IJREAS VOLUME 5, ISSUE 9 (September, 2015) (ISSN 2249-3905)
International Journal of Research in Engineering and Applied Sciences (IMPACT FACTOR – 5.981)

Optimized Apriori Algorithm Using Matrix Data Structure

Rajesh Singh[1],
Akshay Bhala[2],
Juilee Salunkhe[3],
Aanchal Jhirmiria[4]

Department of Computer Science & Engineering,
Thakur College of Engineering and Technology,
Mumbai University, Mumbai, India

Abstract— Association rule learning in data mining is a favored and well researched method for interesting relations between variables in large databases. Association rule of data mining involves picking out the unknown interdependence of the data and discovering the rules between those items .There are several algorithms for finding frequent item sets and mining comprehensive association rules from databases. In classical Apriori algorithm, algorithm may generate abundant number of candidate generations but traditional algorithms are not efficient. Every time Algorithm scans and judge about the candidate generation and frequency of item sets. This results into high frequency in querying, so amount of resources spent in terms of time or space is huge and decreases the efficiency of Classical Apriori Algorithm .To outcast these drawbacks we have proposed an optimized algorithm in this paper with an aim of minimizing the mundane and space complexities by reducing the database scans to one by generating impressed data structure bit matrix(bit_matrix) and by reducing superfluous computations for extracting regular item sets using top-down approach.

Keywords—*apriori algorithm; association; frequent item sets; outcast; mundane; superfluous*

I. INTRODUCTION

A. Basic Concepts

Today the amount of data is enormous and processing of data is an important factor to be considered. The computerization of most of the commercial activities and the advancements in technology has given rise to such huge amount of data. The ease of managing and availability of database systems and using such databases, has also contributed to the huge amount of data and repositories. Powerful tools and techniques are required for extracting information and knowledge from such repositories. As a result data mining has become an interesting research area. Data mining is process of mining useful, hidden and interesting information from the stored data. The process of data mining includes methods that mine different types of knowledge from databases. Knowledge that can be exposed includes association rules, classification rules, generalizations and summarizations, etc. Several business enterprises process large amount of data from various sources on daily basis. Considering an example of a supermarket where every customer carries out different transactions for different products and huge amount of customer purchase data is collected daily at checkout counters. Analyzing these transactions and relationships between different products help retailers to understand the purchasing behavior of their customers. Such nontrivial information can be used to strengthen variety of applications like marketing promotions, inventory management, etc. Association rule is based on discovering frequent item sets. Retail stores use these association rules in order to assist processes in marketing, advertising, predicting various faults in telecommunication network, inventory management. Considering an example table 1.Each row in this table corresponds to a transaction, which contains a unique identifier TID and a set of items bought by a given customer.
The following rule can be extracted from the data set shown in Table 1:

{Tea} → {Milk}

Table1. An example of market basket transactions

TID	Items
1	{Tea, Milk}
2	{ Tea, Diapers, Beer, Eggs}
3	{ Milk, Tea, Cola}
4	{ Tea, Milk, eggs}
5	{ Tea, Milk, Diapers, Cola}

The rule suggests that a sound relationship exist between Tea and Milk because many customers who buy Tea also buy Milk. These types of rules can be used by retailers to understand the association between two such products and customer purchasing behavior. Association analysis is also applicable to other application domains such as medical diagnosis, business management, Web mining, production control, bio-informatics and scientific data analysis. In diagnosis of a particular medical condition or disease, the association rules can help in analyzing those particular symptoms which are associated and relevant to that particular disease.

B. Itemset and Support Count

In association analysis, an item set is termed as a collection of zero or more items. If an item set contains n items, it is called n-item set. The null set is an item set that does not contain any items. The transaction width is defined as the number of items present in a transaction. A transaction tj is said to contain an item set X if X is a subset of tj. Support count is an important property of an item set which refers to the number of transactions that contain a particular item set.

C. Association Rule

An association rule is an implication expression of the form X → Y, where X and Y are disjoint item sets. Support determines the frequency of the application of association rule to a given data set whereas confidence determines how frequently items in Y appear in transactions that contain X.

D. Key Terms-Support and Confidence

The rule X ⇒ Y holds with support s if s% of transactions in D contains X ∪ Y. Rules that have as greater than a user-specified support is said to have minimum support. A low support rule is likely to be trivial because it may not contribute in promoting items.

Support = Occurrence / Total Support

Table1. Example of Support Measure

TID	Items
1	XYZ
2	WXY
3	YU
4	YUX
5	WX

Support results for the above table 1 are as follows:
Total Support = 5
Support {WX} = 2 / 5 = 40%
Support {YU} = 2 / 5 = 40%
Support {XYZ} = 1 / 5 = 20%

The rule X ⇒ Y holds with confidence c if c% of the transactions in D that contain X also contain Y. Rules that have a c greater than a user-specified confidence is said to have minimum confidence. Confidence provides an estimate of conditional probability of Y given X.

Confidence = Occurrence {Y} / Occurrence {X}

Table2. Example of Confidence Measure

TID	Items
1	PQR
2	PQS
3	QR
4	PR
5	QRS

Support results for the above table 2 are as follows:
Confidence {P⇒Q} = 2 / 3 = 66%
Confidence {Q⇒ R} = 3 / 4 = 75%
Confidence {PQ ⇒ R} = 1 / 2 = 50%

The aim of association mining for a given set of transaction D is to find all rules having-
Support >= minsupport threshold
Confidence >= minconfidence threshold

E. Frequent Item Set Generation

A grid structure can be used to enumerate the list of all possible item sets. In general, a data set that contains n items can potentially generate up to $2^n - 1$ frequent item sets, excluding the null set. Because n can be very huge in many pragmatic applications, the search space of item sets that need to be explored is exponentially large.

A brute-force approach for finding frequent item sets is to determine the support count for every candidate item set in the lattice structure. To do this, we need to compare each candidate against every transaction. If the candidate is contained in a transaction, its support count will be incremented. Such an approach can be very expensive because it requires O (NMw) comparisons, where N is the number of transactions, $M = 2^K - 1$ is the number of candidate item sets, and w is the maximum transaction width.

There are several ways to reduce the computational complexity of frequent item set generation.

1) Reduce the number of candidate item sets (M). The Apriori principle, described in the next section, is an effective way to eliminate some of the candidate item sets without counting their support values.

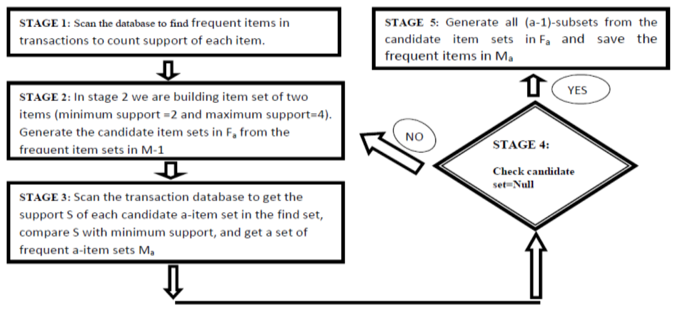
2) Reduce the number of comparisons. Instead of matching each candidate item set against every transaction, we can reduce the number of comparisons by using more advanced data structures, either to store the candidate item sets or to compress the data set.

II. APRIORI ALGORITHM

A. *Apriori Priniciple :*

Association rule learning is a popular and well researched method for discovering interesting relations between variables in large databases. On the concept of strong rules, association rules were introduced for discovering regularities between products in large-scale transaction data recorded by point-of-sale (POS) systems in supermarkets. For example, the rule {Bread, Spinach}→Sandwich found in the sales data of a supermarket would indicate that if a customer buys bread and spinach together, they are likely to also buy sandwich. Such information can be used as the basis for decisions about marketing activities such as, e.g., promotional pricing or product placements.

B. *Steps to perform Apriori Algorithm :*


STAGE 1: Scan the database to find frequent items in transactions to count support of each item.
STAGE 2: In stage 2 we are building item set of two items (minimum support =2 and maximum support=4). Generate the candidate item sets in Fa from the frequent item sets in M-1
STAGE 3: Scan the transaction database to get the support S of each candidate a-item set in the find set, compare S with minimum support, and get a set of frequent a-item sets Ma
STAGE 4: Check candidate set=Null
NO
YES
STAGE 5: Generate all (a-1)-subsets from the candidate item sets in Fa and save the frequent items in Ma

C. *Pseudocode for Apriori Algorithm :*

A frequent item set is an item set whose support is greater than some user-specified minimum support (denoted M_a, where *a* is the size of the item set)

A candidate item set is a potentially frequent item set (denoted F_a, where *a* is the size of the item set)

```
Process 1
    1. Generate the candidate item sets in F1
    2. Save the frequent item sets in M1
Process k
    1. Generate the candidate item sets in $F_a$ from the frequent item sets in M-1
        1. Join $M_a$-1 r with $M_a$-1s, as follows:
           insert into $F_a$
           select r.$item_1$, r.$item_2$, . . . , r.$item_{a-1}$, s.$item_{a-1}$
           from $M_{a-1}$ r, $M_{a-1}$ s
           where r.$item_1$ = s.$item_1$ , . . . r.$item_{a-2}$ = s.$item_{a-2}$ , r.$item_{a-1}$ < s.$item_{a-1}$
        2. Generate all (a-1)-subsets from the candidate item sets in $F_a$
        3. Prune all candidate item sets from $F_a$ where some (a-1)subset of the candidate item set is not in the frequent item set $M_{a-1}$
    2. Scan the transaction database to determine the support for each candidate item set in $F_a$
    3. Save the frequent item sets in $M_a$
```

D. *Example of Apriori Algorithm:*

Example:-Assume that we are having 5 transactions in a database i.e. D=5.

TID	ITEM_ID
1	A,B,C,E
2	C,D,E
3	C,E
4	A,C,D
5	B,D

Step 1-Find frequent items in above transactions to count support of each item.

TID	ITEM_ID	SUP_COUNT
1	A	2
2	B	2
3	C	4
4	D	3
5	E	3

Step 2-Join the table with itself.
In step 2 we are building item set of two items. And these item set can be counted with the original table. Note – In step 1 we are assuming minimum support count=2 and maximum support count =4. So, all items are greater than 2 .As a result no pruning is required.

TID	ITEM_SET	SUP_COUNT
1	AB	1
2	AC	2
3	AD	1
4	AE	1
5	BC	1
6	BD	1
7	BE	1
8	CD	2
9	CE	3
10	DE	1

Step 3- In this step we are pruning the above table on the basis of association rule which pursue equal or more than 2 support counts. But above table contains many item set having less than 2 support counts. So here pruning is required. After pruning the resultant table will be as

TID	ITEM_SET	SUP_COUNT
1	AC	2
2	CD	2
3	CE	3

Step 4- Join the table with itself
If we compare with support of the above table's item set with minimum support (i.e. 2) then none of the transaction sets are qualified. The result of applying APRIORI algorithm on above item sets with minimum support=2 .So, we get 3 frequent item sets as {A, C}, {C, D} and {C, E}.

TID	ITEM_SET	SUP_COUNT
1	ACD	1
2	CDE	1

III. PROPOSED METHOD FOR ASSOCIATION MINING RULES

A. *Basic Concept :*

In this improved algorithm a compressed data structure i.e., bit_matrix is constructed and then this compressed data structure is used later on to generate regular item sets. This algorithm makes use of top-down approach to find out the regular item sets from largest regular item set to smallest regular item set.

B. *Phases of Proposed Algorithm:*

Following are the two phases in which the computation of this improved algorithm takes place:

1. In this phase a matrix named bit_matrix is constructed for the given database which includes all the transactions. The column in this newly constructed matrix denotes the entities or the items in the given database of transactions and each row in the bit_matrix denotes a transaction that takes place. In this newly constructed bit_matrix the value that each of the cell can take can be 1 or 0 which makes it evident that the transaction is taking place. If the value 1 is present in any of the cell corresponding to a given row then the item is present in that particular transaction and if at all the value is 0 then the corresponding item is not present in that particular row. Two more columns cnt and RT_cnt (Redundant Transaction Counter) are added in the bit_matrix table. The cnt column gives us the size of row (the addition of total no of 1's in that particular row) and it eliminates those columns whose addition is not equal or greater than predefined min_support value and then update

cnt column. If we find that a row is duplicated in database then it is depicted by the value in the RT_cnt(Redundant Transaction Counter) column and it then deletes all the duplicate transactions which are not really required and if row does not contain any duplicate transaction then RT_cnt (Redundant Transaction Counter) column is set to 1. Then the bit_matrix table is rearranged in descending order based on cnt column. This is the desired compressed data structure and here the phase 1 of our improved algorithm completes.

2. In the second phase of the computation we generate regular item sets directly from bit_matrix. We need to select first row from bit_matrix table and compare its cnt value with next row cnt respectively. If we find that the next row cnt value is more or equal to the processing row cnt value then we can apply AND operation among the rows and if we find that result is same to the processing row item set structure then we increment the cnt value of support of processing row item set by one and t continue this procedure of matching and AND same process keeps going on through rest of the rows in bit_matrix and then check the value of total support. If the value of total support is greater or equal to predefined min_support cnt then we extract that extract the item set and its subsets and shift them to array list of frequently occurring items. The same procedure will be repeated for rest of the transactions in bit_matrix until all rows are not checked.

C. *Advantages of Proposed Algorithm:*

The major advantage of this improved algorithm is that it reduces the number of comparisons which are done to mine largest item set for duplicate transactions and another major advantage is that once the largest item set is found then we can search for its subsets and move it to the array list of frequently occurring items. Now, while searching for next largest item set it first makes a note of whether the transaction under processing is previously present in the array list of frequently occurring items because of prior largest item set and its subsets, if item set is already in the array list of frequently occurring items, it avoids number of comparisons needed to calculate the support count of item set. Thus less number of scans and time is needed to extract the regular item set and also it requires comparatively less memory.

D. *Working of These Proposed Algorithm Using Example :*

Let us consider the implementation of this improvised algorithm through a sample transactional data as given below. Table-1 shows a transactional database consist of 4 transactions. Set the minimum support counts as 4 (min_support=4).

TID	Items
100	1,3,4,6
200	2,3,5
300	1,2,3,5,6
400	2,3,6
500	1,2,3,4
600	1,3,4,5
700	3,4,6

a) Phase-1

Step1- In this phase we need to scan the transactional database and convert it into desired compressed data structure that is bit_matrix. In this matrix each row represents one transaction and column represents distinct items in whole transactional database. The column cnt represents the number of 1 bit in that particular row. In bit_matrix value entered will be 1, if item is present in the corresponding row and 0, if item is not present in the corresponding row.

TID	I1	I2	I3	I4	cnt	RT_cnt
500	1	1	1	1	4	1
100	1	0	1	1	3	2
300	1	1	1	0	3	1
200	0	1	1	0	2	2
700	0	0	1	1	2	1

Step2- The bit_matrix is then rearranged in descending order based on cnt column after removing all those columns whose sum is not equal or greater than required minimum support value. Here min_support is 4, hence remove columns for items 5 and 6 and update cnt column and also merge the duplicate valued rows in RT_cnt column of bit_matrix to reduce the computations having redundant rows for finding regular item set.

b) Phase-2

Step1- Now select first row TID 500 and extract its item set {1,2,3,4} and calculate its support count in bit_matrix table using AND operation with rows having count value equal or greater than count value of that row. If AND operation results in same item set structure as processing row's item set structure, then we need to increment its support count value. After complete AND operation, check value of support count of item set, if it is equal or greater than required min_support than it is frequent and we can move item set with its subset into array list of frequently occurring items and move to next row. Here item set support is less than required min_support and hence it is not regular, so move to next row.

Step2- select next row TID 100 and extract its item set {1,3,4}.first check item set in frequent array list , if it is present in frequent array list , then it is regular , no need of AND operation with other rows to calculate its support count. But TID 100 is not present in frequent array list. So do AND operation with rows TID 500 and 300.after AND operation its support count is 3. Here, support of the item set is less than required min_support and hence it is not regular, so move to next row.

Step3- select next row TID 300 and extract its item set {1,2,3}.first check item set in frequent array list , if it is present in frequent array list , then it is regular , no need of AND operation with other rows to calculate its support count. But TID 300 is not present in frequent array list. So do AND operation with rows TID 500 and 100.after AND operation its support count is 2. Here, support of the item set is less than required min_support and hence it is not regular, so move to next row.

Step4- select next row TID 200 and extract its item set {2,
3}. After AND operation its support count is 4, hence it is regular, move item set with its subset into frequent array list. So here array list of frequently occurring item sets is {(2, 3), (2) AND (3)}. Then, move to next row.

Step5- select last row TID 700 and extract its item set {3,4}. first check item set in frequent array list , if it is present in frequent array list , then it is regular and there is no need of AND operation with other rows to calculate its support count. But TID 700 is not present in frequent array list, so we need to perform AND operation with rows TID-100,200,300 and 500 after AND operation its support count is 4, hence it is regular, move item set with its subset into array list of frequently occurring items. So here, array list of frequently occurring item sets is {(3, 4), (3) AND (4)}.

Frequent array list :{(2,3),(3,4),(2),(3) and (4)} extracted in less time by avoiding unwanted and vain comparisons as per the improvised algorithm.

IV. COMPARISON

Attributes	Proposed Algorithm Matrix Data Structure	Classical Apriori Algorithm	Graph Based Apriori Algorithm	Transaction Reduction Apriori Algorithm
New Technique Used	Matrix which stores data in binary format	Single items are stored repeatedly	Graph is constructed which stores data in binary format	Cuts down the unnecessary and redundant generation of items
Number Of Scans	1	More than 1	1	More than 1
Storage Structure Used	2-D Matrix Array	Normal Database	Tree Like Structure	Normal Database
Support Count & Execution Time	3% & 4.2 sec	3% & 10.3 sec	3% & 5.5sec	3% & 7.9sec
Advantages	Fast execution & low space required	-	Faster than classical Apriori algorithm	Avoids Redundant Items
Disadvantages	-	Redundancy & take more time	Tree requires more space	Takes more time for execution

V. CONCLUSION

Classical Apriori algorithm suffers from two limitations –The number of candidate item sets generation is large and database is scanned several times. In this paper, an optimized approach for mining regular item set using bit matrix is been proposed. This proposed algorithm needs only single scan of whole transactional database. It increases the efficiency of algorithm by reducing the redundant scanning of database as well as memory space.

REFERENCES

[1] Darshan M. Tank "Improved Apriori Algorithm for Mining Association Rules" I.J. Information Technology and Computer Science, 2014, 07, 15-23 Published Online June 2014 in MECS (http://www.mecs-press.org/) DOI: 110.5815/ijitcs.2014.07.03

[2] Jaishree Singh, Hari Ram, Dr. J.S. Sodhi "Improving Efficiency of Apriori Algorithm Using Transaction Reduction" International Journal of Scientific and Research Publications, Volume 3, Issue 1, January 2013 1 ISSN 2250-3153

[3] Manju "Graph Based Approach for Finding Frequent Itemsets to Discover Association Rules" International Journal of Innovations in Engineering and Technology (IJIET)

[4] Shilpi Singla and Arun Malik "Survey on various improved Apriori Algorithms",International Journal of Advanced Research in Computer and Communication Engineering Vol 3 ,Issue 11 ,November 2014 ISSN 2278-1021

[5] Priyanka and Er.Vinod Kumar Sharma "Apriori Algorithm for mining frequent item sets-A Review",International Journal of Computer Application and Engineering Technology,Vol 3,Issue 3,July 2014.Pp.232-236