Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 5 Issue 05, May-2016

Association Rule Mining on Big Data – A Survey

Dr. R Nedunchezhian
Director of Research
KIT – Kalaignarkarunanidhi Institute of Technology
Coimbatore

K Geethanandhini
PG Scholar
Department of CSE
KIT – Kalaignarkarunanidhi Institute of Technology
Coimbatore

***Abstract*: Frequent pattern mining is the key concept in Association Rule Mining task. Main aim of frequent pattern mining is to find the recurrent patterns occurring in a dataset. Finding patterns identify the relationship between items in an item domain, these relationships are useful for strategic decision making. Data is flooded in a day to day life, called "Big Data", because massive amount of data is produced everywhere. Mining frequent patterns from the huge volumes of data has many challenges due to memory requirement, multiple data dimensions, heterogeneity of data and so on. The complexities related to mining frequent itemsets from a Big Data can be minimized by parallelizing the mining task with Map Reduce framework in Hadoop Cluster [1]. In this paper, an introduction to Big Data, Association Rule Mining, concepts and basic methods for frequent pattern mining are given. The various methods proposed by different authors to mine frequent patterns from enormous dataset effectively are also discussed.**

Keywords: Association Rule Mining (ARM), Frequent Pattern Mining (FPM), Big Data, Map - Reduce, Hadoop

I. INTRODUCTION

A) Big data

Big Data is used everywhere now and it is an emerging technology being used by many enterprises to understand the business trends from their historical data [3]. Big data is a large set of data which consist of different data types with different dimensions. There are three types of data: *i) Structured ii) Semistructured and iii) Unstructured*. Most of the Big Data are Unstructured, because they don't have any predefined patterns; it can be of any form like image, audio, video and geospatial. *Structured* data follow standard schema's. Semistructured data is a combination of both structured & unstructured data such as customer name and compliant. Among the available data only 20% data is structured.

The Big Data is expressed in terms of four dimensions [3] *i) Volume* - it represents the amount of data as big data is defined in terms of petabytes and zettabytes., *ii) Velocity* - *it* deals with the accelerating speed at which data flows in from sources like business processes, machines, networks and human interaction with things like social media sites and mobile devices, *iii) Variety* refers to the increasingly diversified sources and types of data requiring management and analysis. Now data comes in the form of emails, photos, videos, monitoring devices, PDFs and audio which differ from the conventional data sets. *iv) Veracity* - refers to the biases, noise and abnormality in data being generated. The quality of the mining result should not be deteriorated owing to the presence of outliers. Given increasing volume of data at an unprecedented rate and in ever more diverse forms, hence there is a clear need for managing the uncertainty associated with particular types of data.

An important function of Big Data is *"Big Data Analytics",* it is a process of discovering hidden patterns, relationship between the items, and customer interest on a particular item from the large data sets. Different techniques are applied to derive valid, previously unknown and potentially useful patterns from mountains of data. Association rule mining is one among the mining functionalities applied for big data analytics in order to find out the affinities among items. Frequent item set mining helps in locating the relationships between items. The next subsection deals with the introduction of Association Rule Mining (ARM).

B) Association rule mining

Association rules are conditional statements (if/then) that are used to uncover relationships between apparently unrelated data in a relational database or other information repository. It was introduced by Agrawal et al. in 1993. A simple example of an association rule would be "If a customer buys a CPU, he is 80% likely to purchase Keyboard also." Main aim of Association Rule Mining (ARM) is to find the frequent item set in the database [13]. These association rules are used in various fields to enhance business, especially in the field of marketing. Many algorithms, methods and techniques have been proposed by ARM research community used to find the frequent itemsets with minimum complexities.

Association rules have to meet the predefined minimum support and confidence as they set by the user. The goal is to find associations between items that occur together in transactions. Usually the ARM process involves two important steps, first is to find the frequent itemsets from massive amount of datasets and the second step is to generate associations from those derived frequent itemsets.

Let $I=\{i_1,i_2,\ldots..,i_n\}$ be the set of items in a dataset and $T=\{t_1,t_2,\ldots..,t_m\}$ is the set of transactions in the dataset, it contain m transactions. Association rules are expressed in the form of $X=>Y$, where $X,Y \subset I$ are itemsets, and $X \cap Y= \emptyset$. where, X is antecedent and Y is consequent.
Two thresholds of ARM are minimum support (min sup) and minimum confidence (min con) [5].

Support Count (σ): Frequency of occurrence of itemset in transactions $\sigma(\{X, Y\})$.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 5 Issue 05, May-2016

Support (s): Fraction of transaction that contain an itemset, it also determines how often a rule is applicable to a given data set.

Support,

$$s(X \rightarrow Y) = \frac{\sigma(XUY)}{m} \quad 1.1$$

Confidence (c) : It is used to measures how often items in Y appear in transactions that contain X.

Confidence,

$$c(X \rightarrow Y) == \frac{\sigma(XUY)}{\sigma(X)} \quad 1.2$$

Support is an important measure because a rule that has very low support may occur simply by chance, so it can eliminate such rules. Confidence measures the reliability of the inference made by a rule. It provides the conditional probability of Y given X.

It is to find all the interesting rules having support ≥ minsup and confidence ≥ minconf from a dataset D. Total number of possible rules that can be extracted from the dataset with n items in the item domain (I),

$$R = 3^m - 2^{m+1} + 1 \quad [12] \quad 1.3$$

The remainder of this paper is organized as follows. Section II describes about the basic methods used for frequent pattern mining. In Section III different methods proposed for frequent pattern mining in Big Data are discussed.

II. FREQUENT ITEMSET MINING:

Many algorithms have been proposed in the past. The algorithm proposed by Agarwal and Srikant in the year of 1994 called "APRIORI", is a benchmark algorithm for mining association rules. The apriori algorithm is explained in detail below to express the different steps involved in mining frequent itemsets.

A) Apriori Algorithm:

It is an influential algorithm for mining frequent itemsets for Boolean association rules. It follows Bottom Up approach in which frequent subsets are extended one at a time also called candidate set and this step is called candidate generation. Group of candidates are tested against the transaction in the dataset.

*Apriori Property: "*Any subset of a frequent itemset must be frequent"[13]

Steps in the Apriori :

The algorithm works on the fact that the algorithm possess prior knowledge of frequent itemsets.

Step 1: The algorithm visits the dataset for calculating the frequency count of candidates. The number of scans on dataset varies with the maximum length of the frequent itemset in the dataset.

Step 2: After the first scan over the dataset the set of frequent 1-itemsets are created, L_1.Then L_1 is self-joined to generate candidates of the length two, C_2 and so on until no frequent K-itemsets can be found.

Step 3: Scan the dataset again to find the support count of each k itemsets.

Step 4: If the candidate itemset is null it move on to the next step, otherwise step 2 and 3 repeated.

Step 5: For each frequent itemset 1, generate all nonempty subsets of 1.This iterative approach known as *Level wise search*, where K- items are used to explore (K+1)-itemsets from transactional databases for Boolean Rules.

Limitations:

- Generate and test is a major drawback of the Apriori algorithm. It involves large number of candidate generations and repeated visits to the datasets. To find a frequent pattern of size 100 need to generate totally $2^{100} - 2 \approx 10^{30}$ candidates.
- It costs too high to manage and store the large amount of candidate itemsets.

B) ECLAT ALGORITHM:

This method was proposed by Zaki[19], in the year of 1997.The previous algorithm uses horizontal representation but this algorithm follows vertical representation of dataset. Eclat algorithm uses bottom up approach to finds the itemsets from the dataset like depth first search. Eclat algorithm is very simple method to find the frequent item sets. If there is any horizontal database, then we need to convert into vertical database.

There is no need to scan the database further. Eclat algorithm scans the database only once. Each item is stored together with its cover, also called tidlist and uses the intersection operation compute the support of an itemset. It is suitable for small datasets also it requires less time for frequent pattern generation than apriori.

Steps in the Eclat algorithm [19]:

1. Transform the horizontal dataset into vertical format by scanning the entire database only once.
2. Estimate the support count of each itemset used as the length of the TID set of the respective itemset.
3. Construct (k+1) candidate itemsets using he frequent k-itemsets based on the Apriori property.
4. Repeat step 2 and 3 , until no frequent items or no candidate itemsets can be found.

Properties of Eclat algorithm:

- Take the advantage of the Apriori property in the generation of candidate (k+1)-itemsets from k-itemsets
- No need to scan the database to find the support of (k+1) itemsets, for k>=1
- The TID_set of each k-itemset carries the complete information required for counting such support
- The TID-sets can be quite long, hence expensive to manipulate
- Use diffset technique to optimize the support count computation

C) FREQUENT PATTERN TREE (FP – GROWTH):

The algorithm was proposed by Han in the year of 2000[7]. It is more efficient than the previous algorithms because it does not generate candidates. A tree pruning method called frequent pattern tree tis applied to find frequent patterns. It is a two-step approach[7]:

Step 1: Build a compact data structure called FP-Tree. This is built using two passes over the data set.

Step 2: It divides the FP-Tree into a set of conditional datasets and mines each dataset separately and extracts frequent itemsets directly from the FP-Tree.

In this technique it is necessary to perform the first scan of transaction dataset to identify set of frequent items.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 5 Issue 05, May-2016

STEP 1: FP-Tree construction

Pass 1:

- Scan the dataset and find support for each item. Then discard the infrequent items whose support is less than the minsup.
- Sort the frequent items in decreasing order based on their support.
- Based on the order FP-Tree is constructed, so common prefix can be shared.

Pass 2:

Here nodes representing the items and they have counter to update the frequency of items.

- FP-growth reads first transaction at a time and maps it to a path.
- The fixed order is used, so paths can overlap when transactions share same items. If the transactions shares same prefix then the counters are incremented.
- Pointers are maintained between nodes containing the same item. This forms singly linked list.
- If more paths overlap, it needs higher compression then only FP-Tree may fit in memory.
- Finally frequent items are extracted from the FP-Tree.

FP-Tree size:

The FP-Tree usually has a smaller size than uncompressed data because in FP-Tree transactions share the prefix if they have same items [7]. Sharing prefix typically reduce the size of the tree. In the *Best Case Scenario*: all the transactions have same set of items then only one path in the FP-Tree. *Worst Case Scenario:* every transaction has a unique set of items then size of the FP-Tree is as large as the original data. An extra storage requirement for the FP-Tree is needed, to store the pointers between the nodes and the counters. Size of the FP-Tree depends on how the items are ordered, usually decreasing support is used but it does not always lead to the small size.

STEP 2: Frequent Itemset Generation

After constructing the FP-Tree, have to find frequent itemsets. Frequent itemsets are found in the following manner,

- It follows divide and conquer method:

Conditional FP-Tree:

- First it looks for the frequent 1-itemset ending with the first item in sorted list. This process is for all the items in the sorted list. (Eg. c,bc,ac,abc,b,ab,a), Conditional FP-Tree is constructed for every items in sorted list.
- Extract prefix path sub trees ending with a single item with the help of link list.
- Each prefix path sub tree is processed recursively to extract the frequent itemsets.
- Finally solutions of each prefix path sub tree are merged.

Properties of FP-growth:

- FP-growth transforms the problem of finding long frequent patterns to searching for shorter once recursively and concatenating the suffix
- It uses the least frequent suffix offering a good selectivity
- It reduces the search cost
- If the tree does not fit into main memory, partition the database
- Efficient and scalable for mining both long and short frequent patterns

III. FREQUENT PATTERN MINING IN BIG DATA

In Big Data analytics it is important to find frequent items to make decisions. Big Data analytics is an emerging and growing technology used by the enterprises to know their client's interest on their products, by analyzing the previous purchase of their clients. Analyzing the previous purchases we can find frequent patterns, based on this we made a decision about client's habit of purchasing. It can help in improving the profits of a business organization.

In [16], a new method introduced, Frequent Ultrametric Tree (FIUT), to find frequent patterns as an alternate to FP Tree. The proposed FIUT consists of two main phases and it needs only two scans of dataset, D. In phase1 it calculates support for each item in a dataset. Then, a pruning technique is used to remove all infrequent items, leaving only frequent items to generate the k-itemsets. Meanwhile, all the frequent 1-itemset are generated. Phase2 is the repetitive construction of small ultrametric trees, the actual mining of these trees, and their output. FIUT has four main salient features i) it scans the database only twice so it gradually minimize I/O overhead, ii) In case clustering FIUT is an efficient way to partition a database so it reduces the search space, iii) It inserts only the frequent items to FIUT for compressed storage, iv) By checking the leaves of FIUT, frequent itemsets are generated without recursive traversal through which the computing time is reduced.

Merits: Computing time is reduced since it uses compressed data structure in turn the search space also reduced.

In [8], Tidset-based Parallel FPtree (TPFP-tree) and Balanced Tidset-based Parallel FP-tree (BTP-tree) are proposed based on cluster and grid computing. The first parallel FP-tree algorithm based TID is developed for Cluster computing. Cluster computation is homogeneous, the TPFP -tree distributes the workload to each processor equally without considering the efficiency of the processors. The main aim of TPFP-tree is to reduce the execution time of mining information exchange and reduce the cost of transaction exchange. There are five primary stages in the Tidset-based Parallel FP-tree (TPFP-tree) algorithm: (1) create Header Table and Tidset, (2) distribute mining item set, (3) exchange transactions, (4) FP-tree construction and (5) apply FP-growth algorithm. BTP-tree is designed for grid computing because grid is heterogeneous. When using TPFP on grid we will face imbalance problems thus BTP was proposed. The work is distributed based on the computing capability of the processors. There are six stages in a BTP-tree algorithm: (1) create header table and Tidset, (2) evaluate the performance index of computing nodes, (3) distribute mining item set, (4) exchange transactions, (5) create FP-tree and (6) apply FP-growth algorithm. For huge amount of datasets, creation header table require more time.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 5 Issue 05, May-2016

Therefore, a transaction identification set (Tidset) is created at this stage to speed-up the transaction selection for future use.

Merits: It is based on cluster and grid computing it exchanges only necessary informations between clusters so the performance increased.

Demerits: If any cluster failed entire mining task will be affected.

In [9], introduced a method for finding frequent patterns from large databases in cloud computing environments. Here FP- Tree used for mining frequent items, for this FP- Tree construction disk used as a secondary memory. The system is composed of a dataset, a kernel node, n, connection node, and many computing nodes, where n is the number of clouds. Kernel node is responsible for building the FP-Tree and frequent pattern mining tasks are distributed across the computing nodes. The main purpose of using disk based FP- Tree generating method is to solve the memory lacking problem in kernel node. All reads and writes of FP- Tree into disk in serializable way for scalability and do the frequent pattern mining in huge database. This mechanism stores the whole FP-tree in many files, every file records partial FP-tree that the number of nodes is restricted in maxNodes Per File. They also introduce a tree data structure, File Table, to record the mapping of X and file name. A File Table contains a root itemset X, the count of X, a table named IDTable, for mapping file ID and subRoot item, and a table named subRootTable, for mapping subRoot item and subFile Table.

Merits: For efficient mining disk is used as a secondary memory so it solves memory problems.

Demerits: In case of disk failure, the mining task will be affected.

In [10], PARMA is introduced to achieve near-linear speedup while avoiding costly replication of data. In this multiple small random samples are created from the transactional dataset and running a mining algorithm on the samples independently and in parallel. Finally all frequent itemsets from samples are consolidated. PARMA is implemented using Map Reduce framework. It has two stages, in the first stage N samples are created with the help of first Map function and the samples are mined using first Reduce function. Second is the aggregation stage in which all the results of samples are combined to get a whole set of frequent itemsets in a database.

Merits: Instead of processing the whole dataset, it divides into sample in order to give the better and fast resuts.

In [14], two new methods are proposed for mining Big Data, *Dist-Eclat* focuses on speed while *BigFIM* is optimized to run on really large datasets. Both the methods are parallel on the Map Reduce framework where frequency thresholds can be set low. In the first method, *Dist-Eclat*, first it divides the vertical dataset into small blocks and distributes it into the possible amongst the mappers and reducers gives frequent items with count, second map reduce job is applied by distributing the frequent items to the mappers the reducer will give the frequent itemset of size k, it is pure Eclat method. Finally subtree mining applied to ecxtract all frequent itemsets. Mappers require whole dataset to mine subtrees but which can be not suitable for the given network infrastructure. So they proposed another method *BigFIM*, it is a hybrid method and first uses the Apriori algorithm to extract frequent itemsets of length k and later on it moves to Eclat when the projected databases fit in memory.

Merits: Here it is implemented on the hadoop cluster to get a better and fast performance.

In [18], FiDoop is a parallel frequent itemsets mining algorithm. The main aim of FiDoop is to build a mechanism that enables automatic parallelization, load balancing, and data distribution for parallel mining of frequent itemsets on large clusters. In FiDoop, three Map Reduce jobs are implemented to mine the large datasets. The first map reduce job is used to generate frequent one-items. Then the second map reduce job scans the dataset and prunes the infrequent items in each transaction record. Finally, third map reduce job is an important and expensive too, The main goal of each mapper is to achieve two things: i) to decompose each k-itemset obtained by the second Map Reduce job into a list of small-sized sets, where the number of each set is anywhere between 2 to k − 1 and ii) to construct an FIU-tree by merging local decomposition results with the same length. Also they developed FiDoop-HD, an extension of FiDoop, to speed up the mining performance for high-dimensional data analysis.

Merits: It uses FIUT instead of FP Tree and implemented using Map Reduce jobs to tolerate the high amount of data.

In [15], a new pre-processed k-means technique applied on BigFIM algorithm to find frequent itemsets in Big Data. ClustBigFIM uses hybrid approach, clustering using kmeans algorithm to generate Clusters and these clusters consist of datasets from huge databases, and here they used Apriori and Eclat algorithms to mine frequent itemsets from generated clusters using MapReduce programming model. The k-means algorithm is most used technique of clustering, it takes number of clusters as input, random points are chosen as centre of gravity and distance measures to calculate distance of each point from centre of gravity. Each point is assigned to only one cluster based on high intra-cluster similarity and low inter-cluster similarity. ClustBigFIM algorithm has following phases, *a. Find Clusters*, In this, clusters are found using k-means algorithm, *b. Finding k-FIs*, In this phase transaction ID list for large database cannot be handled so datasets in a cluster are mined to get local transaction ID using apriori algorithm, *c. generate single global TID list*, in this Eclat algorithm is used and thereby generating global transaction ID list from local transaction ID list *d. Mining of subtree*, finally (k+1) Frequent Items are mined using Eclat algorithm. Prefix tree generated in phase2 is mined independently by mappers and frequent itemsets are generated.

Merits: It uses preprocessing technique and works on large dataset with increased execution efficiency.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 5 Issue 05, May-2016

IV. CONCLUSION

In this paper, various ARM algorithms for frequent itemset mining are discussed. Methods proposed by various authors to extract frequent itemsets in a large dataset have also discussed. And also merits and demerits of those techniques have been discussed. It is observed that the space and time complexities are the major issues with all algorithms discussed here.

V. REFERENCES

[1] Apachehadoop. http://hadoop.apache.org/, 2013.
[2] B. Goethals. Survey on frequent pattern mining. Univ. of Helsinki, 2003.
[3] Big Data Dimensions, http://www.klarity-analytics.com/392-dimensions-of-big-data.html
[4] Big Data Spectrum by Infosys, https://www.infosys.com/cloud/resource-center/.../big-data-spectrum.pdf
[5] C. C. Aggarwal and J. Han, Frequent Pattern Mining. Cham: Springer International Publishing, 2014.
[6] Dhruba Borthakur. The hadoop distributed file system: Architecture and design. Hadoop Project Website.
[7] J. Han, H. Pei, and Y. Yin. Mining Frequent Patterns without Candidate Generation. In: Proc. Conf. on the Management of Data (SIGMOD'00, Dallas, TX). ACM Press, New York, NY, USA 2000.
[8] K. Yu and J. Zhou. Parallel TID-based frequent pattern mining algorithm on a PC cluster and grid computing system. Expert Syst. Appl., vol. 37, no. 3, pp. 2486–2494, 2010.
[9] Kawuu W. Lin, Pei-Ling Chen, Weng-Long Chang. A novel frequent pattern mining algorithm for very large databases in cloud computing environments. In 2011 IEEE International Conference on Granular Computing (GrC), Page(s):399 – 403.
[10] M. Riondato, J. A. De Brabant, R. Fonseca, and E. Upfal, PARMA: A parallel randomized algorithm for approximate association rules mining in MapReduce. in Proc. 21st ACM Int. Conf. Inf. Knowl. Manage., Maui, HI, USA, 2012, pp. 85–94.
[11] Manjit kaur, Urvashi Grag. ECLAT Algorithm for Frequent Itemsets Generation. International Journal of Computer Systems (ISSN: 2394-1065), Volume 01– Issue 03, December, 2014.
[12] P. Tan, M. Steinbach and V. Kumar, Introduction to data mining. Boston, Mass: Addison- Wesley, 2013.
[13] Rakesh Agrawal and Ramakrishnan Srikant, Fast algorithms for mining association rules in large databases. Proceedings of the 20th International Conference on Very Large Data Bases, VLDB, pages 487-499, Santiago, Chile, September 1994.
[14] Sandy Moens, Emin Aksehirli, and Bart Goethals. Frequent itemset mining for big data. In 2013 IEEE International Conference on Big Data, pages 111–118. IEEE, 2013.
[15] Sheela Gole and Bharat Tidke. Clustbigfim-Frequent Itemset Mining Of Big Data Using Pre-Processing Based On Mapreduce Framework. In International Journal in Foundations of Computer Science & Technology (IJFCST), Vol.5, No.3, May 2015.
[16] Y.-J. Tsay, T.-J. Hsu, and J.-R. Yu. FIUT: A new method for mining frequent itemsets. Inf. Sci.,vol. 179, no. 11, pp. 1724–1737, 2009.
[17] Y. Lai, S. Zhong Zhi, An efficient data mining framework on Hadoop using java persistence API, Computer and Information Technology (2010) 203–209.
[18] Yaling Xun, Jifu Zhang, and Xiao Qin. Fi Doop: Parallel Mining of Frequent Itemsets Using Map Reduce. In IEEE Transactions on Systems, Man, and Cybernetics: Systems, (Volume:PP ,Issue: 99), ISSN :2168-2216, 15 June 2015.
[19] Zaki, M. J. (2000). Scalable algorithms for association mining. IEEE Transactions on Knowledge and Data Engineering12(3): 372–390.
[20] Aggarwal, C, Li, Y, Wang, J & Wang, J 2009, 'Frequent pattern mining with uncertain data', Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, pp. 29-38.
[21] Techapichetvanich, K & Datta 2004, 'Visual Mining of Market Basket Association Rules', Lecture Notes in Computer Science, vol. 3046, pp. 479–488