL = {(I1,I3,I5),(I2,I3),(I1,I3),(I1,I5),(I3,I5),(I1),(I2), (I3),(I5), (I1,I2,I3), (I1,I2)}

IV. DISCUSSION

Apriori algorithm used for extracting frequent itemsets faces two main disadvantages. Firstly it scans the database multiple times and secondly it generates large number of irregular itemsets hence increases spatial and temporal complexities and overall decreases the efficiency of classical apriori algorithm use to a our improve technique for frequent itemset mining. Reduce the execution time compare to the Apriori algorithm. An improved technique that can be used resolve the problem apriori algorithm.

V. CONCLUSION

We can conclude reduce the do AND operation and also compress data structure & find out frequent itemset. Reduce the transaction and input/output cost. Also find the frequent itemset from largest frequent itemset to smallest frequent itemset. Only one time scan the original database.

REFERENCES

[1] R. Agrawal, T. Imielinski, and A. Swami,"Mining association rules between sets of items in large databases," ACM, 1993, pp. 207- 216.
[2] Han, J., J. Pei, and Y. Yin ,"Mining frequent patterns without candidate generation" Proceedings of the ACM, NY, pp. 1–12,2000.
[3] Hanbing Liu and Baisheng Wang," An Association Rule Mining Algorithm Based On A Boolean Matrix, Data Science Journal,2007.
[4] Wei Song, Bingru Yang and Zhangyan Xu, " Index-BitTableFI: An improved algorithm for mining frequent itemsets," Proceedings of the Elsevier, March 2008.
[5] Guoxiaoli, Fengli and Guoping, " Research on Mining Frequent Itemsets based on Bitwise AND Algorithm," Proceedings of the IEEE, 2011.
[6] Predrag Stanisic and Savo Tomovic, " Frequent Itemset Mining Using Two-Fold Cross-Validation Model," Proceedings of the IEEE, 2012.
[7] Fan-Chen Tseng, " Mining frequent itemsets in large databases: The hierarchical partitioning approach," Proceedings of the Elsevier, 2013.
[8] Vipul Mangla, Chandni Sarda and SarthakMadra," Improving the efficiency of Apriori Algorithm in Data Mining", IJEIT, 2013.
[9]. A.Saleem Raja and E.George Dharma Prakash Raj, " CBT-fi: Compact BitTable Approach for Mining Frequent Itemsets," Proceedings of the Advances in Computer Science: an International Journal, sep 2014.
[10] Shalini Dutt, Naveen Choudhary and Dharm Singh., " An Improved Apriori Algorithm based on Matrix Data Structure," Proceedings of the Global Journal of Computer Science and Technology: C Software & Data Engineering, 2014.
[11] Sallam Osman Fageeri, Rohiza Ahmad and Baharum B. Baharudin, " A Semi-Apriori Algorithm for Discovering the Frequent Itemsets," Proceedings of the IEEE, 2014.
[12] Christian Borgelt, "Simple Algorithms for Frequent Item Set Mining."European Center for Soft Computing,Asturias, Spain.
[13] Jiawei Han and Kamber, Data Mining: concept and Techniqes, second edition, Elsevier.