# Revised ECLAT Algorithm for Frequent Itemset Mining

**Bharati Suvalka, Sarika Khandelwal and Chintal Patel**

**Abstract** Data mining is now a day becoming very important due to availability of large amount of data. Extracting important information from warehouse has become very tedious in some cases. One of the most important application of data mining is customer segmentation in marketing, demand analyzes, campaign management, Web usage mining, text mining, customer relationship and so on. Association rule mining is one of the important techniques of data mining used for discovering meaningful patterns from huge collection of data. Frequent item set mining play an important role in mining association rules in finding interesting patterns among complex data. Frequent Pattern Itemset Mining from "Big Data" is used to mine important patterns of item occurrence from large unstructured database. When compared with traditional data warehousing techniques, MapReduce methodology provides distributed data mining process. Dataset can be found in two pattern one is horizontal data set and another one is vertical data set. Tree based frequent pattern MapReduce algorithm is considered more efficient among other horizontal frequent itemset mining methods in terms of memory as well as time complexity. Another algorithm is ECLAT that is implemented on vertical data set and is compared with my proposed revised ECLAT Algorithm. As a result the performance of ECLAT Algorithm is improved in proposed algorithm revised ECLAT. In this paper will discuss improved results and reasons for improved results.

**Keywords** Big data analytics · MapReduce · ECLAT · BFS · FP tree · FP growth

B. Suvalka (✉) · S. Khandelwal · C. Patel
Geetanjali Institute of Technical Studies, Udaipur, India
e-mail: bharatiisuvalka@gmail.com

S. Khandelwal
e-mail: sarikakhandelwal@gmail.com

C. Patel
e-mail: smilingchintal@gmail.com



S.C. Satapathy et al. (eds.), *Information Systems Design and Intelligent Applications*, Advances in Intelligent Systems and Computing 434,
DOI 10.1007/978-81-322-2752-6_21

# 1 Introduction

## *1.1 Big Data*

Nowadays data is not only growing in its size but also increasing in variety and variability. The big data is defined as database that can store, manipulates, analyze and manage huge amount of data whose size is beyond the ability of traditional database tools. The big data and its attributes can be explained via 3V's that is volume, variety, varsity and velocity [1]. The volume refers to the amount of data that can be processed at the same time. Big data have ability to process and store terabytes of data. The frequency by which data is generated by some real time application refers to velocity of the big data. The variety includes different formats of data that is structured, semi structured and unstructured data.

Big Data Analytics is the method to analyze large amount of data to uncover unknown relation, hidden patterns and other useful information about the data [2]. Big data analytics is proved to be beneficial in many areas like business analysis, marketing strategies, customer reviews to make useful decisions for the enterprises. The main objective of big data analytics is to help different enterprises to make better business decisions and other users to analyze huge volumes of transaction data as well as other data which is left by Business Intelligence programs. Predictive analytics for further launching the further business strategies [3].

One of such big data analytics is frequent item set mining. In frequent item set mining the set of items purchased together are examined. This analysis helps the enterprise and stores to launch the schemes that give together packs or money saver offers to tempt the customers. The main purpose of frequent itemset mining is analyse transaction data of the supermarket. This is used to examine the behavior of customer in terms of products purchased. There are various algorithms that are used for frequent item set mining [4]. These algorithms can be implemented on horizontal data format as well as on vertical data format [5].

## *1.2 Frequent Item Set Mining*

Horizontal item set mining takes data in horizontal format and that have two basic algorithms. Most fundamental algorithm for frequent item set mining is Apriori algorithm it converts the database in the form of graph before processing. This algorithm makes multiple passes over the data and it uses multiple iterative approach and BFS to explore itemsets [6].

Apriori algorithm goes through the iterative approach and it scans database for many passes and take database as graph that increases its time complexity. Considering the drawbacks of apriori algorithm FP growth algorithm was

introduced that uses divide and conquer approach. It compresses the database to tree structure called FP Tree that is frequent pattern tree instead of graph [7].

Apriori algorithm and FP growth algorithm were used for horizontal data base and it is found that comparing and scanning capability of vertical data set is less than horizontal so vertical frequent item set mining techniques were introduced.

Then ECLAT algorithm was implemented for frequent data mining. In this paper will discuss the drawbacks of ECLAT and how the proposed algorithm revised ECLAT is used to reduce the time complexity and its graph and improved results are also being explained.

## *1.3 ECLAT Algorithm*

ECLAT Equivalence class transformation was developed to mine frequent item sets efficiently using vertical data format [8]. The length of transaction set in the item set is simply the support count of an item. Starting with k = 1, all the k-item set are used to form the candidate (k + 1) item sets. The intersection of the TID set is used to compute item sets. This process repeats for all transactions.

ECLAT TIDset (T, β)
Input: T is vertical data set and β is support value.
Output: List of frequent item sets

1. Repeat step for i=1 to Li != NULL
   (a) Initialize $P_i$ ={Li ,T(Li) fall itemsets of T}
       Li refers to the item name
       T(Li) refers to the transactions to that particular transactions.
   (b) i=i+1
2. Repeat for j=1 to Pj != NULL
   (a) Xj = Lj and Yj = T(Lj)
   (b) J=j+1
3. Set value of i=1 and k=1
4. Repeat for ( Xj != NULL) and (Yj != NULL)
   (a) J=i+1
   (b) Nij = Xi ∪ Xj and T(Nij) Yi ∩ Yj
   (c) If count of T(Nij) >β then
       Bk= Nij and k=k+1
   (d) i=i+1
5. Print (Bk)
6. End

This ECLAT algorithm have complexity O(k(k + 1)) where k is the no of items. I had gone through the ECLAT and found some scope to reduce complexity in this paper I had proposed the revised ECLAT in which the time complexity is reduced. In revised ECLAT after scanning the data a lexicography algorithm is being applied and then the data having number of transaction count less than that of support count than those items are removed from the list so after that the scanning time is being reduced. Time of process is less for revised ECLAT than that of ECLAT. Improved results and graph are being explained in this paper.

## 1.4 Revised ECLAT Algorithm

In Revised ECLAT algorithm, first step is loading the data from database to the data structure. Then the data is being sorted and arranged in descending order. The transactions having transaction count less than the support count are directly be omitted from the data structure and then the algorithm is implemented.

Revised ECLAT TIDset (T, β)
Input: T is vertical data set and β is support value.
Output: List of frequent item sets

1. Repeat step for i=1 to Li != NULL
   (a) Initialize $P_i$ ={Li ,T(Li) fall itemsets of T}
   Li refers to the item name
   T(Li) refers to the transactions to that particular transactions.
   (b) i=i+1
2. Arrange Pi in lexicography order according to the transaction count of each transaction set corresponding to that item
3. Remove all the item set where count of T(Li) < β
4. Repeat for j=1 to Pj != NULL
   (a) Xj = Lj and Yj = T(Lj)
   (b) J=j+1
5. Set value of i=1 and k=1
6. Repeat for ( Xj != NULL) and (Yj != NULL)
   (a) J=i+1
   (b) Nij = Xi ∪ Xj and T(Nij) Yi ∩ Yj
   (c) If count of T(Nij) > β then
   Bk= Nij and k=k+1
   (d) i=i+1
7. Print (Bk)
8. End

## 1.5 Revised ECLAT Algorithm Without Sorting

Further, the time can be reduced further by eliminating the sorting algorithm from the algorithm and making one to one comparison and directly omitting the item set which is purchased in transaction whose count is less than the support count. The support count is also a taken directly without computation depending upon the scheme to be launched. The support count is decided prior and then it is being taken with inputs. By eliminating the sorting step the processing time is reduced and a good improvement is being observed. The algorithm after improvement is as follow.

Revised ECLAT1 TIDset (T, β)
Input: T is vertical data set and β is support value.
Output: List of frequent item sets

1. Repeat step for i=1 to Li != NULL
   (a) Initialize $P_i$ ={Li ,T(Li) fall itemsets of T}
   Li refers to the item name
   T(Li) refers to the transactions to that particular transactions.
   (b) i=i+1
2. Arrange Pi in lexicography order according to the transaction count of each transaction set corresponding to that item
3. Remove all the item set where count of T(Li) < β
4. Repeat for j=1 to Pj != NULL
   (a) Xj = Lj and Yj = T(Lj)
   (b) J=j+1
5. Set value of i=1 and k=1
6. Repeat for ( Xj != NULL) and (Yj != NULL)
   (a) J=i+1
   (b) Nij = Xi ∪ Xj and T(Nij) Yi ∩ Yj
   (c) If count of T(Nij) > β then
   Bk= Nij and k=k+1
   (d) i=i+1
7. Print (Bk)
8. End

The intersection of transactions and union of itemsets are taken for each transaction with another. When no further intersection is possible or the intersection results in null set then again the support value is compared with the transaction count of each item set and then the transaction set having maximum count will be a printed as resultant itemset and that itemset is called as frequent itemset.

## 2 Results

### *2.1 Observation Table*

Table 1 shows the time difference analyzed by different algorithm in different format having different data volume.

### *2.2 Observation from Graph*

From the graphs it is being observed that as the data volume increases the processing time also increases along with it. Different algorithm uses different technique and the processing time changes. For Apriori algorithm the processing time observed is highest then the ECLAT algorithm was implemented and that algorithm have less processing time as compared to Apriori. The proposed algorithm enhanced ECLAT have processing time less than both Apriori and ECLAT Algorithm. In enhanced ECLAT as the data volume was increasing the processing

**Table 1** Experimental result shows time required to process the data

| S. no. | Data volume | Horizontal data format | Vertical data format | | |
|---|---|---|---|---|---|
| | | Apriori algo. (s) | ECLAT (s) | Enhanced ECLAT (s) | Enhanced ECLAT without sorting (s) |
| 1 | 200 MB | 1.48 | 1.28 | 1.06 | 1.02 |
| 2 | 500 MB | 2.17 | 1.47 | 1.15 | 1.08 |
| 3 | 750 MB | 2.37 | 2.08 | 1.52 | 1.55 |
| 4 | 1 GB | 2.54 | 2.27 | 2.17 | 2.03 |
| 5 | 1.25 GB | 3.08 | 3.02 | 2.53 | 2.37 |
| 6 | 1.5 GB | 4.52 | 4.28 | 4.02 | 3.04 |
| 7 | 1.75 GB | 5.08 | 4.38 | 4.23 | 3.52 |
| 8 | 2 GB | 5.27 | 4.42 | 5.2 | 4.28 |

time was also increasing and when the data volume was 2000 MB the processing time of ECLAT is 4.42 s and for enhanced ECLAT the time increases to 5.2 s. (Fig. 1).

The clear comparison can be observed from the bar graph given below. Processing time for different technique for different data volume is being compared and clearly observed that the Enhanced ECLAT1 Algorithm in which sorting is being eliminated gives the best result among all other algorithms (Fig. 2).

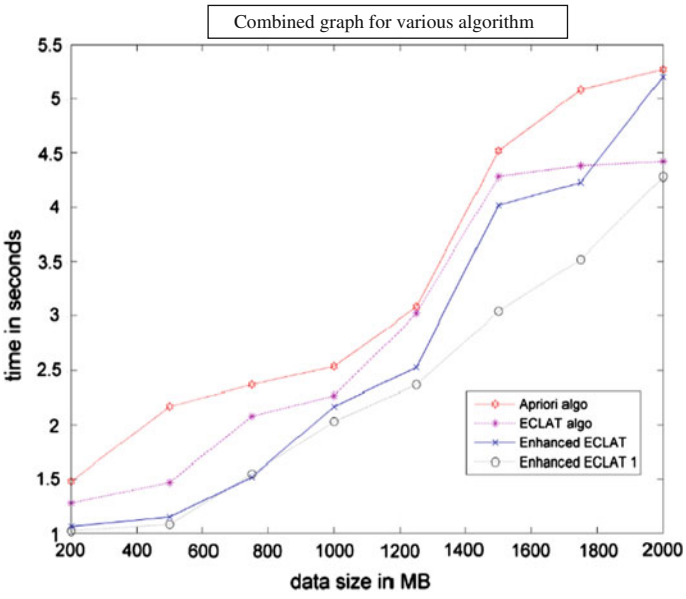


**Fig. 1** Different plot for different algorithm for different data format

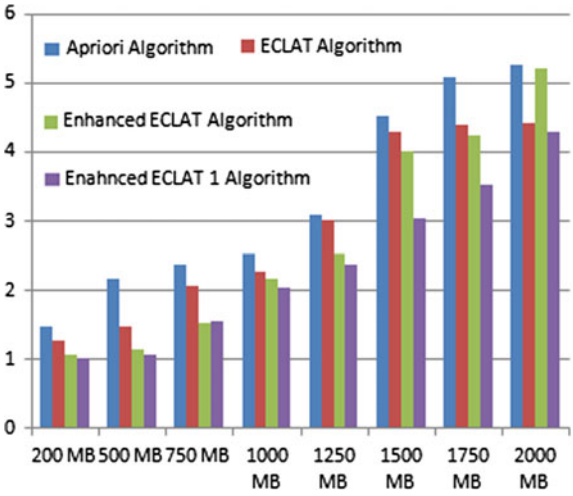


**Fig. 2** Bar graph comparison of different algorithm for different data volume

## *2.3 Inference*

From the observation table, graphs and bar graph it is being observed that as size of data increases the processing time of algorithm also increases. The process time of Apriori Algorithm for horizontal data format is more than that of ECLAT algorithm. ECLAT algorithm is used for vertical data set. The frequent item set mining of data in horizontal format take time more than that of data in vertical data format. After going through ECLAT algorithm new algorithm Enhanced ECLAT is being proposed and observed that processing time is less than that of ECLAT algorithm. In enhanced ECLAT as the data volume was increasing the processing time was also increasing and when the data volume was 2000 MB the processing time of ECLAT is 4.42 s and for enhanced ECLAT the time increases to 5.2 s. The sorting algorithm was eliminated and data is one to one compared and the data with transaction count less than that of support count is directly eliminated from the data structure and then the algorithm is implemented. It is being observed that the Enhanced ECLAT without sorting algorithm give us the best results.

# 3 Limitation

Limitation of revised ECLAT algorithm is that with increasing the size of data the processing time will not give significant changes. When data volume is small then revised ECLAT is most efficient algorithm but with increase in size of data volume it gives the change but much refined.

## 4 Conclusion

Frequent data mining is one of the most important big data analytics. In frequent data mining the analyst search the products purchased together and then it is used to find strategies and consider customer reviews. This helps enterprises to launch their business intelligence strategies. This paper explained frequent data mining on vertical data format using ECLAT algorithm and then proposed a new algorithm named revised ECLAT that took less processing time as shown in result table and graphs.

## References

1. Bharati Suvalka, Sarika Khandelwal, Siddharth Singh Sisodiya "Big Analytics using meta machine learning" in international journal of innovative research in science engineering and technology in August 2014.
2. Tilmann Rabl, Mohammad Sadoghi, Hans-Arno Jacobsen "solving big data challenges for enterprise application Performance management".
3. Noll, Michael G. "Running hadoop on ubuntu linux (single-node cluster)." Mar- 2013 [Online]. Available: http://www.michael-noll.com/tutorials/running-hadoopon-ubuntu-linux-single-nodecluster/.
4. Lam, Chuck. Hadoop in Action. Manning Publications Co., 2010.
5. Han, Jiawei, Micheline Kamber, and Jian Pei. Data mining: concepts and techniques. Morgan kaufmann.
6. Zikopoulos, Paul, and Chris Eaton. Understanding big data: Analytics for enterprise class hadoop and streaming data. McGraw-Hill Osborne Media.
7. Zikopoulos, Paul, Krishnan Parasuraman, Thomas Deutsch, James Giles, and David Corrigan. Harness the Power of Big Data The IBM Big Data Platform. McGraw Hill Professional.
8. Shvachko, Konstantin, Hairong Kuang, Sanjay Radia, and Robert Chansler. "The hadoop distributed file system." In Mass Storage Systems and Technologies (MSST), 2010 IEEE 26th Symposium.