Volume 5, Issue 7, July 2015 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

An Improved Apriori Algorithm for Large Data Set

Suruchi Kannoujia*
Computer Science and Engineering Department, Kanpur Institute of Technology (UPTU), Kanpur, Uttar Pradesh, India

Abstract— In recent years, there has been a huge accumulation of data. The amount of data collected is said to be almost doubled every 9 months. Seeking knowledge from massive data is one of the most desired attributes of Data Mining. Several techniques have been evolved in order to retrieve the interesting patterns by data mining. One of them is Apriori Algorithm, which scans the database several times before pointing out the frequent patterns. But its drawback is that the time and storage space of this algorithm is very high because of repetitive scanning of database. In this paper ,my approach focuses on removing this drawback by scanning individual transaction at a time from a given transaction set and generate all the subsets in the initial stage ,therefore the given set is not used again for further scanning, thereby it reduces execution time and storage space requirements. My algorithm scans the database only once, and produces the frequent patterns in almost constant time.

Keywords—DataMining, Apriori , Support CounTtransaction,Datasett, Frequent Items,

I. INTRODUCTION

Nowadays, most of the varieties of the information are stored electronically. Data mining or knowledge discovery in databases (KDD) is a collection of exploration techniques based on advanced analytical methods and tools for handling a large amount of information. The techniques will find interesting patterns that may help a firm/company in forecasting and understanding its business patterns more efficiently. In the last decade, there has been a gigantic growth in the storage and generation of electronic data. According to a survey in 2010, the number of e-mails sent in a day worldwide was calculated to be a staggering 294 billion. This shows that the e-mail servers have to store millions of e-mail id's which need a large amount of separate space on servers .Traditional database systems are often designed for running on daily basis in an organisation and are called Online Transaction Processing (OLTP) systems. These systems are designed to capture business transactions online and are optimized for high throughput, a high level of availability and in some cases high security of information [1].In unfavourable drug reaction surveillance, the Uppsala Monitoring Centre has, since 1998, used data mining methods to routinely screen for reporting patterns indicative of emerging drug safety issues in the WHO global database of 4.6 million suspected unfavourable drug reaction incidents. Recently, similar procedures have been evolved to mine large collections of electronic health records for temporary patterns associating drug prescriptions to medical diagnoses [2].Many systems and processes of human activity have become increasingly dependent on assembled, stored and managed information. Like, the use of debit/credit cards is growing rapidly. In a survey of 10 major countries, the total number of cards was over five billion and it keeps on growing. Also, the number of mobile phones has grown severely in developing countries like India and China. It has been reported that over 100 million phones were sold in India in 2010 and more than 200 million phones are likely to be sold by 2014. If we were to assume only 10 transactions per credit card and only 10 calls per year per mobile phone in India, the credit card and mobile companies would be collecting billions of transactions each year and as noted above the credit and debit cards and mobile phone numbers are growing rapidly in India[3].

II. DATA MINING

Data mining refers to extracting or "mining" knowledge from large amounts of data .Data mining is the collection of different methods and techniques for adequate discovery of understandable and interesting patterns in huge databases .The term is actually a misnomer. Remember that the mining of gold from rocks or sand is referred to as a gold mining rather than rock or sand mining. Thus, data mining is also referred to as "**Knowledge Discovery from Data**" (KDD)[4].As given in the following figure, we can see the steps involved in the process of KDD or data mining. Here first require the data, which are obtained by various databases, data warehouses, the World Wide Web or the other information repositories. Then it goes through the process of **data cleaning** in which noise and inconsistencies of the data are removed. Multiple data sources are combined in the process of **data integration**. Further, the useful data is obtained from the database by **data selection** .Then, based on the user's request, the **database servers** fetch the appropriate data. After that, there is the **data mining engine**, which is responsible for various data mining activities such as associations, predictions, correlations, classifications, clustering, and outlier analysis. We have a knowledge base that is the store house of knowledge that guides us in our pursuit of interesting patterns. The **pattern evaluation** module sets the measures and thresholds of interestingness. It filters the interesting patterns from the rest of the data by using various techniques through which, reports are generated, results are visualized and explained for the end users.

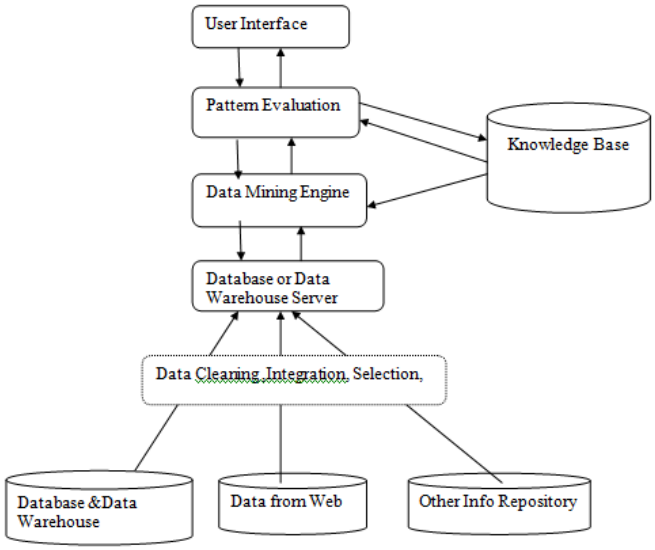


Figure 1 Architecture of Typical Datamining System

III. APRIORI ALGORITHM AND ITS DRAWBACKS

To generate Frequent patterns from data frequent pattern mining is used .Classical frequent pattern mining algorithm is Apriori algorithm .Apriori algorithm is easy to execute and very simple, is used to mine all frequent item sets in database. The algorithm makes many searches in database to find frequent item sets where k-item sets are used to generate k+1-item sets. Each k-item set must be greater than or equal to minimum support threshold to be frequency. Otherwise, it is called candidate item sets. In the first, the algorithm scan database to find frequency of 1-item sets that contains only one item by counting each item in database. The frequency of 1-item sets is used to find the item sets in 2-Item sets which in turn is used to find 3-item sets and so on until there are not any more k-item sets. but the major limitation is its time and space requirements. as reduce the redundant generation of sub-items during pruning the candidate item sets, which can form directly the set of frequent itemsets and eliminate candidate having a subset that is not frequent.

Description of the Apriori algorithm

Input: D, Database of transactions; min_sup, minimum support threshold
Output: L, frequent item sets in D
Method:
(1) L1=find_frequent_1-item sets(D);
(2) for(k=2; Lk-1≠Φ; k++){
(3) Ck=apriori_gen(Lk-1, min_sup);
(4) for each transaction t∈D{
(5) Ct=subset(Ck,t);
(6) for each candidate c∈Ct
(7) c.count++ ;
(8) }
(9) Lk={c∈Ck |c.count≥min_sup }
(10) }
(11) return L=UkLk ;
Procedure apriori_gen(Lk-1:frequent(k-1)-item sets)
(1) for each item set l1∈ Lk-1{
(2) for each item set l2∈ Lk-1{
(3) if(l1 [1]= l2 [1])∧ (l1 [2]= l2 [2]) ∧…∧(l1 [k-2]=l2 [k-2]) ∧(l1 [k-1]< l2 [k-1]) then
{
(4) c=l1∞l2;
(5) ifhas_infrequent_subset(c, Lk-1) then
(6) delete c;
(7) else add c to Ck ;
(8) }}}
(9) return Ck;
Procedure has_infrequent_subset(c: candidate k-item set;
Lk-1:frequent(k-1)-item sets)
(1) for each(k-1)-subset s of c {
(2) if s ∉ Lk-1 then
(3) return true; }
(4) return false;

Drawbacks

1) It does multiple scan over the database to generate candidate set.
2) The number of database passes are equal to the max length of frequent item set.
3) For candidate generation process it takes more memory, space and time.
4) Assumes transaction database is memory resident.
5) It only explains the presence and absence of an item in transactional databases.
6) In this Algorithm, Minimum support threshold used is uniform. Whereas, other methods can address the problem of frequent pattern mining with non-uniform minimum support threshold.

My Proposed Algorithm:

In order to improve the Apriori algorithm, many improvements have been suggested but with limitations. For improvement of algorithm I have introduced an attribute Transaction_Subsets (TS), containing all the subsets of individual transaction in a database which eliminates redundancy of candidates by avoiding multiple phases of candidate generation also Null sets are not considered by using only present items in transaction set.

Algorithm Description

Input: D:Database of transaction; minimum support threshold S_0.

Output: frequent itemsets Fitem_set.

Method:

Step:1 Scan the transaction database D.

Set Fitemset ={ϕ}

For each transaction t∈D

Step:2for (t=0; t<length(D) ; t++)

GenerateTS: All subsets of individual transaction t

Count[] =0; //Total count

Step3: Generate each present element, count all the possible subsets Scount(j) of that transaction t.

Count[] =Count[] +Scount(j)

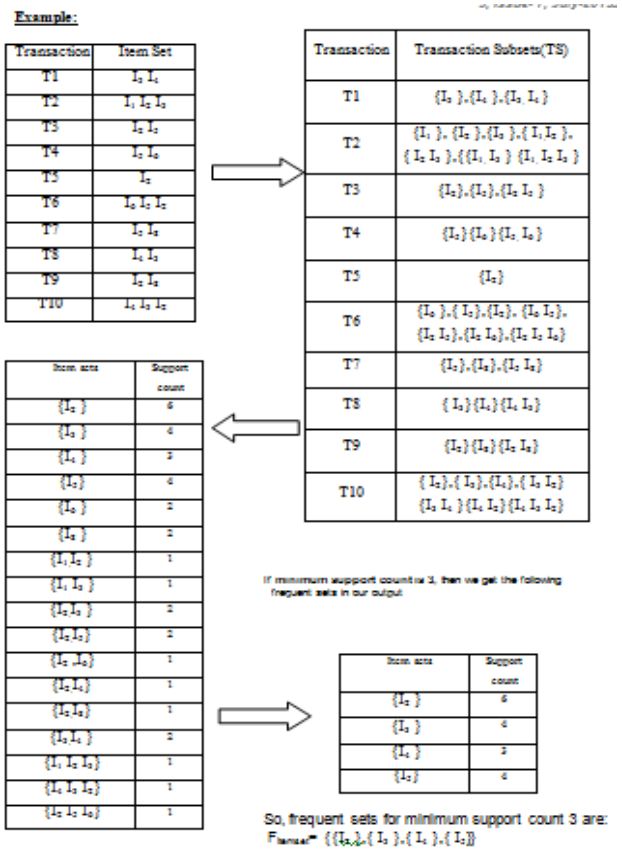
Step:4 If (count[]≥ S_0)

Then Fitem_set = Fitem_set U Ai

Step:5 Go to step 2

Step: 6 End For

Step: 7 return Fitem_Set



In the above algorithm the entire possible item sets were not generated, items **already present** in transaction sets were considered for frequent set Analysis. Hence it reduce time and space requirement by avoiding multiple candidate generation.

It is showed by the Example (Page 5)

IV. RESULT ANALYSIS

For the analysis of the improvements in our algorithm, we have compared the Apriori and the improved algorithm both on similar databases, whose number of transactions (D) vary from 640 to 5120.The minimum support counts (S_0) also vary for each type length i.e. 40, 80,120,160,240.All the operations have been performed on a Windows based CoreI5 Processor machine. The programs have been written on C++ language.

Table 1

No. of transactions (D)	Different Minimum Support Counts(S_0)				
	40	80	120	160	240
640	0.060	0.032	0.036	0.032	0.044
1280	0.172	0.132	0.064	0.064	0.064
2560	1.168	0.836	0.820	0.832	0.744
5120	1.204	0.844	0.816	0.748	0.836

In the above table, *table 1* we see the execution times of the Apriori Algorithm for various lengths of datasets along with different support counts.
Further, we will see that how have these values turned up on a graphical format.

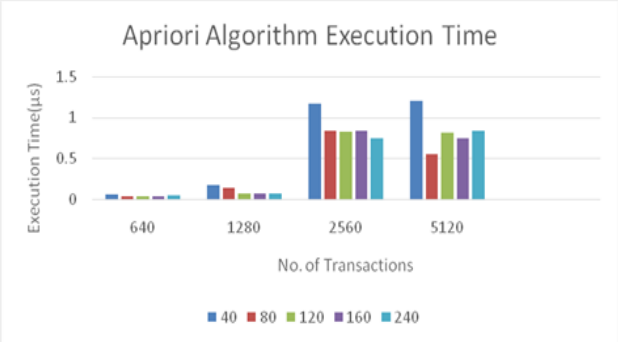


Chart 1

In the above *Chart1* we see the graphical representation of the Apriori execution times against the number of transactions.
We can clearly see that Apriori works in time <0.5µs for smaller databases, but as the number of transactions increases the execution time also increases linearly(more than 1µs).

Table 2

No. of transactions (D)	Different Minimum Support Counts(S_0)				
	40	80	120	160	240
640	0.004	0.004	0.005	0.004	0.004
1280	0.004	0.004	0.004	0.004	0.004
2560	0.004	0.004	0.004	0.005	0.004
5120	0.004	0.004	0.004	0.008	0.004

In the above table, *table 2* we see the execution times of the Improved Algorithm for various lengths of datasets along with different support counts.
Further, we will see that how have these values turned up on a graphical format.

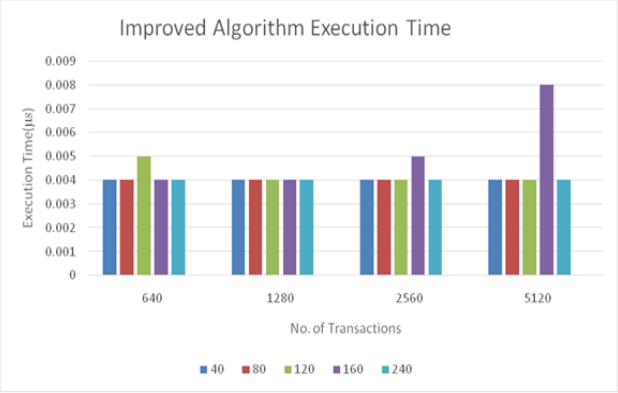


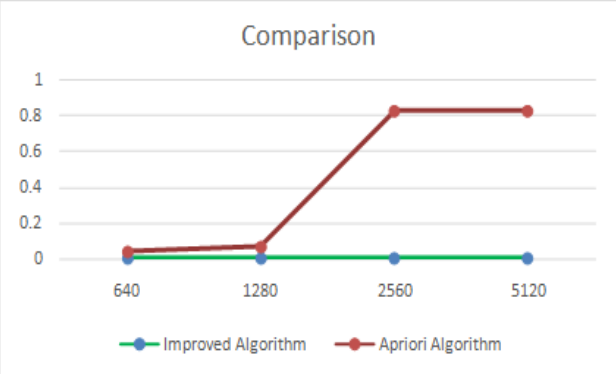
Figure 2

In the above figure, *figure 2* we see the graphical representation of the improved algorithm's execution times against the number of transactions.

It can be seen that Apriori works in time <=0.005µs for smaller databases, but as the number of transactions increases the execution time increases negligibly and we can say that the algorithm runs in an almost constant time.

V. CONCLUSION

- As in the comparison charts, we can see that our algorithm is taking almost constant time, and the Apriori algorithm is increasing almost linearly as the dataset's length is increasing.
- Another point to be noted is that our algorithm is producing more frequent sets than the Apriori algorithm, still it is taking almost 1/100th of the execution time.
- This proves that our algorithm is performing better even for the larger datasets.



REFERENCES

[1] M. M. Gaber, A. Zaslavsky, and S. Krishnaswamy, "**Mining data streams: A review**", ACM SIGMOD Record, Vol. 34, no. 1, 2005.

[2] C. C. Aggarwal, "**Data Streams: models and algorithms**", Springer, 2007.

[3] Du Ping; Gao Yongping, "**A new improvement of Apriori Algorithm for mining association rules**", Computer Application and System Modeling (ICCASM), 2010 International Conference on , vol.2, no., pp.V2-529- V2-532, 22-24 Oct. 2010.

[4] Nicholson, S. The Bibliomining Process: "**Data Warehousing and Data Mining for Library Decision Making**". Information Technology and Libraries, 2003, 22(4):146-151.

[4] Agrawal, Rakesh, "**Fast Algorithms for Mining Association Rules in Large Databases**", Proceedings of the ACM SIGMOD International Conference Management of Data,Washington, 1993, pp.207-216.

[5] Chaudhary, M. Rana, A. Dubey, "**Online Mining of Data to generate association rule mining in large databases**", Recent Trends in Information Systems (ReTIS), International Conference on Dec. 2011,IEEE.

[6] Lu, Lin; Pei-qi, "**Study on Improved Apriori algorithm and its application in Supermarket**", Information Sciences and Interaction Sciences (ICSI), 20103rd International Conference on,vol.,no.,pp.441-443, 23-25 June 2010.

[7] Sheng Chai, Jia Yang, and Yang Cheng, "**The Research of Improved Apriori Algorithm for Mining Association Rules**", Proceedings of the Service Systems and Service Management,2007.

[8] Jinwei Wang and Haitao Li, "**An Interpolation Approach for Missing Context Data Based on the Time-Space Relationship and Association Rule Mining**", Multimedia Information Networking and Security (MINES), 2011,IEEE.

[9] Jinguo, Xin, Tingting and Wei, "**The application of association rules mining in data processing of private economy statistics**", E-Business and E-Government (ICEE), 2011 IEEE.

[10] Shilpa and Sunita Parashar, "**Performance Analysis of Apriori Algorithm with Progressive Approach for Mining Data**", International Journal of Computer Applications (0975 – 8887) Volume 31- No.1, October 2011, pp 13-18.

