Improved Aprori Algorithm Based on bottom up approach using Probability and Matrix

Sunil Kumar S[1], Shyam Karanth S[2], Akshay K C[3], Ananth Prabhu[4],Bharathraj Kumar M[5]

**[1] Computer Science, Visveshvarya Technology University, Sahyadri College of Engineering and Management
Mangalore, Karnataka ZIP/Zone, India**

**[2] Computer Science, Manipal University, Manipal Institute of Technology
Manipal, Karnataka 576104, India**

**[3] Information and Communication Technology, Manipal University, Manipal Institute of Technology
Manipal, Karnataka 576104, India**

**[4] Computer Science, Visveshvarya Technology University, Sahyadri College of Engineering and Management
Mangalore, Karnataka 575007, India**

**[5] Computer Science, Visveshvarya Technology University, Shree Devi Institute of Technology
Mangalore, Karnataka 575007, India**

Abstract

Knowledge Discovery through mining association rule among data from a large database is the one of key area of research. The first proposed Algorithm apriori is used to mine frequent items from the large database which leads to mine Association Rule between the data for discovering the Knowledge from the large database. Due to the limitation and complexity of Apriori algorithm, lot of research is underway for discovering new algorithms with a motive of minimizing the time and number of database scans for Knowledge Discovery through mining Association Rule among data from a large database. This paper propose one such kind of new algorithm which takes less number of scans to mining the frequent items from the large database which leads to mine the association rule between the data.

Keywords: *Apriori, Confidence, Support, matrix, AND operation, frequent Item Set,Probablity*

1. Introduction

The challenging task is to extracting useful information from the large collection of data in Dataware house and data base. Around the world lot of research is underway to discover the knowledge from the large collection of data in data warehouse. In this process many algorithms has been proposed to identify the associations between the data in the database, leads to mine the association rule among the data. Association rules are used for knowledge discovery and to take useful managerial decision in the organization based on the results of associations among data stepping toward to make a smarter system. In this regard, the first algorithm Apriori [1] was proposed in the year 1994 by Agarwal and Srikanth to mine the frequent item set. Time constraint and efficiency of algorithms leads to lot of research in the area of o algorithm to build efficient algorithm which takes less time and few number of database scans to mine frequent Item set and association rule.

The first most famous algorithm Apriori is proposed in the year 1994 by Agarwal and Srikanth to mine the frequent item set, which leads to mine association rule. Apriori algorithm suffers from many numbers of database scans required to identify the frequent items set and take more time if the database size is increased. Then Partition algorithm was proposed, where instead scanning entire database and searching for frequent item set only the part of the data base is scanned at a time by making different partition of large database. Pincer Search Algorithm was proposed with a new methodology of Bottom up Approach with a proper make use of the Downward closure property leads to identify the frequent Item set in less time with minimum number of database scans.

FP-Tree algorithm came into picture in the year 1999 by which reduces number of data base scan required by constructing FP Tree Structure but it suffers from the time required to construct a FP-Tree structure for large database. The increase in the size of the FP Tree with respect to the growth of database leads to difficulties in constructing, search and insert operation on large FP Tree.This paper propose a new improved apriori algorithm with a main motive of reducing time and number of scans required to identify the frequent Item set and association rule among data. The new algorithm Improved Apriori Algorithm using Probability Measure and Matrix incorporates the concept of probability, Matrix and Bitwise AND operation to minimize The time and number of scan. the improved Apriori Algorithm using Probability measure and Matrix is best illustrated with following example.

2. IMPROVED APRORI ALGORITHM BASED ON BOTTOM UP AND TOP DOWN APPROACH USING THE MATRIX AND REDUCED TRANSACTIONS

The proposed algorithm Improved Apriori algorithm based on Bottom up approach using probability and Matrix used to identify frequent item set. In proposed algorithm probability measure [2]of each item occurrence to total number of transaction is used along with the Bottom up Approach to find the frequent item set from largest frequent Item set to smallest frequent item set. Algorithm works in 2 phases, Probability Matrix Generation and Bottom Up approach to mine frequent items set.

In first phase initial matrix M1 will be generated for the given data set. Rows in matrix represent transaction. Columns in matrix represent items. Each cell will have the values either 0 or 1 for representing presence of items in the transaction. Entry value 1 indicates the corresponding item is present in transaction and value 0 indicates the corresponding item is not present in transaction. Initial Matrix leads to the generation of Probability matrix M2, where each entry value of 1 is replaced by the probability of occurrence of corresponding item to the total number of transactions and inserting two more columns to the probability matrix to hold total probability and count of elements in each row respectively. The Probability Matrix will be rearranged as per the descending order of Total probability leads to the generation of Sorted Probability Matrix M3. In second phase, Non zero entries in Sorted Probability Matrix will be replaced by the value of 1 leads to the generation of Sorted Probability Matrix M4.

Select first transaction from M4 and compare its total probability and count with next transaction total probability and count respectively. If the next transaction total probability and count greater than the next transaction probability then do the BITWISE AND operation between the transaction, if the resultant is equal to first transaction structure then increase the support count of first transaction item set by 1. Continue this process of Comparing and Bitwise AND operation with remaining transaction until it satisfies the condition of First transaction total probability and count is less than or equal to next transaction and checks the total support count if its greater than the required support count extract the item set of that transaction and all its subset and move it to frequent Item set. The same process will be repeated for remaining transaction until it finds unseen transaction in the given data set.

The Major advantage of this algorithm is it avoids comparison of currently chosen transaction with other transaction to mine the frequent item set if the total probability or count of number of elements of the other transactions on which comparison needs to be done is lesser than the chosen transaction. Since the lesser probability in next transaction indicates that transaction does not contain the all items or item set of transaction under scanning process. It reduces the number of comparison required to mine the largest frequent items set. Another beauty of this algorithm is once the largest frequent item set is found all its subsets will be identified and moved to frequent items set. While considering next transaction to find next largest frequent item set first it checks whether item set of transaction under scanning process is already in frequent items set because of previous largest frequent item set and its subset, if its already in frequent item set, it avoids another set of comparison required to find the support of item set leads to reduction in number of scans and time required to mine the frequent item set.

2.1 Illustration

Consider the given sample dataset.

Table I: Simple Datasets

Transcation number	*Items Procured*
B1	I1, I2, I3
B2	I4, I5
B3	I2, I3, I5
B4	I4, I5
B5	I2, I3, I6
B6	I1, I2, I3, I5
B7	I6, I7
B8	I2, I3, I5
B9	I5
B10	I1, I2, I3, I5

Scan the database and generate Initial Matrix representation M1 as shown in TABLE II below, each row corresponding to one transaction and column corresponding to Items respectively. Containing cell value as either 0 or 1, 1 if the items present in the Corresponding Transaction 0 If the items doesn't present in the Transaction.

2.2 Algorithm 1 – Improved Apriori Algorithm

Input : Sample Data Set, Support
Output : Frequent Item Set
T – Transaction sets, **n** – number of transactions, **t** – transactions

Scan the database and generate Initial Matrix M1, rows corresponding to transactions and columns corresponding to Items respectively. Containing cell value as either 0 or 1, for indicating the presence of item in respective transaction.

Generate the probability matrix M2 by replacing all Non zero values of items to their corresponding probability of occurrence to total number of transactions. Insert two more columns to hold total probability and count of number of elements in each row respectively.

Generate the sorted probability matrix M3 by rearranging the transaction based on the descending order of total probability.

Replace all non-zero values to 1 in sorted probability matrix M4 by non-volatile the order of transaction.

```
**Initialize** FIS=NULL supportcount = 0
  for all t € T do
    for all i=1 to n do
      IS=extract(ti)
      if (IS – FIS != NULL ) then
        for all j=i+1 to n do
          if(prob(tj)≥prob(ti)ANDcount(tj )≥count(ti))then
            Res=ti Bitwise AND tj
          if (res==ti) then
            Supportcount++
          end if
        end if
      end for
      if (supportcount>=support) then
        Res1=subset(IS)
        FIS=FIS U IS U res1
      end if
    end if
  end for
end for
Output : Frequent Item Set.
```

Table II: Matrix M1

Transaction	I1	I2	I3	I4	I5	I6
B1	1	1	1	0	0	0
B2	0	0	0	1	1	0
B3	0	1	1	0	1	0
B4	0	0	0	1	1	0
B5	0	1	1	0	0	1
B6	1	1	1	0	1	0
B7	0	0	0	0	1	0
B8	0	1	1	0	1	0
B9	0	0	0	0	1	0
B10	0	0	0	0	1	0

Generate the Probability Matrix M2 by taking a probability of occurrence of each item with total number of transactions. Replace the entry with a value of 1 of items with its corresponding probability of occurrence to total number of transactions. Find the total probability in each row and insert two more column to hold the total probability of each row and number of items present in each transaction respectively as shown below TABLE III.

Table III: Matrix M2

Transaction	I1	I2	I3	I4	I5	I6	TotProb	Count
B1	0.3	0.6	0.6	0	0	0	1.5	3
B2	0	0	0	0.2	0.7	0	0.9	2
B3	0	0.6	0.6	0	0.7	0	1.9	3
B4	0	0	0	0.2	0.7	0	0.9	2
B5	0	0.6	0.6	0	0	0.2	1.4	3
B6	0.3	0.6	0.6	0	0.7	0	2.2	4
B7	0	0	0	0	0.7	0.2	0.9	2
B8	0	0.6	0.6	0	0.7	0	1.9	3
B9	0	0	0	0	0.7	0	0.7	1
B10	0.3	0.6	0.6	0	0.7	0	2.2	4

In Matrix M2, TABLE III, sort the total probability column in descending order and rearrange the Matrix M2 as per the descending order of total probability as shown in TABLE IV Matrix M3.

Table IV: Matrix M3

Transaction	I1	I2	I3	I4	I5	I6	TotProb	Count
B6	0.3	0.6	0.6	0	0.7	0	2.2	4
B10	0.3	0.6	0.6	0	0.7	0	2.2	4
B3	0	0.6	0.6	0	0.7	0	1.9	3
B8	0	0.6	0.6	0	0.7	0	1.9	3
B1	0.3	0.6	0.6	0	0	0	1.5	3
B5	0	0.6	0.6	0	0	0.2	1.4	3
B2	0	0	0	0.2	0.7	0	0.9	2
B4	0	0	0	0.2	0.7	0	0.9	2
B7	0	0	0	0	0.7	0.2	0.9	2
B9	0	0	0	0	0.7	0	0.7	1

Generate Matrix M4 by replacing each Non-zero entry in M3 to 1 by maintaining the order of transaction as like M4 as shown TABLE V.

Table V: Matrix M3

Transaction	I1	I2	I3	I4	I5	I6	TotProb	Count
B6	1	1	1	0	1	0	2.2	4
B10	1	1	1	0	1	0	2.2	4
B3	0	1	1	0	1	0	1.9	3
B8	0	1	1	0	1	0	1.9	3
B1	1	1	1	0	0	0	1.5	3
B5	0	1	1	0	0	1	1.4	3
B2	0	0	0	1	1	0	0.9	2
B4	0	0	0	1	1	0	0.9	2
B7	0	0	0	0	1	1	0.9	2
B9	0	0	0	0	1	0	0.7	1

Select first transaction B6 and extract B6 transaction Item set {I1; I2; I3; I5}. Compare B6 transaction total probability and count with next transaction B10's total probability and count i.e

Probability (B6) >=Probability (B10) And count (B6) >= count (B10)

If the above condition satisfies, find the support of largest Candidate Frequent Item set in a given data set {I1; I2; I3; I5} in B10 by doing the Bitwise AND operation between B6 and B10. If the result of the Bitwise AND operation resembles the B6 Transaction, increment the support count of Largest Candidate Frequent Item set{I1; I2; I3; I5} and checks the support count meets the min support required to judge Item Set as Frequent Item Set. If Largest Candidate Item Set {I1; I2; I3; I5} have the min support move it to the Frequent Items Set and find all its Subset and move it to the Frequent Item set.

In the above case result of B6 Bitwise AND B10 resembles B6 transaction, conclude that largest candidate Item set {I1; I2; I3; I5} gain one support count, which conclues item set {I1; I2; I3; I5} presence on two transactions B6 and B10. It meets the min support of 2 and moved to Frequent Item set along with all its subsets.

Frequent Item Set : {{I1,I2,I3,I5}, {I1,I2,I3}, {I1,I2,I5}, {I2,I3,I5},{I1,I3,I5}, {I1,I2}, {I2,I3}, {I1,I3}, {I1,I5}, {I3,I5}, {I2,I5}, {I1}, {I2}, {I3}, {I5}}

Select Next Transaction B10 and extracts B10 transaction Items Set {I1,I2,I3,I5}. Check whether the item set is already present in Frequent Items set, if Item set already present in the Frequent Items Set, it do not required scan of the database to check its support once again. algorithm move on to scan the next transaction item set.

Select Next Transaction B3 and extract B3 transaction Item set {I2,I3,I5}. It is already present in the Frequent Item set, which avoids scan of data base to check its support leads to extract Next Item set in Next Transaction.

Select Next Transaction B8 and extract B8 transaction Item set {I2,I3,I5} which is already present in the Frequent Items Set avoid the comparison required to find its support in the data base. Move on to scan the next transaction for next item set. Select Next Transaction B1 and extract B1 transaction Item set {I1,I2,I3} which is already present in the Frequent Items Set avoid the comparison required to find its support in the data base. Move on to next transaction for next item set.

Select Next Transaction B5 and extract B5 transaction Item set {I2,I3,I6} which is Not present in the Frequent Items Set needs to be checked. In this process it needs not check its support in the transaction which is having their total probability and count less than the B5 transaction Total Probability and count. Hence B5 transaction Item set {I2,I3,I6} needs to check its support only in B6,B10,B3,B8,B1 transactions and skipping the scanning operation on remaining transaction.

None of the transactions B6,B10,B3,B8,B1 Bitwise AND operation with B5 result in B5, which concludes that Item set of B5 {I2,I3,I6} is not frequent.

Select Next transaction B2 for Next item Set {I4,I5}. Item Set {I4,I5} is not present in Frequent Item set leads to search of its support in transactions B6,B10,B3,B8,B1,B5,B4,B7 only, which are having total probability and count greater than the total probability and count of B2.

The B2 Bitwise AND B4 results in B2 makes the support count of Item set {I4,I5} to 2, which leads to conclude and move the Items set {I4,I5} along with all its subset to Frequent Item set.

Frequent Item set : {{I1,I2,I3,I5}, {I1,I2,I3}, {I1,I2,I5}, {I2,I3,I5}, {I1,I3,I5}, {I1,I2}, {I2,I3}, {I1,I3}, {I1,I5}, {I3,I5}, {I2,I5}, {I4,I5}, {I1}, {I2}, {I3},{I4}, {I5}}

Loop continues to select next transaction B4 and extracts B4 transaction Item set {I4,I5} but which is already present in the frequent item set avoid the scan of data base to check its support.

Loop continues to Select Next transaction B7 for Next item Set {I5,I6}. Item Set {I5,I6} is not present in Frequent Item set leads to search of its support in transactions B6,B10,B3,B8,B1,B5,B2,B4 only, which are having total probability and count greater than the total probability and count of B7.

None of the transactions B6,B10,B3,B8,B1,B5,B2,B4 Bitwise AND operation with B7 result in B7, which concludes that Item set of B7 {I5,I6} is not frequent.

Loop continues to select next transaction B9 and extracts B9 transaction Item set {I5} but which is already present in the frequent item set avoid the scan of data base to check its support. The final Frequent Item Set will be

Frequent Item set : {{I1,I2,I3,I5}, {I1,I2,I3}, {I1,I2,I5}, {I2,I3,I5}, {I1,I3,I5}, {I1,I2}, {I2,I3}, {I1,I3}, {I1,I5g}, {I3,I5}, {I2,I5}, {I4,I5}, {I1}, {I2}, {I3}, {I4}, {I5}} is generated with a less time by avoiding unnecessary comparison as per our new algorithm.

3. PERFORMANCE ANALYSIS

To analyses the relative performance of the Improved Aprori Algorithm Based on bottom up and top down approach using the matrix and reducing transactions we take Apriori and Apriori algorithms and compare each other. We use a small part data from real store database stored 10000 transactions. Figure 1 demonstrates the relative performance of these three algorithms. Five experiments are carried out accomplished using the same database with different minimum support factors. The experiment is in WindowsXP Professional operating system, CPU with Intel (R) 2.93GHz, memory with 1GB, the algorithm language used is java.

Experiments results show that the time needed IApriori algorithm is less than Apriori algorithm under the same support condition.And time needed for Improved Aprori Algorithm Based on bottom up and top down approach using the matrix and reducing transactions is much lesser than IApriori algorithm. So we can have the conclusion that the proposed algorithm outperforms the Apriori and IAprori algorithm in computational time.

4. CONCLUSION

The paper addresses the importance of knowledge mining from a large data set and overview of existing algorithm and its flaws and innovative solution with a new algorithm for data mining from the large data set. As seen in the observation and analysis new proposed algorithm performs much better than the existing Apriori. the relative performance of the proposed algorithm using probability and matrix under different minsupport specify its excellence and features. the improved algorithm can be utilized in many areas such as medical, image processing and database and ERP etc with a reduced time and space complexity requirements.

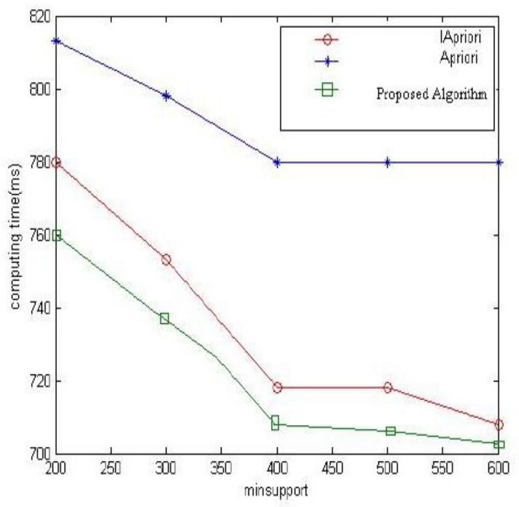

Fig. 1. Relative performance under different minsupport

REFERENCES

[1] Agrawal R, Imielinski T, Swami A.Mining association rules between sets of items in large databases[C]. In: Proc. of the l993ACM on Management of Data, Washington, D.C, May 1993. 207-216

[2] Gao Hongbin, Pan Gu, Huang Yiming. A improved Apriori algorithm based on the character of the frequent itemsets[J]. Computer engineering and design. 2007, 28(10):2273-2275.

[3] Li Xiaohong, Shang Jin. A new improved Apriori algorithm[J]. Computer science. 2007, 34(4):196-198

[4] Tong Qiang, Zhou Yuanchun, Wu Kaichao, Yan Baoping. A quantitative association rules mining algorithm[J]. Computer engineering. 2007 33(10):34-35

[5] Feng WANG , Yong-hua LI . An improved Apriori algorithm based on the matrix. In Proc. Of 2008 International Seminar on Future BioMedical Information Engineering

[6] Yiwu Xie, Yutong Li, Chunli Wang, Mingyu Lu . The Optimization and Improvement of the Apriori Algorithm . In proc. Of 2008 International Workshop on Education Technology and Training 2008 International Workshop on Geoscience and Remote Sensing.

Sunil Kumar S is a Asst Professor in Dept of Computer Science & Engineering, Sahyadri College of Engineering & Management, VTU. His research interest areas include Database, Data Mining, Mathematical Logic, Alloy Specification Language, Digital Image Processing and computer network and security.

Shyam Karanth, received his Bachelor's degree in Computer Science in 2006 and Master Degree in Computer Science & Engineering in 2010 from Manipal University. Currently he is working as Assistant Professor in MIT, Manipal University. His area of interests are Data Mining, Database, Web Services Network Security.

Akshay KC, received his Bachelor's degree in Information Science in 2008 and perusing MTech in Software Engineering from Manipal University. Currently he is working as Assistant Professor in MIT, Manipal University. His area of interests are Software engineering, Object Oriented Design, Software Architecture, Data Mining.

Ananth Prabhu G, holds a Bachelor's degree in Computer Science & Engineering, Master Degree in Computer Science & Engineering and Master in Business Administration from Manipal University. Currently he is working as Associate Professor. His area of interests are Data Mining, High Performace computing and Digital image processing.

Bharatraj Kumar M, holds a Bachelor's degree in Computer Science & Engineering, and Master Degree in Computer Science & Engineering. Currently he is working as Assistant Professor. His area of interests are Data Mining, Database, Parallel computing and Digital image processing.