**Available Online at www.ijcsmc.com**

**International Journal of Computer Science and Mobile Computing**

**A Monthly Journal of Computer Science and Information Technology**

**ISSN 2320–088X**

***IJCSMC, Vol. 4, Issue. 5, May 2015, pg.450 – 456***

**RESEARCH ARTICLE**

# Mining Frequent and Correlated Items Using Map Reduce Framework and Tree Data Structure

**Prajapati Khushali[1], Mr. Mahesh Panchal[2]**

[1]Kalol Institute of Technology & Research Centre, India

[2]Deputy Director at Gujarat Technological University, India

[1] Prajapatikhushali4292@gmail.com; [2] mkhpanchal@gmail.com

***Abstract— Big Data has a characteristics like large like volume, velocity, variability, complexity value. Big Data mining is the capability of extracting useful information from these large datasets or streams of data. Fr The combinatorial explosion of FIM methods becomes even more problematic when they are applied to Big Data Fortunately; recent improvements in the field of parallel programming already provide good tools to tackle this problem. However, these tools come with their own technical challenges, e.g. balanced data distribution and inter-communication costs. There are various methods applied on big data for finding the frequent Itemset. Combination of algorithms like Apriori, Fp-Growth, and Eclat are also necessary to apply them on the large dataset. All the methods in literature review are useful for big data mining using Map-Reduce.to balance the load on different mappers and to decrease the processing time the method which use the concept of both Fp-Growth and Eclat is proposed here.to generate Fp Tree for very large dataset is very complex and increase the overhead. This is overcome by the method which use dataset in vertical format and conditional tree is mine separately for each Item. By applying this methods results are generated which improve the processing time and also reduced the overhead.***

***Keywords— Big data, Hadoop, Map Reduce, Dist Eclat, Frequent Itemset Mining***

## I. INTRODUCTION

The amount of data to be produce day by bay is vast, we can say that big data is a big business, but having data and limited power it is not necessary that it will give the meaningful results. big data[1] has size in terms of petabyte, terabyte etc.so it is very hard to process it on a single machine with the traditional methods.it is also require some structure to process the complex data.it is the recent technology in the market which provide higher benefit to the business organization. We also have to introduce some tools which are support the processing of large amount of data like Hadoop.in order to identify the meaningful and frequent item from the big data, we have to deal with some issue and challenges. Frequent item set mining is the most useful concept to identify the frequent items from the large dataset which are stored on the clusters. We can define this them as "a process to find useful information according to frequently occurring events". We have to define threshold value for the database and the items which are fall below the value of support count are consider as a frequent items.

Frequent Itemset Mining (FIM) is one of the most well-known techniques to extract knowledge from data. The Combinatorial explosions of FIM methods become even more Problematic when they are applied to Big Data. To apply FIM methods on the big data there is need to handle. So it is capability of extracting useful information from the large dataset. So there are many techniques for frequent item set mining. it is nothing but the process of find out the most occurring item set from the dataset. For big data mining, we can apply those method on Hadoop map reduce. Now, The Goal is to identify the useful and frequent items from the long Database. There is different frequent Itemset Mining techniques are Available. The traditional method like Apriori, FpGrowth, Eclat, etc. due to large size of Data these methods are inefficient to handle it. So, Parallel programme is necessary to deal with data.

We believe that, although the initial design principles of the MapReduce framework do not fit well for the Frequent Itemset Mining problem, it is important to provide MapReduce with efficient and easy to use data mining methods, considering its availability and wide spread usage in industry. For big data mining, we can apply those method on Hadoop map reduce frame work with different modification and enhancement.

## II. BACKGROUND KNOWLEDGE

The term "Big data[1]" is capture the meaning of this emerging world in the terms of data coming from various domain like scientific sector, user generated data, data on the internet, business sector of world etc. there is a brief history about the big data is demonstrate and it is follow by the various attributive definition of the big data and also some of the challenges of the big data. There is also comparison between traditional data and big data is also shown here. Big data and its Analysis is the centre of our Modern world there are various source like Logs, Search queries, Streams, Videos, Emails, Images, etc. the history of the big data is presented in terms of the data size.

### (A) Characteristic of Big Data[1][2]

- Variety

The data coming from variety of sources. On the internet there are different sites like twitter, you-tube, social media, and emails are available. So the data may be in the type of raw, semi-structured structured, un-structured.

- Volume

The Big word in Big data itself defines the volume. At present the data existing is in petabytes and is supposed to increase to zettabytes in nearby future.

- Velocity

It is focus on the speed of the data coming from the different source. The uploading of data Is grow up day by day.so a huge amount of data is collected with different speed of their sources

- Complexity

The complexity is generally high due to size and distributive source. Her data are very complex because it is coming from the various sources so there are heterogeneity between the data.so we have to define some method to link those data and to identify the relationship between this data. For that various algorithms are established.

- Variability

Variability considers the inconsistencies of the data flow. Data loads become challenging to be maintained especially with the increase in usage of the social media which generally causes peak in data loads with certain events occurring.

### (B) Frequent Itemset Mining[4]

Frequent set are very important in data mining task and they are useful in so many user applications, because they used to find the interesting pattern from the database which is occur more than one time.to apply frequent itemset mining on the supermarket database, they are find the pattern by analysing the buying behaviour of the customer in terms of purchased product.so they give very much benefit in the business sector to increase their productivity. The frequent itemset mining is the process of finding the

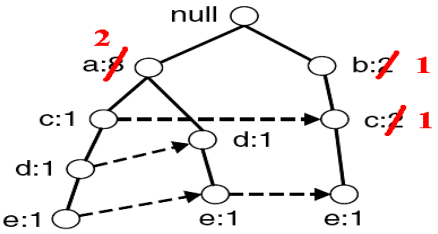
frequently occurring events. For most of the database algorithms user have to define the threshold value. And then the items with the lower value are said to be infrequent and they are removed from the database.by using the association rules the correlation between the item are found.

1) **Itemset:-** it is a collection of number of items in the database. For different database the itemset are contain different values.so according to the value of the database the itemset contain the one or more items. Example : { Milk, Bread, Diaper}
2) **K-Itemset:-**An Itemset that contains k items. Here instead of K there value like 1,2,3…which means that the itemset contain that number of items.
3) **Support-count (σ):-**the occurrence of that particular item or we can say that the frequency of that item. Example: σ ({Milk, Bread, Diaper})
4) **Support: -** Fraction of transactions that contain an Itemset. Example: s ({Milk, Bread, Diaper}) = 2/5
5) **Frequent Itemset:-**An Itemset whose support is greater than or equal to a minisup threshold. There are various method for frequent Itemset mining. And also there are different algorithms are developed for it. Basically there are traditional algorithms which are most useful like Apriori, Fp growth, Eclat.

**Fp Growth [4]**

**Step 1: Build a compact data structure called "FP tree".**
Here, Fp tree is constructed using two different pass over the transactional database.
**Pass 1**: In the first pass of the database scan ,it will count the frequency of each item and then stored all the items into the descending order in the database. Like suppose there is set of items called {a,b,c,d},and their frequency values are like {4,6,7,3}.so in this scan the sorting is occur and the items are stored in this manner {c,b,a,d}.the main reason behind this is to share more and more prefix as can as possible.so we generate the compact FP tree for that.
**Pass 2:** Construct the FP tree
In the second pass over the database, the FP tree is generated, one by one row of the database is scan, and the items in the transactions are fetched and add as a child of the tree. common prefix are add in the tree and their support count value is increased according to the support value of that particular item.so from root to the leaves the transaction with the common items are overlapped with each other and just frequency of that item is increased.

**Conditional FP-Tree[4]:** conditional FP tree is a more compact tree structure which is only contains the item for which we want to find the frequent set. It is only show the transaction of this particular item. And then recursively it will find the items with prefix of that particular item.




[Figure 1: Example of conditional tree[4]]

**Eclat algorithm [5]**

Equivalence Class Clustering and bottom up Lattice Traversal is known as ECLAT [4] algorithm. This algorithm is also used to perform item set mining. It operates into the two steps, called as a **candidate set generation and pruning.** It uses the vertical representation of the database. And use the depth first search method to search the frequent items.

**Candidate generation:** An $k-itemset$ is generated by taking union of two $(k-1)-itemsets$ which have $(k-2)$ items in common, the two (k-1)-itemsets are called parent itemsets of the k-itemset. Fox example, $\{abc\} = \{ab\} \cup \{ac\}$, {ab} and {ac} are parent of {abc}. The (k-1) items are sorted in some order either ascending or descending. Whenever generating a candidate itemset, we also generate its tidset by intersecting the tidsets of its parent. And the support of the candidate is the size of its tidset, so that support counting is trivial and it is done simultaneously with candidate generating.

**Pruning Step:** in the vertical format of the database. After generating the candidate set the support count for each item is counted again and the item with lower value than the support are prune in the second step. And then again with remaining itemset this two steps are perform iteratively until the condition is not satisfy.
**Advantages:** support count is very easy

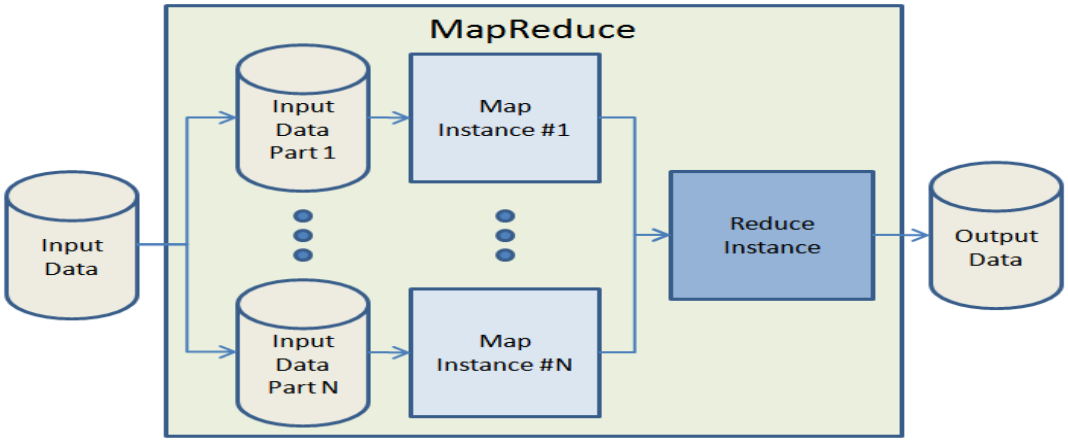
**(C) Mapreduce with Hadoop [6]**

**1)** Firstly, the input data is split into the M no of pieces. They are stored in the equal size block which have size of 64 MB and stored them and manage by the HDFS.so our large database is distributed among the M no of mappers and process by them parallel. Usually the size of the mapper is decided by the user. You can choose the number of mappers according your data size and storage and processing requirements.

**2)** There is also two important entity plays a major roles. Called job tracker and task tracker. Job tracker assign the JOB ID to each part of the job and task tracker assign the local memory and also indicate that whether the task is ready or not? Then job tracker assign job to th mappers.

**3)** Among so many nodes on of the node is called master node which is responsible for the management of the other nodes and copies of the chunks of the data. The mapper generate the <key, value> pairs of the given input data. There are so many intermediate <key, value>pair files, they are stored into the local disk as an intermediate results.

4) After copy the intermediate results into the local buffer the location of each input split are Assign to the master node .then master node notify the reducer about the location of the intermediate input files.




**[Figure 2: mapreduce working [6] ]**

5) When a reduce worker is notified by the master about these locations, it uses remote procedure calls to read the buffered data from the local disks of the map workers. When a reduce worker has read all intermediate data. after reading the data the input files are sort all the intermediate results and then merge the results with the same no of key value and then total the count of that particular key from all the data chunks which are mapped by the mappers.so we can say that there is a two process done on is sorting of the <key, value>pairs and the other is the merging of those data.

6) The reducer then write the results generated by the reducer worker to the output file and store the final results. When all map tasks and reduce tasks have been completed, the master wakes up the user program. At this point, the MapReduce call in the user program returns back to the user code.

## III. LITERATURE REVIEW

### 1) Dist-Éclat[7]

It is a MapReduce implementation of the well-known Eclat algorithm optimized for speed in case a specific encoding of the data fits into memory. The prefix tree[5] that is used by Eclat can be partitioned into independent groups. Each one of these independent groups can be mined separately on different machines the total execution time of an algorithm highly depends on the running time of the longest running sub task. In the case of Eclat, running time is a function of the number of frequent Itemset. There are three basic steps of Dist-Eclat

- **Finding the Frequent Items**
  During the first step, the vertical database is divided into equally sized blocks (shards) and distributed to available mappers. Each mapper extracts the frequent singletons from its shard. In the reduce phase, all frequent items are gathered without further processing.
- **k-FIs Generation**
  In the second step of this algorithm there is k-frequent Itemset are generated from the transactional database
- **Sub Tree Mining**
  In the last step, the prefix tree which is generated from the stored dataset is mine by the mappers to identify the frequent pattern generated from the prefix tree.

### 2) BPFP (balanced parallel FP-Growth algorithm)[9]

- **Sharding:** in this step database DB is divided into equal partitions. And all partitions are storing into the P different machins.in the Hadoop the database need to copy into the HDFS.so they can Sharding the database automatically.
- **Parallel Counting**: in the second phase we have to give the input to the mapper as a one shard of data. And then mapper generate the result in terms of <key, value> pair. Then reducers combine all the items with the same key and count the total value of that item. This whole process is done in parallel so the processing time can be fairly reducing compare to other distributed environment.

- **Balanced Grouping**: Balanced Grouping fairly divides all the items in the F-List into Q groups, balancing load among all groups, in order to improve parallelization of the whole mining process.

### 3) MREclat [10]

- **The Initial Step:** in this step database DB is divided into equal partitions. And all partitions are storing into the P different machins.in the Hadoop the database need to copy into the HDFS.so they can Sharding the database automatically.

- **Balanced Group:** the balance between the group is done using this equations. where W denote the weight , prefix of the items are denoted by A. the no of intersect operations between frequent itemset a and b is equal to length of a and length of b.

- **The Redistribution and Parallel Eclat Step**: In this step, firstly, we want to redistribute 2-itemsets with the same prefix into a same computing node. For example, a 2-itemset set “ab, ac, ad, bc, bd, cd…” and their tid-lists, “ab”, ”ac” , ”ad” have the same prefix “a”. “bc”, “bd” have the same prefix “b”. Then, we will send “ab”, ”ac” , ”ad” accompany with their tid-lists to one computing node A, “bc”, “bd” accompany with their tid-lists to the computing node B. Secondly, we compute frequent Itemsets on different nodes separately.

## IV. PROPOSED METHODOLOGY

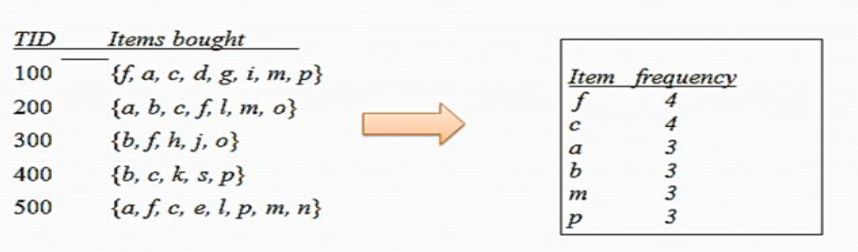
In the methodology which is describe here is basically the combination of some basic rules of FP-growth and Eclat algorithm. Here step wise with example the method is explain. Basic concept of this concept is to provide easy and speedup the processing of large dataset. For that we have use the mapreduce platform to deploy our method. The dataset is stored vertically into the database Item wise. And then items are distributed on the mappers. For distribution of search space we use the approach of we can map more than one item to single node whose threshold is higher. And we will distribute the items with lower threshold to unique mappers. Then direct conditional tree is generated on the mapper. After that reducer will collect all the items from different mapper. And then return the no of items generation each node.

**STEP-1**Firstly, Scan the transaction database and generate

a) 1-frequent Itemset

b) Sort items within transactions in descending order of frequency

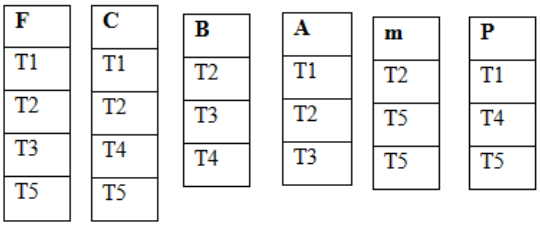


| TID | Items bought |
|---|---|
| 100 | {f, a, c, d, g, i, m, p} |
| 200 | {a, b, c, f, l, m, o} |
| 300 | {b, f, h, j, o} |
| 400 | {b, c, k, s, p} |
| 500 | {a, f, c, e, l, p, m, n} |

| Item | frequency |
|---|---|
| f | 4 |
| c | 4 |
| a | 3 |
| b | 3 |
| m | 3 |
| p | 3 |

[Figure 3**:** count the frequency of database]

**STEP-2:-** Store transaction database into the vertical format in the vertical format there is item name and the no of transaction of that item are listed in the vertical column. Here in this figure the item are stored vertically and shown here



| F | C | B | A | m | P |
|---|---|---|---|---|---|
| T1 | T1 | T2 | T1 | T2 | T1 |
| T2 | T2 | T3 | T2 | T5 | T4 |
| T3 | T4 | T4 | T3 | T5 | T5 |
| T5 | T5 | | | | |

[Figure 4: vertical representation of database]

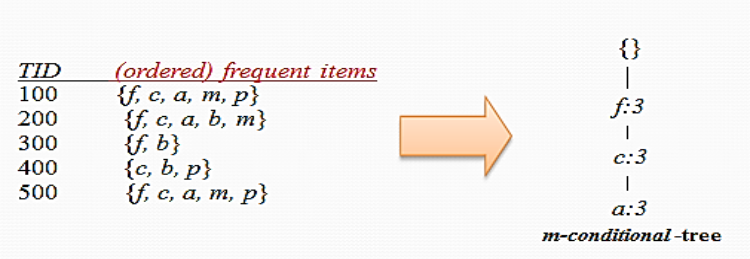
**STEP-3:-** Items are distributed on mappers considering the search space of finding frequent Item sets. Maximum no of mappers are equal to number of distinct items

**a) Distribution of Items is based**

1) You can map the different items with higher support count on single node.

2) You can map the different items with lower support count on individual mapper.

**b) Generate Conditional Tree of each item on mapper**



| TID | (ordered) frequent items |
|---|---|
| 100 | {f, c, a, m, p} |
| 200 | {f, c, a, b, m} |
| 300 | {f, b} |
| 400 | {c, b, p} |
| 500 | {f, c, a, m, p} |



[Figure 5: generation of condition tree]

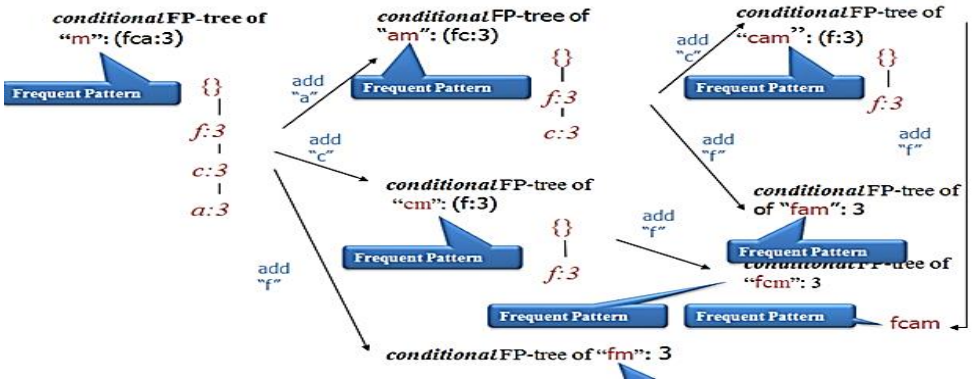
**STEP-4:-**Represent the conditional tree of each item into an array structure. Here the advantage of doing this is listed below:-

- Less memory require
- reduce the complexity
- reduce the overhead

| No | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| Label() | root | F | c | a | B |
| Count() | 0 | 2 | 2 | 3 | 1 |
| Parent() | null | 0 | 1 | 1 | 1 |

**STEP-5:-** Recursively mine the conditional- tree.Recursively mine conditional FP-trees and grow frequent patterns obtained so far. If the conditional FP-tree contains a single path, simply enumerate all the patterns




## V. CONCLUSION AND FUTURE WORK

In this paper there are various fundamental about the Hadoop, Big Data and methods which are suitable for big data mining to find frequent itemset. There are various algorithms like Dist-Eclat, balanced parallel FP-Growth algorithm, MREclat are explaining. The drawback of Dist Eclat algorithm are overcome by this proposed method, by using some load balancing and the array structure to represent the tree in compact form ,are increase the performance and also reduce the overhead between the mappers.so the total execution time to find the frequent items are reduced.in the future work we can also apply some the load balancing policy and check for the best result. There is also need to deal with the problem regarding the memory requirement which is require by this method.

## REFERENCES

[1] Sachchidanand Singh, Nirmala Singh, *"Big Data Analytics*", IEEE, International Conference on Communication, Information & Computing Technology (ICCICT), Oct.19-20, 201.

[2] Wei Fan, Albert Bifet "*Mining Big Data: Current Status, and Forecast to the Future",* SIGKDD Explorations Volume 14, Issue 2S.

[3] Mr Swapnil A. Kale1, Prof. Sangram S.Dandge, Understanding "*The Big Data Problems And Their Solutions Using Hadoop and Map-reduce*", International Journal of Application or Innovation in Engineering & Management (IJAIEM), Volume 3, Issue 3, March 201.

[4] Jiawei Han, Jian Pei, and Yiwen Yin" *Mining Frequent Patterns without Candidate Generation",* School of Computing Science.

[5] S.Vijayarani, P.Sathya," Mining Frequent Item Sets over Data Streams using Éclat Algorithm", International Conference on Research Trends in Computer Technologies (ICRTCT - 2013).

[6] Jeffrey Dean, Sanjay Ghemawat," MapReduce: Simplified Data Processing on Large Clusters", USENIX Association OSDI '04: 6th Symposium on Operating Systems Design

[7] Emin Aksehirli and Bart Goethals, Sandy Moens"Frequent Itemset Mining for Big Data,".

[8] Avita Katal, Mohammad Wazid, R H Goudar "Big Data: Issues, Challenges, Tools and Good Practices", 2013 IEEE.

[9] Le Zhou, Zhiyong Zhong, Jin Chang Junjie Li, Joshua Zhexue Huang Shengzhong Feng," Balanced Parallel FP-Growth with MapReduce", 2010-IEEE

[10] Zhigang Zhang, Genlin Ji*, Mengmeng Tang, "MREclat: an Algorithm for Parallel Mining Frequent Itemsets", 2014-IEEE.

[11] Apache Hadoop. http://hadoop.apache.org/, 2013.