Frequent Pattern Mining Using Apriori Based Algorithm

R. Suganya
Research Scholar, Department of Computer Science
Dr. SNS Rajalakshmi College of Arts and Science
Coimbatore, India

R. Tamil Selvi
Assistant Professor, Department of Computer Science
Dr. SNS Rajalakshmi College of Arts and Science
Coimbatore, India

***Abstract*—Association Rule Mining (ARM) is one of the data mining techniques used to extract hidden knowledge from database that can be used by an organization's decision makers to improve overall profit. Apriori algorithm is classical for association rule mining. But it repeatedly scans the database and the process of generating candidate itemsets takes long time. In this paper, Apriori based algorithms, such as "Reduced Candidate Set (RCS) algorithm" and "Boolean Matrix based algorithm" are compared and time taken by these algorithm are denoted in the graph. Extensive experiments are proposed for comparing the performance of RCS algorithm and Boolean Matrix based algorithm with the standard Apriori algorithm. This paper have shown that RCS algorithm and Boolean Matrix based algorithm have attempt to improved the algorithmic efficiency by reducing the repeated scan over the entire database and without producing candidate item sets.**

Keywords - Apriori algorithm, Association mining, CPU and I/O overhead, data mining, large size database, RCS algorithm

I. INTRODUCTION

Data mining is extracting the knowledge from large amount of data. 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. The aim of the paper is to implement the Apriori based algorithms (RCS and Boolean matrix) along with the comparative study of standard Apriori algorithm. The approach is to derive large patterns of data from the database. It involves identification of unique patterns and extract of different association rules from the database in supportive to the measures, support and confidence.

II. ASSOCIATION RULE PROBLEM

Association rule mining is one of the data mining techniques used to extract the itemsets which frequently appearing in the database. The value of minimum support and confidence are used to find the frequent patterns. It deals with two problems:

- One is to find those itemsets whose occurrences exceed a predefined threshold in the database; those itemsets are called frequent or large itemsets.
- The second problem is to get association rules from those massive itemsets with the constraints of minimal confidence.

I. SUPPORT

Support of an association rule is defined as the fraction of records that contain X ∪ Y to the total number of records in the database.

$$Support\ (XY) = \frac{Support\ count\ of\ (XY)}{Tot\ no.\ of\ trans.\ in\ D}$$

III. CONFIDENCE

Confidence(C) of an association rule is defined as the percentage of the number of transactions that contain X ∪ Y to the full range of number of records that contain X. Confidence is a measure of strength of the association rules, suppose the confidence of the association rule X⇒Y is 80%, it means that 80% of the transactions that contain X also contain Y together.

$$Confidence\ (X|Y) = \frac{Support\ (XY)}{Support\ (X)}$$

IV. IMPLEMENTATION

A. Apriori Algorithm

- The first pass of the algorithm counts item occurrences to see massive large 1-itemsets.
- This process is repeats until no new large 1 itemsets are identified.
- (k+1) length candidates itemsets are generated from length k large items.
- Candidate itemsets containing subset of length k that are not large are pruned.
- Support of each candidate itemset os counted by scanning the database.
- Eliminate candidate itemsets that are smaller than minimum support value.

B. *Reduced Candidate Set (RCS) algorithm*

- RCS algorithm uses dataset in the transposed form, so candidate set and frequent item set generation process will be changed as compare to Apriori algorithm.
- The candidate 2-itemset will be generated by performing dot-multiplication of rows of array as array consist of boolean values (i.e. the resultant cell will result in "1" if the corresponding cells of the respective rows have "1" otherwise "0" in the resultant cell).
- In this process a new array consisting of candidate 2-item sets are generated to get the higher order of item set, the above process between rows of array.

C. *Boolean Matrix algorithm*

- Scanning the transaction database D, building corresponding boolean matrix T. Find the maximum k on the leading diagonal by computing matrix T.
- Scanning each row of T, compute the number of elements whose value is bigger than or equal to k in every row, then find whether minimum support is tenable or not.
- If it is tenable, then the row is the possible row to which k-frequent item sets correspond and record the vector, and then continually scanning.
- If the scanning is over and of each row is smaller than minimum support, then find the sub-maximum k-1 on the leading diagonal from T, then repeat the above step until finding the right row.
- At last, analyze the vector and logical AND using any minimum support vectors, and then obtaining the frequent item sets that meet the minimum support.

V. COMPARISON BETWEEN APRIORI, RCS ALGORITHM AND BOOLEAN MATRIX ALGORITHM

A. *Apriori Algorithm*

Let following is the transactional database. Consider a database consisting of 5 transactions

ITEMSET TABLE

TID	Itemset
T100	i1,i2,i3
T101	i1,i2,i3,i4,i5
T102	i1,i3,i4
T103	i1,i3,i4,i5
T104	i1,i2,i3,i4

Suppose min. support count required is 2 (i.e. min_supp = 2/9 = 22 %)

Let minimum confidence required is 70%.

PASS 1

Itemset	Supp. Count
{i1}	5
{i2}	3
{i3}	5
{i4}	4
{i5}	2

C1

Itemset	Supp. Count
{i1}	5
{i2}	3
{i3}	5
{i4}	4
{i5}	2

L1

PASS 2

Itemset	Supp. Count
{i1,i2}	3
{i1,i3}	5
{i1,i4}	4
{i1,i5}	2
{i2,i3}	3
{i2,i4}	2
{i2,i5}	1
{i3,i4}	4
{i3,i5}	2
{i4,i5}	2

C2

Itemset	Supp. Count
{i1,i2}	3
{i1,i3}	5
{i1,i4}	4
{i1,i5}	2
{i2,i3}	3
{i2,i4}	2
{i3,i4}	4
{i3,i5}	2
{i4,i5}	2

L2

PASS 3

Itemset	Supp. Count
{i1,i2,i3}	3
{i1,i2,i4}	2
{i1,i2,i5}	1
{i2,i3,i4}	2
{i2,i3,i5}	1
{i3,i4,i5}	2

C3

Itemset	Supp. Count
{i1,i2,i3}	3
{i1,i2,i4}	2
{i2,i3,i4}	2
{i3,i4,i5}	2

L3

PASS 4

Itemset	Supp. Count
{i1,i2,i3,i4}	2
{i2,i3,i4,i5}	1
{i1,i3,i4,i5}	2

C4

Itemset	Supp. Count
{i1,i2,i3,i4}	2
{i1,i3,i4,i5	2

L4

PASS 5

We cannot generate the candidate frequent itemsets because the itemsets doesn't have beginning with the same 3 items frequent 4 itemsets.

B. RCS Algorithm

Following is the transpose of the transactional database used in Apriori algorithm. Assume the user-specified minimum support count is 2, and then the steps for generating all frequent itemsets in RCS algorithm.

TRANSPOSE OF TRANSACTIONAL DATABASE

i1	1	1	1	1	1
i2	1	1	0	0	1
i3	1	1	1	1	1
i4	0	1	1	1	1
i5	0	1	0	1	0

PASS 1

Itemset	Supp. Count
{i1}	5
{i2}	3
{i3}	5
{i4}	4
{i5}	2

C1

Itemset	Supp. Count
{i1}	5
{i2}	3
{i3}	5
{i4}	4
{i5}	2

L1

PASS 2

Itemset	Supp. Count
{i1,i2}	3
{i1,i3}	5
{i1,i4}	4
{i1,i5}	2
{i2,i3}	3
{i2,i4}	2
{i2,i5}	1
{i3,i4}	4
{i3,i5}	2
{i4,i5}	2

C2

Itemset	Supp. Count
{i1,i2}	3
{i1,i3}	5
{i1,i4}	4
{i1,i5}	2
{i2,i3}	3
{i2,i4}	2
{i3,i4}	4
{i3,i5}	2
{i4,i5}	2

L2

PASS 3

Itemset	Supp. Count
{i1,i2,i3}	3
{i1,i2,i4}	2
{i1,i2,i5}	1
{i2,i3,i4}	2
{i2,i3,i5}	1
{i3,i4,i5}	2

C3

Itemset	Supp. Count
{i1,i2,i3}	3
{i1,i2,i4}	2
{i2,i3,i4}	2
{i3,i4,i5}	2

L3

PASS 4

Itemset	Supp. Count
{i1,i2,i3,i4}	2
{i2,i3,i4,i5}	1
{i1,i3,i4,i5}	2

C4

Itemset	Supp. Count
{i1,i2,i3,i4}	2
{i1,i3,i4,i5	2

L4

PASS 5

We cannot generate the candidate frequent itemsets because the itemsets doesn't have beginning with the same 3 items frequent 4 itemsets.

A. *Boolean Matrix Algorithm*

Transform the database into Boolean matrix by using minimum support=2.

PASS 1

	i1	i2	i3	i4	i5
T 100	1	1	1	0	0
T101	1	1	1	1	1
T102	1	0	1	0	0
T103	1	0	1	1	1
T104	1	1	1	1	0

PASS 2

(i1^i2)(i1^i3)(i1^i4)(i1^i5)(i2^i3)(i2^i4)(i2^i5)(i3^i4)(i3^i5)(i4^i5)

T100	1	1	0	0	1	0	0	0	0	0
T101	1	1	1	1	1	1	1	1	1	1
T102	0	1	1	0	0	0	0	0	0	0
T103	0	1	1	1	0	0	0	1	1	1
T104	1	1	1	0	1	1	0	1	0	0

↓ Min. supp count=2

(i1^i2)(i1^i3)(i1^i4)(i1^i5)(i2^i3)(i2^i4)(i3^i4)(i3^i5)(i4^i5)

T100	1	1	0	0	1	0	0	0	0
T101	1	1	1	1	1	1	1	1	1
T102	0	1	1	0	0	0	0	0	0
T103	0	1	1	1	0	0	1	1	1
T104	1	1	1	0	1	1	1	0	0

PASS 3

	(i1^i2^i3)	(i1^i2^i4)	(i1^i2^i5)	(i2^i3^i4)	(i2^i3^i5)	(i3^i4^i5)
T100	1	0	0	0	0	0
T101	1	1	1	1	1	1
T102	0	0	0	0	0	0
T103	0	0	0	0	0	1
T104	1	1	0	1	0	0

↓ Min. supp count=2

	(i1^i2^i3)	(i1^i2^i4)	(i2^i3^i4)	(i3^i4^i5)
T100	1	0	0	0
T101	1	1	1	1
T102	0	0	0	0
T103	0	0	0	1
T104	1	1	1	0

PASS 4

	(i1^i2^i3^i4)	(i2^i3^i4^i5)	(i1^i3^i4^i5)
T100	0	0	0
T101	1	1	1
T102	0	0	0
T103	0	0	1
T104	1	0	0

↓ Min. supp count=2

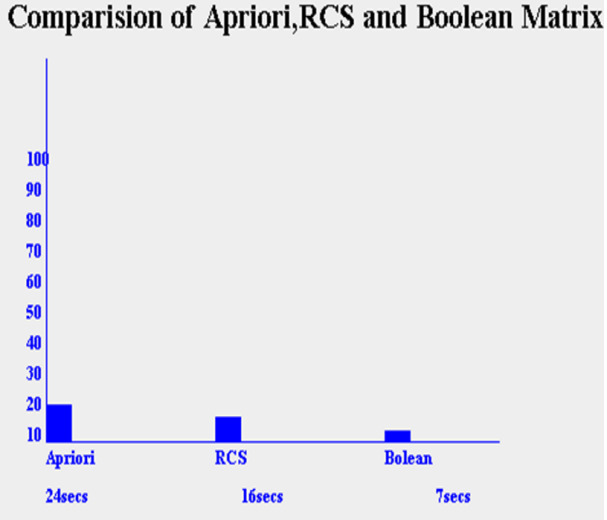
	(i1^i2^i3^i4)	(i1^i3^i4^i5)
T100	0	0
T101	1	1
T102	0	0
T103	0	1
T104	1	0

PASS 5

We cannot generate the candidate frequent itemsets because the itemsets doesn't have beginning with the same 3 items frequent 4 itemsets.

VI. EXPERIMENTAL EVALUATION

All the experiments are performed on a 2.10 GHz Pentium Intel core i5 laptop machine with 4.0 main memory, running on Windows 7 operating system. Apriori, RCS and Boolean Matrix algorithm were developed under the java compiler in the version 1.6 and taking input in the form of text files of dataset. The performance of Apriori, RCS, Boolean Matrix algorithm and these results are shown in the figure.



Comparison of Apriori, RCS and Boolean Matrix Algorithm on the basis of execution time.

VII. CONCLUSION AND FUTURE ENHANCEMENTS

The implemented algorithms are not only efficient but also fast for discovering association rules in large database. A necessary, contribution of this approach is that, it drastically reduces the I/O overhead related to Apriori algorithm. These algorithms will be useful for many real-life database mining scenarios where the data is stored in boolean form. The current problem is that these algorithms are only suitable for boolean dataset and needs future work to make it applicable to all kinds of datasets.

REFERENCES

1. Manoj Bahel, chhaya Duel "Analysis of Frequent Itemset generation process in Apriori and RCS (Reduced Candidate Set) Algorithm" Int. J. Advanced Networking and Applications 539 Volume: 02, Issue: 02, Pages:539-543 (2010).
2. Pratima Gautam , K. R. Pardasani "A Fast Algorithm for Mining Multilevel Association Rule Based on Boolean Matrix" Pratima Gautam et. al. / (IJCSE) International Journal on Computer Science and Engineering Vol. 02, No. 03, 2010, 746-752.
3. Neelu Khare , Neeru Adlakha and K. R. Pardasani "Karnaugh Map Model for Mining Association Rules in Large Databases," (IJCNS) International Journal of Computer and Network Security,Vol- 1, No. 1, October 2009.
4. R.S Thakur, R.C. Jain, K.R.Pardasani, "Fast Algorithm for Mining Multilevel Association Rule Mining," Journal of Computer Science, Vol-1, pp. 76-81, 2007.
5. Shweta. Kanwal Garg " Mining Efficient Association Rules Through Apriori Algorithm Using Attributes and Comparative Analysis of Various Association Rule Algorithms" Shweta et al., International Journal of Advanced Research in Computer Science and Software Engineering 3(6), June - 2013, pp. 306-312
6. Rachna Somkunwar, "A Study on Various Data Mining Approaches of Association Rules", In: proceeding of International Journal of Advanced Research in Cmputer science and Software Engineering, ISSN 2277-128X, Volume-2, Issue-9, Page-141-144, September-2012.
7. Partibha Parikh Dinesh Waghela, "Comparative Study of Association Rule Mining Algorithms", In: Proceeding of UNIASCIT, ISSN 2250-0987, Vol. 2, Issue 1, 2012.