Integrated Intelligent Research (IIR)
International Journal of Data Mining Techniques and Applications
Volume: 03 Issue: 02 December 2013, Page No.64- 68
ISSN: 2278-2419

A Framework to Discover Association Rules Using Frequent Pattern Mining

Shajeeah. M[1], Safana. N[2], FathimathRajeela. K.A.[3], Kadeejath Sajida[4], Zahid Ansari[5]

Department Of Computer Science Engineering, P. A. College of Engineering, Mangalore, India
Email: [1]shajeeah.m@gmail.com, [2]ksafa22@gmail.com, [3]rajeela.nrs@gmail.com, [4]kdjthsaji@gmail.com, [5]zahid_cs@pace.edu.in

Abstract-Mining association rule is one of the key problems in data mining approach. Association rules discover the hidden relationships between various data items. In this paper, we propose a framework for the discovery of association rules using frequent pattern mining. We use preprocessing to transform the transaction dataset into a 2D matrix of 1's and 0's. Mining association rule must firstly discover frequent itemsets and then generate strong association rules from the frequent itemsets. The Apriori algorithm is the most well known association rule mining algorithm and is less efficient because they need to scan the database many times and store transaction ID in memory, so time and space overhead is very high. Especially they are less efficient when they process large scale database. Here we propose improved Apriori algorithm by including prune step and hash map data structure. The improved algorithm is more suitable for large scale database. Experimental results shows that computation times are reduced by using the prune step and hash map data structure.

Keywords— Apriori; Frequent Itemset; Association Rule; Market Basket Analysis; Support; Confidence.

I. INTRODUCTION

Association rule mining is an important technique for data mining. The aim of association rule mining is to find the correlations, frequent patterns, associations, or casual structures among sets of items or objects in transaction databases, relational databases, and other information repositories. Association rule mining finds out the association rules which satisfies pre specified minimum support and confidence from the database [1].Association rule mining can be viewed as a two-step process. First step is to find all the item sets which will occur at least as frequently as a predetermined minimum support. These itemsets are called as frequent itemsets. A subset of frequent itemset must also be frequent itemset. The support of an itemset Xis defined as the proportion of transactions in the data set which contain the itemset.

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

The second step is to generate strong association rules from frequent itemsets using the criteria confidence.The confidence of an association rule R = "X → Y" (with item sets X and Y) is the support of the set of all items that appear in the rule (here: the support of S = X ∪ Y) divided by the support of the antecedent (also called "if-part" or "body") of the rule (here X) [2]. That is,

$$\text{conf}(R) = \frac{\text{supp}(X \cup Y)}{\text{supp}(X)}$$

Rules that satisfy both a minimum support (min_sup) and minimum confidence (min_conf) are called strong. By convention we writesupport and confidence values so as to occur between 0%and 100% rather than 0 to 1.0[3].The first step is further divided into two steps: candidate itemset generation and frequent itemset generation. Association rule mining is defined in the following manner: let I be the set of items where I= $\{I_1, I_2, I_3,\ldots,I_m\}$ and D be the set of database transactions where each transaction T is the set of items such that $T \subseteq I$. Each transaction has an identifier associated with it, called TID. Let M be set of items. A transaction T is said to contain M if and only if $M \subseteq T$. Association rule is an implication of the form $M \Rightarrow N$, where $M \subset I$, $N \subset I$, and $M \cap N = \Phi$ [4].

The broad applications of frequent pattern mining are Market Basket Analysis, Cross-marketing, Catalog Design, Sale Campaign analysis, Web Log Analysis and DNA Sequence Analysis. In this paper, we use the example of Market Basket Analysis to discover association rules. Market basket analysis is one of the most common and useful types of data analysis for marketing and retailing. This process analyzes customer buying habits by finding associations between the different items that customer places in their shopping baskets. The discovery of such associations can help retailers develop marketing strategies by gaining insight into which items are frequently purchased together by customers. For example, if customers buying milk also tend to buy bread at the same time, then placing the milk closer or opposite to bread may help to increase sales of both these items.There are different mining algorithms of association rules. Apriori is one of the algorithms which are used to find frequent itemsets from the database and generate association rules [5].First, we need to preprocess the transaction dataset to transform it into a 2D matrix of 1's and 0's. The output of preprocessing is given to the improved apriori algorithm. The algorithm generates frequent itemsets by generating candidate itemsets using prune step. We used hash map to store the results. Association Rules are generated for the obtained frequent itemsets. If there are N frequent itemsets, only N/2 number of processing is required to generate association rules.Further, experimental results shows that computation times are reduced by using the proposed method.

II. RELATED WORKS

Researchers have developed too many methods for determining association rules. The first algorithm which was put forward for mining association rule was the AIS (Agrawal, Imielinski and Swami) algorithm [6]. The AIS algorithm generated only one item consequent association rules, that is the rules contain only one item, for example rules like $A \cap B \Rightarrow C$ are generated not $A \Rightarrow B \cap C$. But this algorithm generated many candidate itemsets that turned out to be small that requires more memory and too many passes of the database.TBAR (Tree-Based Association Rule Mining) algorithm was an efficient algorithm

International Journal of Data Mining Techniques and Applications
Volume: 03 Issue: 02 December 2013, Page No.64- 68
ISSN: 2278-2419

proposed for association rule mining that used techniques such as Direct Hashing and Pruning (DHP) [7]. It reduced the problem of database scanning time and scanned the database once.PAFI (Partition Algorithm for Frequent Itemset) uses the clustering technique for improving the computing time and efficiency by reducing the number of scans of the database [8]. The database scanning time for the PAFI algorithm can still be reduced if improved if improved Apriori is applied after partitioning.Kumar *et al.* [9] proposed method to identify website usage pattern from web log files. They used Apriori algorithm and FP Growth algorithm for this purpose. For the Apriori algorithm, the cost of generating candidate set is costly and the FP Growth algorithm lacks a good candidate generation method.Wur *et al.* [10] proposed an effective Boolean algorithm for association rule mining in large databases of transactions in sales. It uses logic OR and AND operations for computing frequent itemsets and uses logic AND and XOR for deriving the interesting association rules from the frequent itemsets.

III. PROPOSED METHOD

The main goal of the proposed method is finding the association rules from frequent itemsets. The method operates in following three modules:

- Pre-processing
- Frequent Itemset Generation
- Association Rule Generation

We used Hash Map data structure for storing candidate itemsets and for generating frequent itemsets and association rules.

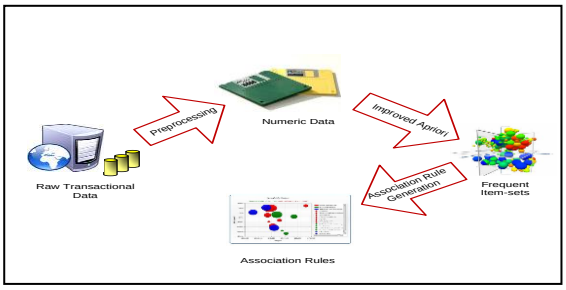

Figure. 1 Proposed Framework

A. Pre-processing

Here, transaction dataset shown in Table 1 is given as an input to the pre-processing algorithm and the algorithm is given in Algorithm 1.

TABLE 1
TRANSACTION DATASET

Transaction ID	Items from customers who brought more than 1 item
1	Apple Banana Mango
2	Banana Lemon
3	Banana Orange
4	Apple Banana Lemon
5	Apple Orange
6	Banana Orange
7	Apple Orange
8	Apple Banana Orange Mango
9	Apple Banana Orange

Algorithm 1: Preprocessing. Transform the transaction database into 2D matrix of 1's and 0's.

Input: D, a database of transaction.

Output: M, 2D matrix of 1's and 0's of D

Method:

(1) I = getIds (D); // create Id's for each items
(2) M = create_matrix (I);

ProceduregetIds (D)

(1) **for each** transaction t ∈ D
(2) **for each** item i ∈ t
(3) **if** (i ∉ I) {
(4) assign new id, Id to i;
(5) add item i and Id to I;
(6) }
(7) **else**
(8) add item i and Id of i to I;
(9) **return** I;

Procedure create_matrix (I)

(1) **for each** transaction t ∈ I
(2) **for each** item i ∈ t
(3) **{**
(4) assign 1's to present items and 0's to absent items;
(5) }
(6) **return** M;

Pre-processing algorithm process the transaction dataset and assign Transaction ID's to each items and then the dataset is converted into 2D matrix of 1's and 0's which is shown in Tables 2 and Table 3 respectively.

TABLE 2
LIST OF ITEM ID'S FOR ITEMS IN TABLE 1

Transaction ID	Item Id's
1	1 2 3
2	2 4
3	2 5
4	1 2 4
5	1 5
6	2 5
7	1 5
8	1 2 5 3
9	1 2 5

TABLE 3
MATRIX REPRESENTATION OF TABLE2

Transaction ID	Apple	Banana	Mango	Lemon	Orange
1	1	1	1	0	0
2	0	1	0	1	0
3	0	1	0	0	1
4	1	1	0	1	0
5	1	0	0	0	1
6	0	1	0	0	1
7	1	0	0	0	1
8	1	1	1	0	1
9	1	1	0	0	1

International Journal of Data Mining Techniques and Applications
Volume: 03 Issue: 02 December 2013, Page No.64- 68
ISSN: 2278-2419

B. Frequent Itemset Generation

Apriori algorithm is used to generate frequent itemsets. The output of pre-processing is given as input to apriori algorithm. First, candidate itemsets are generated for generating frequent itemsets. We used prune step reduce the size of candidate itemset i.e. if any subset of candidate itemset is not frequent that itemset is removed from the candidate set.The algorithm for apriori is given in Algorithm 2.

Algorithm 2: Apriori. Find frequent itemsets using n iterative level-wise approach based on candidate generation.**Input:**

- M, a 2D matrix of 1's and 0's;
- min_sup, the minimum support count threshold.

Output:L, frequent itemsets in M.

Method:

(1) L_1= find_frequent_ 1-itemsets (M);

(2) **for** (k = 2;$L_{k-1} \neq \emptyset$; k+ +) **{**

(3) C_k = apriori-gen (L_{k_1}); //New candidates

(4) **foreach** transactions t∈M {

(5) C_t = subset (C_k,t); //Candidates contained in t

(6) **for each** candidates c ∈C_t do

(7) c.count++;

(8) }

(9) L_k = {c ∈C_k|c.count ≥ minsup};

(10) }

(11)**return** L=∪$_k L_k$

Procedure apriori_gen(L_{k-1}:frequent(k-1)-itemsets)

(1) **for each** itemset $l_1 \in L_{k-1}$
(2) **for each** itemset $l_2 \in L_{k-1}$
(3) **if** (l_1 [1] = l_2 [1]) ∧ (l_1 [2] = l_2 [2]) ∧ ... ∧ (l_1 [k-2] = l_2 [k-2]) ∧ (l_1 [k-1] = l_2 [k-1])
(4) **then** {
(5) c = l_1 x l_2; // join step: generate candidates
(6) **if** has_infrequent_subset(c, L_{k-1}) **then**
(7) delete c; // prune step: remove unfruitful //candidate
(8) **else** add c to C_k;
(9) }
(10) **return** C_k;

Procedure has_infrequent_subset(c: candidate k-itemset; L_{k-1}: frequent of (k-1) itemsets); // use prior knowledge

(1) **for each** (k-1)-subset s ofc
(2) **if** s∉L_{k-1}**then**
(3) **return** TRUE;
(4) **return** FALSE;

Improved Apriori algorithm uses a Hash map data structure to overcome some of the weaknesses of the Apriori algorithm by reducing the number of candidate k-2-itemsets, since it is the key to improve the performance. Once frequent 1-item set L1 is generated, from the candidate item set in C1, candidate 2-itemsets are stored in Hash map. Hash map puts them into the different buckets and maintains the bucket counts. A 2-itemset whose corresponding bucket count in the hash table is below the support threshold cannot be frequent and hence removed from the candidate set. The output of Apriori algorithm is given in Table 4. The minimum support is given by the user and frequent itemsets are generated if their support count is greater than the minimum support.

TABLE 4
FREQUENT ITEMSETS FOR MINIMUM SUPPORT = 22%

Frequent 1 Itemset	
Items	**% Support**
Apple	66.66667
Banana	77.77778
Mango	22.222223
Lemon	22.222223
Orange	66.66667
Frequent 2 Itemset	
Items	**% Support**
Apple Banana	44.444447
Apple Mango	22.222223
Apple Orange	44.444447
Banana Mango	22.222223
Banana Lemon	22.222223
Banana Orange	44.444447
Frequent 3 Itemset	
Items	**% Support**
Apple Banana Mango	22.222223
Apple Banana Orange	22.222223

C. Association Rule Generation

For each frequent itemset l, generate all non-empty subsets of l. For every non-empty subset s of l, output the rule "s⇒ (l-s)" if (support_count (l)/support_count(s)) ≥ min_conf, where min_conf is the minimum confidence threshold.The algorithm for association rule generation is given in Algorithm 3.

Algorithm 3: Association. To generate association rules for the frequent itemsets.

Input:

- *L*, frequent itemsets.
- min_conf, minimum confidence.

Output: A, association rules.

Method:

(1) A = generateAssociationRules(*L*);
(2) **return** A;

Procedure generateAssociationRules (*L*: frequent itemsets)

(1) **for** (n=2; $L_n \neq \Phi$; n++){
(2) **for each** frequent set *l* in L_n**{**
(3) s_n = subsets (L_n);
(4) **for each** subset m in s_n**{**
(5) f = m;
(6) s = m+1;
(7) **if** (! issubset (f, s) && union (f, s) = = n){
(8) A_1 = { f→s, f, s ∈s_n | ((f∪s.count)/f.count) ≥ min_conf };
(9) A_2 = { s→f, f, s ∈ s_n | ((s∪f.count)/s.count) ≥ min_conf};
(10) A = A ∪ A_1 ∪ A_2;
(11) }
(12) }

International Journal of Data Mining Techniques and Applications
Volume: 03 Issue: 02 December 2013, Page No.64- 68
ISSN: 2278-2419

(13) }
(14) }

Procedureissubset (f: subset, s: subset)

(1) **If (**f ⊂s)
(2) **return** 1;
(3) **else if**(s⊂f)
(4) **return** 1;
(5) **return** 0;

The output of Association Rule Generation algorithm is given in Table 5. The minimum confidence is given by the user and frequent itemsets are generated based on the minimum support.

IV. EXPERIMENTAL RESULTS

The algorithm is implemented using Java Eclipse and run on a 2.5GHz Intel Core Processor with 4GB RAM in Windows/Linux platform. The experiment was performed using transaction database containing 100, 500 and 1000 transactions per database and some of the results are shown below. In the proposed method we used hash map to store the results which reduces the execution time.

TABLE 5
ASSOCIATION RULE FOR FREQUENT ITEMSET SATISFYING MINIMUM CONFIDENCE = 60%

X	->	Y	% Confidence
Lemon	->	Banana	100.0
Apple	->	Banana	66.666664
Mango	->	Banana	100.0
Mango	->	Apple	100.0
Orange	->	Banana	66.666664
Apple	->	Orange	66.666664
Orange	->	Apple	66.666664
Banana Mango	->	Apple	100.0
Apple Mango	->	Banana	100.0
Mango	->	Apple Banana	100.0

We experimented this by using transaction database containing 100, 500 and 1000 transactions per database and minimum support is taken as 0.2(20.0%). Comparative results of execution time of the algorithm by using hash map and vector is shown in "Fig. 2". The figure shows that execution time of the algorithm is reduced by using hash map when compared to vector.

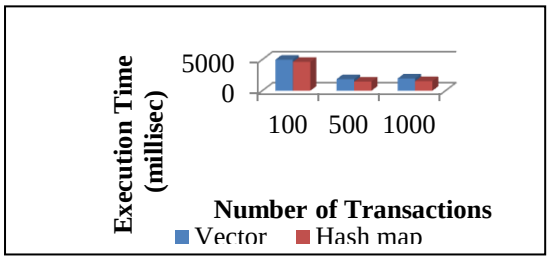

Figure. 2 Comparative results of execution time of the algorithm when using hash map and vector with minimum support=0.2(20%).

Second, Comparative results of execution time of the algorithm with and without prune step is shown in "Fig. 3". Minimum support is taken as 10.0% and the algorithm is tested by taking 100,500 and 100 transactions per database. The figure shows that the algorithm takes more time to execute without prune step and less time with prune step. In the proposed method we included prune step to improve the efficiency that is to reduce the execution time.

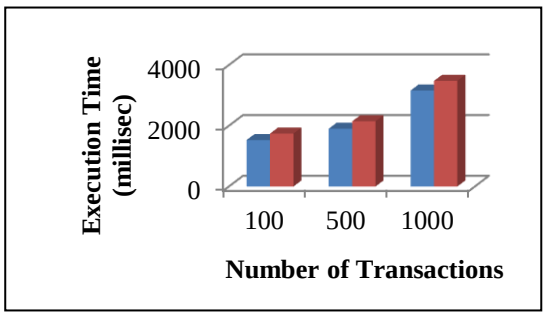

Figure. 3 Comparative results of execution time of the algorithm with and without prune step with minimum support=0.1(10%).

Third, Execution time of the algorithm is tested by giving different values for the support count. "Fig. 4" shows comparative results of execution time of the algorithm by taking minimum support as 0.1(10%) and 0.2(20%) for 100, 500 and 1000 transactions per database. The figure shows that execution time is more when the minimum support count is less and execution time is less when the minimum support count is more.

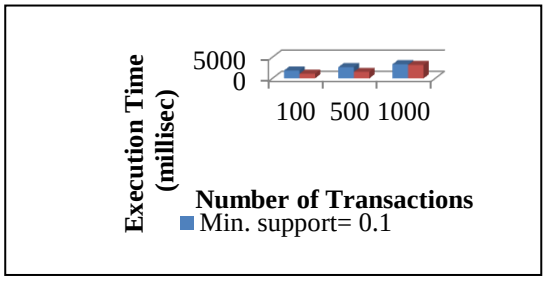

Figure. 4 Comparative results of execution time of the algorithm for varying minimum support.

The computation time increases as the number of transactions per database is increased. The result also shows that the smaller the minimum support count, more number of association rules are generated.

IV. CONCLUSIONS

In this paper, we proposed the discovery of association rules by generating the frequent itemset using improved Apriori algorithm. The improved Apriori algorithm uses hash map data structure and prune step, which reduced the time overhead. Also the improved algorithm is more suitable for large scale databases. We also proposed a preprocessing algorithm that transforms the input transaction dataset into a 2D matrix of 1's and 0's, which eliminates the need of scanning the transaction each time required.Experimental results show that the computation times are reduced by using the proposed method.

We express our gratitude to our institution and management for providing us with good infrastructure, laboratory facilities, and qualified staff whose guidance was of immense help in completion of this project successfully.We also thank our

parents for being so supportive and for their blessings. We are grateful to Department of Computer Science and Engineering and our institution P.A College of Engineeringfor imparting the knowledge with which we can do our best.

REFERENCES

[1] Sotiris Kotsiantis, and Dimitris Kanellopoulos, "Association Rule Mining: A Recent Overview," *GESTS International Transaction on Computer Science and Engineering*, vol. 32 (1), pp.71-82, 2006.

[2] Hipp, Jochen, Ulrich Güntzer, and Gholamreza Nakhaeizadeh. "Algorithms for association rule mining—a general survey and comparison." *ACM sigkdd explorations newsletter* vol. 2(1) pp. 58-64, 2000.

[3] A.A. Raorane, R.V. Kulkarni, and B.D. Jitkar, "Association rule-Extracting knowledge using market basket analysis", *Research Journal of Recent Sciences*, vol.1 (12), pp. 19-27, Feb.2012.

[4] Jiawei Han and Micheline Kamber, *Data Mining Concepts and Techniques*, 2nd ed, Morgan Kaufmann Publishers, March 2006.

[5] H. M Najadat, M. Al-Maolegi, B. Arkok, "An improved apriori algorithm for association rules," *International Research Journal of Computer Science and Applications*, vol. 1, pp. 01-08, June 2013.

[6] R. Agrawal, T. Imielinski, and A. Swami, "Mining association rules between sets of items in large databases," in *Proc. ACM SIGMOID Record*, vol. 22, pp. 207-216, 1993.

[7] Hanirex, D. Kerana, and M. A. Rangaswamy. "Efficient Algorithm for Mining Frequent Itemsets using clustering techniques."*International Journal on Computer Science & Engineering* vol. 3(3) 2011.

[8] F. Berzal, J. C. Cubero, N. Marin, J. N. Serano, TBAR, "An efficient method for association rulemining in relational database," *Elsevier Data and Knowledge Engineering, ,* pp. 47-64. 2001

[9] B. S. Kumar, K.V Rukmani, "Implementation of web usage mining using Apriori and FP growth algorithms," *International. Journal. of Advanced Networking and Applications*, vol. 01, issue: 06, pp. 400-404, 2010.

[10] Wur, Suh-Ying, and Yungho Leu. "An effective Boolean algorithm for mining association rules in large databases." In *Database Systems for Advanced Applications, 1999. Proceedings,IEEE6th International Conference on*, pp. 179-186. 1999..