International Journal of Research (IJR)
e-ISSN: 2348-6848, p- ISSN: 2348-795X Volume 2, Issue 08, August 2015
Available at http://internationaljournalofresearch.org

Hybrid Apriori- An Improvement

Suruchi Kannoujia [1]& Akhilesh Kosta [2]
Department of Computer Science and Engineering, Kanpur Institute of Technology (UPTU), Kanpur
kannoijiasuruchi@rediffmail.com [1]& akhileshkosta@gmail.com[2]

Abstract

In recent years, there has been a huge accumulation of data. The amount of data collected is said to be almost doubled every year. The paper explores the sense of Data Explosion .Seeking knowledge from massive data is one of the most desired attributes of Data Mining. Several techniques have 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 cost of this algorithm is very high because of repetitive scanning of database. So, our approach focuses on removing this drawback. Our algorithm scans the database only once, and produces the frequent patterns in almost constant time. We have run our algorithm on databases having 100 to 10000 transactions, which showed that it took almost constant time.

Key Words: Data Set; Item Sets; Support; Frequent Items; Apriori;

Paper

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 nontrivial extraction of implicit, previously unknown, and potentially useful information is generated. In the last decade, there has been a gigantic growth in the storage and generation of electronic data. Traditional database systems are often designed for running on daily basis in an organization 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 . The word 'Frequent Pattern Mining' means to extract frequently occurring patterns from data set.That may help a firm/company in forecasting and understanding its business patterns more efficiently . "Association Rules" are the building blocks for frequent pattern mining. Association Rules describes association relationships among the attributes in the set of relevant data.Frequent pattern mining is guided through two features "Support" and "Confidence" .For a Rule:-

***Body ==> Consequent* [*Support* , *Confidence*]**

***Body*:** represents the examined data.

***Consequent*:** represents a discovered property for the examined data.

***Support*:** represents the percentage of the records satisfying the *body* or the *consequent*.

Confidence*:** represents the percentage of the records satisfying both the ***body and the ***consequent*** to those satisfying only the *body*.

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.

International Journal of Research
International Journal of Research (IJR)
e-ISSN: 2348-6848, p- ISSN: 2348-795X Volume 2, Issue 08, August 2015
Available at http://internationaljournalofresearch.org

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;

According to above algorithm, in the first pass, the support of each individual item is counted, and the large ones are determined. In each subsequent pass, the large itemsets determined in the previous pass is used to generate new itemsets called candidate itemsets.The support of each candidate itemset is counted, and the large ones are determined.This process continues until no new large itemsets are found .The major drawbacks are time and space requirements for data analysis because it scans database repeatedly to generate candidate set .The number of database passes are equal to the max length of frequent item set so it takes more memory, space and time. For candidate generation process Assumes transaction database is memory resident. 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 avoided by using only present items in transaction set.

Algorithm Description

Input: D:Database of transaction; minimum

support threshold S0.

Output: frequent itemsets Fitem_set.

Method:

Step:1Scan 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

International Journal of Research
International Journal of Research (IJR)
e-ISSN: 2348-6848, p- ISSN: 2348-795X Volume 2, Issue 08, August 2015
Available at http://internationaljournalofresearch.org

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

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

Step:4If (count[]≥ S0)

Then Fitem_set = Fitem_set U Ai

Step:5Go 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

RESULT ANALYSIS:

For the analysis of the improvements in My algorithm, I have compared the Apriori and the improved algorithm both on similar databases, The number of transactions (D) vary from 6oo to 5500.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++ and C language.

Table 1

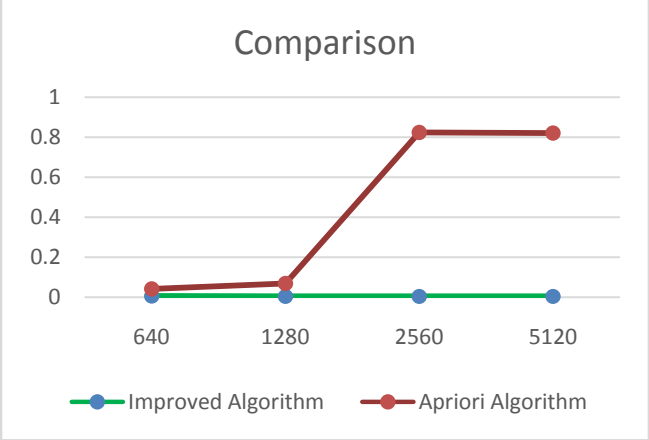
In the above table, *table 1* we see the execution time of the Apriori Algorithm for various lengths of datasets along with different support counts.

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.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.

Chart1

No. of transactions (D)	Different Minimum Support Counts(S_0)				
	40	80	120	160	240
640	0.065	0.032	0.036	0.032	0.043
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

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 $1/100^{th}$ of the execution time.
- This showes 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] 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] Binesh Nair, Amiya Kumar Tripathy, "Accelerating Closed Frequent Itemset Mining by Elimination of Null Transactions", Journal of Emerging Trends in Computing and Information Sciences, Volume 2 No.7, JULY 2011, pp 317-324.

[7] 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.

[8] G. Cormade and M. Hadiielefteriou, "Finding frequent items in data streams", In Proceeding of the 34th International Conference on Very Large Data Bases (VLDB), pages 1530-1541, Auckland, New Zealand, 2008.

[9]Improved Data mining approach to find Frequent Itemset Using Support count table- Ramratan Ahirwal, Neelesh Kumar Kori and Dr.Y.K. Jain.

[10]Improving the efficiency of Apriori Algorithm in Data Mining -Vipul Mangla, Chandni Sarda and Sarthak Madra.