ISSN 2347 - 3983
Volume 8. No. 10, October 2020
International Journal of Emerging Trends in Engineering Research
Available Online at http://www.warse.org/IJETER/static/pdf/file/ijeter248102020.pdf
https://doi.org/10.30534/ijeter/2020/248102020

Distributed Frequent Itemset Mining Using Size Based Assignment Technique

Manoj Sethi[1], **Dr. Rajni Jindal**[2]
[1]Department of CSE, Delhi Technological University, Bawana Road, Delhi-42, India, manojsethi@dce.ac.in
[2]Department of CSE, Delhi Technological University, Bawana Road, Delhi-42, India,rajnijindal@dce.ac.in

ABSTRACT

Distributed data mining is attracting researchers due to the globalisation and increase in the distributed databases. Very little work has been done in this area. Algorithms which are available, mostly first partition the database, distribute them amongst different sites for parallel processing. In the real life scenario data generated at different sites are not under control of centralised database and the numbers of transactions at each site are highly varied. Due to this some sites are heavily loaded and some sites are comparatively free. A novel approach, size based assignment, is proposed in this paper which takes care of the database size available at each site while distributing the load for finding the global frequent itemsets. It also reduces the communication load by pruning and no-broadcasting techniques. The algorithm is compared with similar algorithms on execution time. Results show that the new technique performed best amongst them in time execution.

Key words: Association Rule, Database, Distributed Mining, Frequent itemset, Partition.

1. INTRODUCTION

Association rule mining means finding interesting association, correlations and frequent patterns amongst a large number of items or objects that are contained in a transaction database, relational database or some other kind of data repository. It helps in decision making process for business purpose like in supermarkets for catalogue design, basket data analysis etc. Various algorithms have been proposed to find the frequent itemsets. Association rule mining was first introduced by R.Agrawal in 1993 [1]. After that a lot of research has been done in this area and new approaches and algorithms have been proposed.

Huge amount of data is generated by different organisations at various locations which varies in number of transactions. Size of data is increasing and the parallel frequent mining algorithms are not fit for that, rather algorithm which considers the number of transactions at each site to distribute the work load are required for optimum use of the computational capabilities available at each site. Some Distributed Association Rule Mining algorithms(DRAM) [2] are proposed in the literature. In DARM, data is stored at different sites, parallel processing is used to improve the efficiency. It finds local frequent itemsets at various sites, communicates with all other sites for finding the global frequent itemsets. There are some popular distributed mining algorithms like AprTidRec [1],“Fast Distributed Mining of Association Rules (FDM)” [2],“Optimized Distributed Association Rule Mining (ODAM)” [3] etc.The size of data has impacton the execution time[4]. Various DARM algorithms are proposed which use different data structures proposed for better performance. Some popular data structures are FP-tree introduced by FP-Growth [5]; Nodesets, an efficient data structure [6], a prearranged Trie [7]or radix, or prefix tree data structure.

The steps which are generally involved in the DARM are scanning database for finding Local frequent itemsets, storage, local pruning, sharing local counts, global pruning, finding Association rule etc. The DARM algorithms work on these problems for the improvement of the time performance. An efficient technique is presented for the frequent itemset mining considering the number of transitions at each site for load balancing. It uses best techniques for reducing the load on the network by reducing candidate sets and no-broadcasting technique along with a new Size Based Assignment (SBA) technique for polling site assignment for utilizing the under-utilised resources in a better way. It takes the advantage of local efficiency, load smoothing on unbalanced data and reduced communication cost in finding association rules in place of processing on a centralized node.

Rest of the paper has 5sections. Section2explores the literature. Section 3 explains the methodologies used and explains the size based assignment technique in detail .In section 4 new algorithm Size Based Distributed Association Rule Mining (SBDARM) is presented. Results are analysed and discussed in section 5. Lastly the work is concluded and future scope of the work is discussed in section 6.

2. RELATED WORK

Many surveys for finding association rules have been conducted [8], [9].Association rule mining is popular field of research [10],cited over22000 times as per google scholar.

The apriori based algorithms are of "anti-monotone property" [11]. This is widely used in finding the association rules. These algorithms use an approach to create and test candidate sets [1]. AprioriTID and AprioriHybrid [1]are two variations of apriori in literature. AprioriTID [1] uses the database once only for finding the frequency of the items. AprioriHybrid [1]uses both Apriori initially and AprioriTID at the end.[5] proposed FP-Growth, a tree structure which is created after database scan for mining frequent itemsets.

Algorithm FIN, "Fast mining frequent itemset" [6]uses Nodesets data structure. This structure uses PPC-Treeto store information of node. It generates the structure based onpostorder or preorder of the node. DFIN [12] algorithm suggested a new structure diffnodesets which is based on nodeset."DT-DPM (Decomposition Transaction for Distributed Pattern Mining)"[13]framework is proposed by researcher. It integrates "Density-Based Spatial Clustering of Applications and distributed computing represented, CPU multi-cores and Single CPU for solving pattern mining problems". Performance of any algorithm [14]is also effected by the number of nodes in the distributed data system. With the increase of the transactions or nodes, the performance improves. Execution time increases when the number of nodesincreases but number of transactions reduces[15]. FDM algorithm performance improves when optimised with FP-Growth and DiffSet-mining.

2.1.Distributed Data Mining Algorithms

A parallel algorithm that runs on a distributed database with uniform capabilities at different sites requires to divide the workload equally for best load [16].Count Distribution (CD) [17] is a simple algorithm where data is parallelized and apriori algorithm runs parallel where local support count is found for each itemset and then communicated to each other site and hence the global frequent itemsets are found by all the sites. AprTidRec, an algorithm based on apriori algorithm was proposed in 2011 [11]. It is different from apriori because it deploy only the joint step but no pruning step. It creates a record structure called tidRec and has lesser execution time than apriori algorithm. FDM [2]mainly uses apriori and CD algorithms. This finds the locally large itemsets, which are sent to the assigned polling sites and then the global counts are found to finally find global frequent itemsets. The total support count message exchange is just O(n).

DMA [18] is another algorithm for distributed association rule mining, which generates a small candidate sets and O(n) messages are exchanged for n sites in a distributed database. ODAM (Optimized Distributed Association Rule Mining) algorithm for distributed data association rule mining proposed by [3]. After discovering the global frequent-1 itemsets, it removes the infrequent ones and inserts the transactions and their count in a temporary file which is then used to find the frequent itemsets of larger lengths. [19] proposed PFIN algorithm for mining frequent itemsets using nodeset structure. It breaks the large problem into sub-problems, executed in parallel. Using the map-reduce approach also, many algorithms can be processed in distributed environment, like the MRPrepost [20] algorithm gives the processing of prepost algorithm on the Hadoop platform. Nadar proposed nagFIN algorithm using new nagNodeset [21] data structure which is based on the nodes in the prefix tree. There are some negative association [22] in data which are also interesting and useful. [23] proposed algorithm based on "optimized matrix computation for Multi party data computation"which has some challenges. Applications of association rule mining [24] in Large and Dispersed database are businesses, defence, public safety, GIS, medical diagnosis, Hospital etc. As the data is updated on regular basis and bringing all at one place is not feasible and time consuming and mining data must be up-to-date[25] otherwise it affects the decisions. Distributed data mining also helps [26] in maintaining privacy, reducing transmission cost, and sharing resources like memory.

3. METHEDOLOGY

3.1. Distributed Association Rule Mining

This research focuses on DARM, where data is not distributed rather generated in distributed manner at different sites and number of transaction varies. FDM is one of the popular DARM where Apriori algorithm generates the local frequent itemsets at each site. In this work, the proposed algorithm uses some of the properties of FDM along with some other proposed techniques to find the association rule in the distributed data. New techniques of no-broadcasting and size based assignment of polling site are proposed for reduction of communication overload and load balancing amongst the sites. The problem statement [2] is as under:

Let DB be a database, contains$I = \{i_1, i_2, \dots, i_m\}$ set of items. T is a transaction of items where$T \subseteq I$. Itemset$Z \subseteq I$, belongs to T only if $Z \subseteq T$. An association rule(AR)is represented [2] as $\Rightarrow Y$, where,$Z \subseteq I \ and \ Y \subseteq I$ and$Z \cap Y = \phi$.The AR$Z \Rightarrow Y$ present in the DB, with a confidence 'c'means the probability of a transaction containing Z also contains Y is 'c'. The AR $Z \Rightarrow Y$ with support 's' means that the probability of a transaction contains both Z and Y is 's'. The task here is to find all such ARs with support greater than support threshold and confidence greater than minimum confidence threshold.

Z.sup isits support count for an itemset Z. If its support count of Z is not less than the minimum support threshold then Z is frequent. k-itemset is k sized itemset. The problem is AR mining [10] is: (i) "to find all frequent itemsets for the given minimum support threshold value", and (ii) "to generate the association rules using the frequent itemsets".The focus of mining is on the development of some efficient method for the (i) [10] as main cost is involved in (i).Distributed algorithm rule mining [2]is stated as :

To find the association rulesin a distributed databaseDB with D transactionsstored at n-sites $S_1, S_2, \dots, S_n$with different data partitions$\{DB_1, DB_2, \dots, DB_n\}$respectively. $D_i$is the size of the data partitions DB_iwherei = 1, 2, . . . , n. The support count is written as Z.sup, and$Z.Sup_i \ in \ partition \ DB_i$. For each site

S_i, $Z.Sup_i$is the local support count of Z and Z.supis the global support count. Zis also globally large itemset if $Z.sup \geq s \times D$; correspondingly, Zis locally large itemset at site S_i, if $Z.sup_i \geq s \times D_i$. LetL be the globally large itemset [2] in the database, and $L_{(k)}$be the globally large k-sized itemsets.

3.2. Candidate Set Pruning

Pruning of the candidate sets is done in order to reduce the size of the candidate sets for all k-itemsets where k=1…n[2] at each site. It is based on the assumption that if an itemset is not local frequent at-least at one of the sites, it can't be global frequent. Pruning removes all such itemsets which are not locally large i.e. their support count is less than the required minimum support count. This helps in reducing the load on the communication channel by reducing the size of the candidate sets and improves the performance.

3.3.No-Broadcasting Technique

Broadcasting of the local frequent itemsets to all sites is a heavy load on the communication network. No-broadcasting reduces the load on the network. All local frequent itemsets are sent to a dedicated site for assignment of polling site for finding the global frequent itemsets. There is no-broadcasting of the local frequent itemsets. Suppose there are 5 sites then all sites send frequent itemsets to other four sites means 5 x 4 = 20 packets but in no-broadcasting all 5 sites send local frequent itemsets to one site only so 5 x 1 = 5 packets are being sent on the communication channel. With the increase of the number of sites there is a big reduction in the packets communicated over network in the no-broadcasting technique and the performance of the algorithm improves.

3.4. SizeBased Assignment (SBA) Technique for Polling Site Assignment

In the distributed setup,data is generated or createdat different locations. The number of transactions at various sites differ [14] from a few hundred to millions of transactions.In this setup, resources at each site is also limited and are distributed. Data mining requires great amounts of resources [27] so technique for flexible distribution of work load amongst sites needs to be developed. The sites with little number of transactions are less occupied as they require less memory, computational capabilities and time for maintaining data, scanning for frequent itemsets and maintaining candidate sets. In this paper a new technique SBA for polling site assignment is presented for the real distributed database considering the below assumptions.

Assumptions:
- Database is distributed around the globe
- Data is gathered at different locations
- Number of transactions at each site differ
- No site is having data size more than the double of the average data size

Considering the above assumptions, new technique SBA is proposed to assign the polling sites to the locally large itemsets received by a designated site. The poling site finds the globally large itemsets from the itemsets received. The novel technique takes care of the sites with the large data partitions and distribute load considering the number of transitions at each site and balance the load.

Definition : Size Based Assignment Technique:
For sites $S = \{S_1, S_2, \ldots, S_n\}$ and candidate sets $\{CG_1, CG_2, \ldots, CG_n\}$sent by n sites.
Total Transactions $TT = \{T_1, T_2, \ldots, T_n\}$
For size k-itemsets
Average Transactions percent per site $AT = \frac{100}{n}$

Actual number of Transaction in percent at site

$$PS_i = \frac{T_i}{\sum T_i} x\, 100$$

Load difference at site in percentS_i is $\Delta_i = AT - PS_i$

For all k-sized itemsets, complete candidate set

$$CG = \{CG_1, CG_2, \ldots, CG_n\}$$

CG' = Complete Candidate sets received from all sites without any duplicates
$CG' = \{CG_1 \cup CG_2 \cup \ldots \cup CG_n\}$

Average candidate sets at each site in percent

$$ACG = \frac{CG'}{n}$$

Arrange sites as per the partition size S'
Local Frequent itemsets assigned to the polling site S_i is = $\lceil ACG + \Delta_i \text{x } CG \rceil$

After finding the average frequent itemsets to be allocated to each site, from sites S' arranged in the order of the number of transactions, the assignment of polling sites to the frequent itemsets is done in this order by assigning upper integer value of average candidate set plus load difference. Assignment continues till the entire candidate set exhausted This way the highly loaded sites are assigned nil or very little number of locally large itemsets for finding globally large itemsets. This technique assigns the load inversely proportional to the site load and hence balance the load of polling station.
Let there be five sites S_1, S_2, S_3, S_4, S_5 having transactions 10% , 15%, 20%, 25% and 30% respectively. The candidate sets sent by sites$S_1\{1, 4, \}$, $S_2\{2, 4\ 6, \}$, $S_3\{3, 8, 17, 16\}$, $S_4\{12, 17, 1, 18\}$, $S_5\{2, 8, 19, 11\}$. Applying SBA technique, the ordered candidate set is $\{1, 2, 3, 4, 6, 8, 11, 12, 16, 17, 18, 19\}$ with 12 locally large itemsets and ordered site set is $\{S_1, S_2, S_3, S_4, S_5\}$ with average transactions 20% as there are 5 sites.

The load difference at S_1 is 10% so local frequent itemsets to be assigned to site S_1 is
ceiling integer $\left(\frac{12}{5} + \frac{10}{100} \text{x } 12\right) = 4$ i.e. { 1, 2, 3, 4} items

Similarly assignment to S_2 is
ceiling integer$\left(\frac{12}{5} + \frac{5}{100} \text{x } 12\right) = 3$i.e. {6, 8, 11} items

Assignment to S_3 is
ceiling integer$\left(\frac{12}{5} + \frac{0}{100} \text{x } 12\right) = 3$i.e. {12, 16, 17} items

Assignment to S_4 is
ceiling integer$\left(\frac{12}{5} + \frac{-5}{100} \text{x } 12\right) = 2$. i.e.{18, 19} items

Assignment to S_5 is Zeroitems as candidate list exhausts.

The sites with smalldata partition or with less number of transactions are not fully occupied. These sites have less processing scan and memory needs, as compared to the sites with more transactions. ProposedSBA technique assigns load inversely proportional to the site occupancy by considering the partition size and balances the load.

4. SBDARM: THE PROPOSED ALGORITHM

This section discusses the Size Based Distributed Association Rule Mining (SBDARM) algorithm. It uses a novel technique of size based assignment (SBA) of polling site for finding globally large itemsets based on the data size available at each site. Globally large itemsets are found by sites which are less occupied hence increase the overall computational capabilities and improve the performance. It uses local as well as global pruning to reduce the candidate sets. There is no-broadcasting of candidate sets which further reduces the load on the communication network.

Symbol Description [2]
s –minimum Support;
D - Total transactions ;
L_k – Globally large k-itemsets;
Z.sup - Global support count of Z;
CA_k – Candidate sets size k;
D_i - transactions in partition S_i;
$GL_{i(k)}$–globally large itemset size k at S_i;
$CG_{i(k)}$ - Candidate sets size k at site S_i;
$LL_{i(k)}$ - Locally large size-kitemsets in $CG_{i(k)}$;
$Z.sup_i$ – Local support count of $Z\ at\ S_i$
$LP_{i(k)}$ – Local pruning k-itemset at site S_i

Algorithm-1: Size Based Assignment Technique (SBA)
Input: Locally large k-itemsets from each siteCG_i and data size of all sites $\boldsymbol{S}_i(i = 1,2, \dots n)$
Output: Assigned Polling sites list

```
1. $if\ k = 1\ itemsets$
2. $for\ all\ site$
3. $find\ the\ average\ number\ of\ transactions$
        $each\ site\ in\ percentage\ AT$
4. Compute the transaction size at each site $Si$
        $in\ percentage\ PS_i$
5. $Load\ difference\ at\ site\ in\ Si\ in\ percent\ \Delta_i$
6. $for\ all\ locally\ large\ itemsets$
7. $arrange\ in\ order\ removing\ duplicates\ LL_{i(k)}$
8. $compute\ the\ average\ candidate\ sets\ per\ site$
9. $in\ percentage\ ACG$
10. $for\ all\ sites$
11. $for\ all\ ordered\ candidates$
12. $assign\ the\ polling\ site\ in\ size\ based\ assignment$
        $average\ candidates\ sets\ plus\ load\ difference$
        $in\ percent$
13. $broadcast\ the\ polling\ site\ list\ for\ k-candidate\ sets$
```

Algorithm-2: Size Based Distributed Association Rule Mining (SBDARM)
Input: database $DB_i(i = 1,2, \dots n)$
Output: Globally large itemsets.
Method: Running algorithm for all k-itemsets for k=1..n, on all partitions

```
1. $for\ all\ sites$
2. $for\ k = 1$
3. $find\ the\ local\ frequent\ itemset\ T_{i(1)}$
4. $find\ the\ support\ count\ T_{i(1)}$
5. $for\ k > 1$
6. $find\ the\ size-k\ itemset\ usin\ apriorigen$
7. $for\ all\ itemsets\ Z\ belongs\ to\ frequent\ itemsets$
    $T_{i(k)} generate\ local\ pruning\ list$
8. $if\ support\ count\ is\ locally\ large\ then$
9. $for\ all\ sites$
10. $insert\ itemset\ into\ locally\ large\ list\ LL_{i(k)}$
11. $for\ all\ sites$
12. $communicate\ locally\ large\ list\ LL_{i(k)} and\ data\ size$
        $to\ site\ for\ assignment\ of\ polling\ site$
13. $using\ SBA\ get\ list\ of\ polling\ sites$
14. $for\ all\ sites,$
15. $send\ locally\ large\ LL_{i(k)}\ to\ polling\ site\ S_m$
16. $for\ all\ itemsets\ Z\ belong\ to\ local\ pruning\ LP$
17. $send\ polling\ request\ for\ itemset\ Z\ to\ all\ sites$
18. $all\ sites\ reply\ polling\ request\ from\ (T_{i(k)})$
19. $send\ support\ counts\ Z.sup_m$
20. $for\ all\ itemsets\ Z\ in\ the\ polling\ set\ LP_{i(k)}$
21. $receive\ support\ count\ Z.sup_m\ from\ all\ sites$
22. $for\ all\ the\ itemsets$
23. $calculate\ global\ support\ Z.\text{sup}\ by$
        $sumup\ of\ all\ local\ support\ where$
        $if\ Z.Sup > the\ global\ support\ threshold$
24. $Add\ toglobal\ frequent\ itemset\ G_{i(k)}$
25. $broadcast\ global\ frequent\ itemset\ G_{i(k)}$ ;
26. $if(k = 1)\ remove\_infrequent(DB_i)$;
27. $Generate\ set\ of\ all\ globally\ large\ itemsets$
28. $return\ globally\ large\ k-itemset\ L_{(k)}$
```

The steps are discussed below:

(i) Database DB_iat all sites are scanned, local frequent itemsets of size-1 are found.For k>1 locally large itemsets are found using apriorigen. This generates locally large itemsets from all partitions and make candidate set $CG_{i(k)}$.

If $CG_{i(k)}$ is empty, no generation and process stops. (line 1-7)

(ii) Local pruningis done to generate candidate sets,locally large k-itemsets$LL_{i(k)}$having the count greater than the minimum support threshold.(line 8-10)

(iii) The locally large itemset $LL_{i(k)}$are communicated to the site which assigns the polling sites to the locally large itemsets. Size based assignment algorithm receives list of sites with number of transactions and locally large items communicated by each site. It communicates the polling sites for locally large k-sized itemsets $LL_{i(k)}$sites S_i.(line 11-13, call algorithm 1)

(iv) All the sites$S_i$send the local counts for the locally large items $LL_{i(k)}$ to the polling sites assigned in the last step. Polling sites store all information about the itemsets in $LP_{i(k)}$ and Z.large_sites. (line 14-17)

(v) Each Polling site receives counts, computes global counts for assigned locally large itemsets$LL_{i(k)}$. It generates the global large itemset, stores in $G_{i(k)}$ after removing the itemsets having counts less than the support threshold value. Then globally large itemsets are communicated to all sites. (line 18-25)

(vi) All home site receives the global frequent itemsets, update and remove all infrequent 1-itemset. In the next pass home sites find the 2-itemset, i.e. locally large size k (k =2…n), repeat the process. Remove all infrequent k-itemsets..(line 26-28)

4.1 Efficiency at Each Site

There are n number of sites $\{S_1, S_2, \dots, S_n\}$and data is partitioned and stored, called distributed database DB_i. Sites generate the local frequent k-itemset (k= 1..n) using efficient algorithm. The polling sites are assigned on the basis of the partition size where site with less number of transactions are assigned more local frequent itemsets for finding global frequent itemsets and sites handling bigger partition size are assigned less workload considering the number of transactions at each site. Sites with small data size uses less memory, capabilities etc in handling the data, so less occupied and the same is taken care by the proposed algorithm. Size based assignment technique is developed which considers the load on each site while assigning the polling sites for finding the globally large itemsets. Using the size based technique, it allocates less load to the sites have large data partition and more load to less occupied sites with small data partition. The proposed technique is best for the unbalanced data partitions for the effective use of the resources and balancing the load for finding the globally large items using locally large items. This technique utilises all resources and as all the sites participate as per their availability so there is no extra load on a centralised or any other site in unbalanced way. This ensures a good amount of parallelism in the real distributed database where centralised database has no control on the partitions.

4.2. Low Communication Overhead

The algorithm also takes care of the load on the communication channelby reducing the size of the candidate sets by pruning at each site. All the sites first find the frequent itemsets and then through pruning process remove the frequent itemsets having counts less than the required support counts to become eligible for communication and may not be globally frequent.

Let frequent 2-itemset at site3be{ad, eg, jg, ht }. After applying pruning process, removing not eligible itemsets whose support count is less than the minimum support thresholdi.e.{ad, jg}are removed. The reduced set {eg, ht } after pruning is communicated to the site for the assignment of polling sites.The technique reduces the size of the candidates sets to half and reduces the communication overhead. The algorithm uses no-broadcasting technique where all sites send candidate sets to a polling assignment site in place of broadcasting it to all sites. If reduces the number of candidate set communication to O(n) messages and hence there is less load on the communication network. This technique is network efficient with reduced size of the data communicated.

5. RESULTS AND DISSCUSSION

5.1. Running Environment

The setup used to perform the evaluation of the algorithms and comparing the same with the existing algorithms on various parameters is explained. Experiments are performed on the setup of five nodes or sites with windows 10 Operating systems,8 GB RAM, HDD 500 GB, 64 bit system with clock 3.20 GHz loaded with JDK 1.7.

Datasets: The algorithms are run on mushroom, connect, chessand T10I4D100K datasets. The datasets are available for research on data mining on the FIMI data repository [28]. Mushroom dataset is created using attributes of species mushrooms, connect prepared based on UCI,chess is game situations and T10I4D100K is a synthetic database generated using the IBM Quest generator. The datasetsspecifications are given in Table 1.

Table 1: Datasets and their specifications

Dataset	Total Trans.	Number of Items	Average Length	Type	File Size
Mushroom	8124	119	23	Dense	570 KB
Connect	67557	129	43	Dense	9.3 MB
Chess	3196	75	37	Dense	342 KB
T10I4D100K	1,00,000	1000	40	Sparse	4 MB

Data is partitioned and stored at five sites with varying size. Data partitions with size available on each site, based on number of transactions are shown in Table 2.

Table2: Partition size at different sites

	Partition Size	Mushroom	Connect	Chess	T10I4D100K
DB_1	10%	812	6756	320	10000
DB_2	15%	1219	10134	479	15000
DB_3	20%	1625	13511	639	20000
DB_4	25%	2031	16889	799	25000
DB_5	30%	2437	20267	959	30000
Total	100%	8124	67557	3196	100000

5.2. Performance Analysis

The new proposed algorithm SBDARM implemented and executed for execution performance comparison with some of the existing algorithms FDM and PFIN. The SBDARM algorithm effectively uses the resources at all the sites, reduces the local load and communication load. It gives best throughput by using pruning techniques for reducing candidate sets for communication along with the size based assignment technique. There is no-broadcasting of frequent itemsets by all the sites rather all sites communicate the frequent itemsets to one site for the assignment of the polling site. This reduces the number of messages and hence load on the communication network.

In the experiments, algorithms are compared on time of execution where sites have varying size of the data partitions shown in Table 2.In the first experiment all algorithms are executed on dataset mushroom with varying minimum support threshold values i.e. 10%, 20%, 30%, 40%, 50%, and 60%. Table 3 shows the local frequent 1-itemsets generated at each site on the mushroom datasets. Similarly in the second experiment these are run on connect dataset with same minimum support threshold 10%, to 60%. Third experiment is performed on chess dataset with 10%, 15%, 20%, 25%, 30%, 35% minimum support threshold. Lastly on T10I4D100K dataset with minimum support threshold 0.1%, 0.2%, 0.3%, 0.4%, 0.5%, and 0.6%. The local frequent 1-temsets generated by the proposed algorithm at each site on connect, chess and T10I4D100K datasets are shown in Table 4, 5, and 6 respectively. The data available at each site is different but frequent 1-itemsets generated are almost same and it reduces with the increase the support threshold.

Table 3: Local frequent 1-itemsets generated at each partition on Mushroom dataset

	----- Support Count Threshold (%) -------					
Partition	10	20	30	40	50	60
DB1	50	34	26	21	16	14
DB2	48	37	30	25	18	14
DB3	39	37	26	23	17	15
DB4	51	38	27	19	18	12
DB5	44	41	31	22	19	14

Table 4: Local frequent 1-itemsets generated at each partition on Connect dataset

	----- Support Count Threshold (%) ------					
Partition	10	20	30	40	50	60
DB1	66	57	46	43	39	38
DB2	69	56	46	43	38	35
DB3	69	54	47	42	40	38
DB4	68	56	49	42	37	37
DB5	70	56	46	43	40	36

Table 5 : Local frequent 1-itemsets generated at each partition on Chess dataset

	----- Support Count Threshold (%) -----					
Partition	10	15	20	25	30	35
DB1	52	50	47	46	44	42
DB2	52	50	48	46	42	41
DB3	55	53	48	46	44	41
DB4	61	58	58	53	50	43
DB5	66	57	54	52	51	49

Table 6 : Local frequent 1-itemsets generated at each partition on T10I4D100K dataset

	---- Support Count Threshold (%) ------					
Partition	0.1	0.2	0.3	0.4	0.5	0.6
DB1	789	736	682	621	560	515
DB2	793	742	684	630	559	516
DB3	798	740	689	633	563	520
DB4	794	738	691	625	566	515
DB5	794	743	690	628	561	516

The frequent itemsets are sent to one site only using no-broadcast technique for the assignment of the polling sites. In the proposed algorithm the polling site assignment is done using size based assignment technique. Table 7-10 show that the local frequent 1-itemsets assignment of polling site for finding the global frequent itemsets by the SBDARM algorithm. The same process is repeated for k-itemsets for all k>1. This assignments balance the load on the sites as it allocates the load for finding the global frequent itemsets i.e. assignment of polling sites inversely proportional to the partition size on sites. In the other two algorithms the assignments are not based on the size of the partition rather using some hash function or random, count distribution. The assignment in FDM, PFIN increases the load on the already occupied site and load balancing is poor. This size based assignment technique is effective in the distributed data environment with varied data size and it reduces the time of execution.

Table 7 : Pruning site assignment by SBDARM to local frequent 1-itemsets on Mushroom dataset

	---- Support Count Threshold (%) ------					
Partition	10	20	30	40	50	60
DB1	18	14	10	9	6	6
DB2	15	12	8	7	5	5
DB3	12	9	7	6	4	4
DB4	9	7	5	5	3	3
DB5	5	3	2	0	2	0
Total	59	45	32	27	20	17
Globally Large	56	43	28	21	13	8

Table 8 : Pruning site assignment by SBDARM to local frequent 1-itemsets on Connect dataset

	------- Support Count Threshold (%) ------					
Partition	10	20	30	40	50	60
DB1	23	19	15	14	14	13
DB2	19	16	13	12	11	11
DB3	15	13	10	9	9	9
DB4	12	10	8	7	7	7
DB5	6	5	4	3	3	1
Total	75	63	50	45	44	41
Globally Large	73	59	46	41	38	36

Table 9 : Pruning site assignment by SBDARM to local frequent 1-itemsets on Chess dataset

	------ Support Count Threshold (%) ------					
Partition	10	15	20	25	30	35
DB1	21	19	18	17	17	17
DB2	17	16	15	14	14	14
DB3	14	13	12	12	12	11
DB4	11	10	9	9	9	9
DB5	5	4	6	4	4	3
Total	68	62	60	56	56	54
Globally Large	61	57	54	51	50	45

Table 10: Pruning Site assignment by SBDARM to local frequent 1-itemsets on T10I4D100K dataset

	--- Support Count Threshold (%) ----					
Partition	0.1	0.2	0.3	0.4	0.5	0.6
DB1	240	223	209	189	172	157
DB2	200	186	174	158	143	131
DB3	160	149	140	126	115	105
DB4	120	112	105	95	86	79
DB5	80	73	68	62	55	50
Total	800	743	696	630	571	522
Globally Large	796	740	691	628	568	516

Figures 1 - 4 show that the performance of the proposed algorithm on all datasets outperform the other two algorithms intime execution. The time performance of SBDARM is best as the load is balanced amongst the sites. The transactions at each site differ means the resource utilization also differs. The sites with more number of transactions takes more time for data scan, use more memory, and more processing. SBDARM utilises the load differencesas edge over other algorithms by assigningmore load to less loaded sites as compared to highly loaded sites. Some sites with less number of transactions,are having less load of processing, data scan, generate less number of candidate sets and are comparatively less occupied. This is the best load balancing at each site by the assignment of polling sites for finding global frequent itemset inversely proportional to the data partition size available at each site. The number of local frequent k-itemsets for k> 1, are further reduced and the SBA algorithm further balance the load amongst the sites.

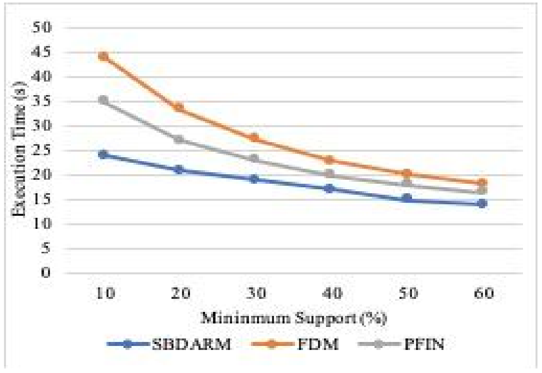

Figure 1: Execution time on Mushroom dataset

With the increase of the support threshold less number of frequent itemsets are generated and execution time of all the algorithms also reduce. The performance of the proposed algorithm takes less time although the time difference reduces as number of global frequent itemsets also reduce. The difference in the performance is also due to the reduced communication load, local pruning and no-broadcasting technique which reduces the load on the network and reduces the time delay. In Figure 4 dataset used is sparse so when value of k increases to 2,3,...n the number of frequent itemsets generated reduce, therefore for low minimum support threshold, the execution time is not very high as compared to the execution time for the high minimum support threshold for all the algorithms.

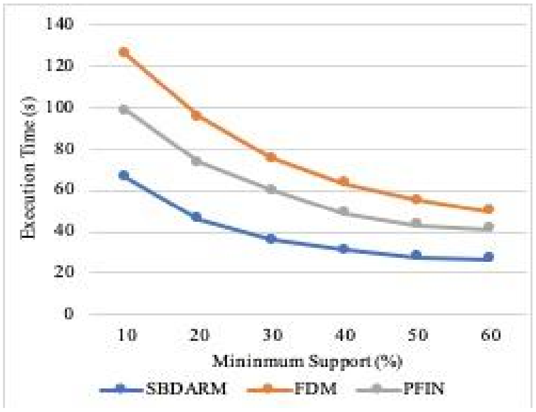

Figure 2: Execution time on Connect dataset

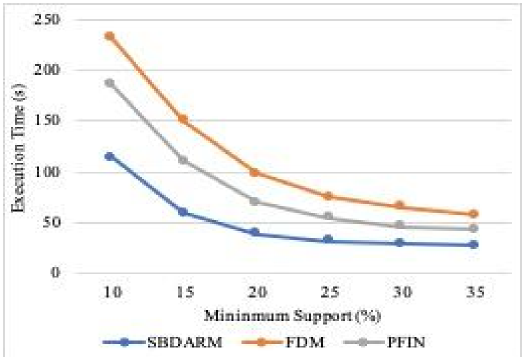

Figure 3: Execution time on Chess dataset

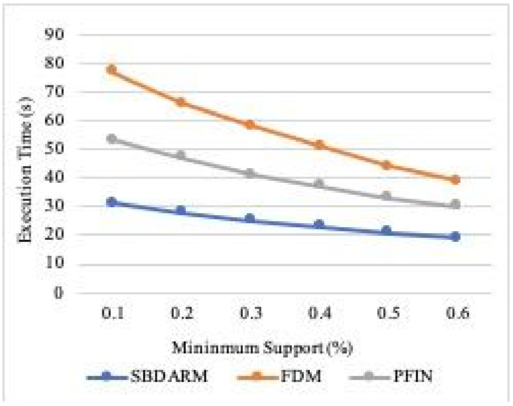

Figure4: Execution time on T10I4D100K dataset

The analysis shows that the proposed algorithm SBDARM performs best in low and high minimum support threshold in all four comparisons with PFIN and FDM having varying partition sizes. It shows SBDARM outperforms other two algorithms with low minimum support threshold. It uses the advantages of the no-broadcasting by reducing communication and size based assignment reducing load on heavy loaded sites and pruning reducing candidate sets.

6. CONCLUSIONS

New algorithm SBDARMis proposed for finding frequent itemsets in the distributed data where size of the partitions are varying in size. The proposed algorithm applies the technique of local pruning, no-broadcasting and size based assignment of the polling sites. The algorithm performance is evaluated on four datasets on the 5-node setup with varying minimum support threshold. The experiments are performed on the data partitions with varying number of transactions on each site. The performance of the algorithm is compared with the FDMand PFIN algorithm for distributed data mining.

Algorithm SBDARM outperforms other algorithms on the time of execution comparisons. The new size based assignment technique is effective in the distributed environment where data is generated at different sites and data is skewed. i.e. highly imbalanced in terms of number of transactions at each site. It performs best as the data skewness is not effecting the performance and is well adjusted. In addition to the reduction in candidate sets by pruning, the proposed no-broadcasting technique reduces the communication load and improve the execution time of the proposed algorithm.

In the future the research can be further extended for mining in larger setupwith more number of sites and large datasets. The resources and the capabilities available at each site can also be considered while allocating load to different sites for further improvement.

REFERENCES

1. Agrawal R. and Ramakrishnan Srikant, "**Fast algorithms for mining association rules**", in *Proc. International Conference on Very Large Scale Data Base*, 1994, pp. 487-499.
2. D. W. Cheung, Jiawei Han, V.T. Ng, A. W. Fu, and Yongjian Fu, "**A fast distributed algorithm for mining association rules**”, *Fourth International Conference on Parallel and Distributed Information Systems*, IEEE, 1996.
3. Ashrafi, Mafruz Zaman, David Taniar, and Kate Smith, "**ODAM: An optimized distributed association rule mining algorithm**”, *IEEE distributed systems online*, 2004.
4. P. Naresh and Dr. R. Suguna, “**Association rule mining algorithms on large and small datasets: a comparative study**”, in *Proc. International Conference on Intelligent Computing and Control Systems (ICICCS 2019) IEEE: CFP19K34-ART*, pp. 587-592.
5. Han, Jiawei, Jian Pei, and Yiwen Yin., "**Mining frequent patterns without candidate generation**”, in *Proc. of the 2000 ACM SIGMOD international conference on Management of data*, vol. 29. no. 2, pp. 1-12, 2000.
6. Deng, Zhi-Hong and Sheng-Long Lv., "**Fast mining frequent itemsets using Nodesets**”, *Expert Systems with Applications*, vol. 41 no. 10, pp. 4505-4512, 2014.
7. F. Bodon and L. Rónyai, “**Trie: An alternative data structure for data mining algorithm**”, *Mathematical and Computer Modelling*, vol. 38, Issue 7–9, pp. 739-751, 2003.
8. Vinaya Sawant and Ketan Shah, “**A survey of distributed association rule mining algorithms**”, *Journal of Emerging Trends in Computing and Information Sciences*, vol. 5, pp. 391-398, 2014.
9. Han, Jiawei, et al., "**Frequent pattern mining: current status and future directions**”, *Data Mining and Knowledge Discovery*, vol.15.1, pp. 55-86, 2007.
10. Agrawal R., Tomasz Imieliński, and Arun Swami, "**Mining association rules between sets of items in large databases**", *ACM SIGMOD Record* 22.2, pp. 207-216, 1993.
11. Ailing Wang, "**An improved distributed mining algorithm of association rules**”, *Journal of Convergence Information Technology*, vol. 6, no. 4, pp.118-122, 2011.
12. Deng, Zhi-Hong, "**DiffNodesets: An efficient structure for fast mining frequent itemsets**”, *Applied Soft Computing*, vol. 41, pp. 214-223, 2016.
13. Vinaya Sawant and Ketan Shah, “**Performance evaluation of distributed association rule mining algorithms**”, *7th International Conference on Communication, Computing and Virtualization, Procedia Computer Science* 79, pp. 127-134, 2016.
14. Asma Belhadi, Youcef Djenouri, Jerry Chun-Wei Linand Alberto Cano, “**A general-purpose distributed pattern mining system**”, *Springer-Applied Intelligence*, vol. 50, pp. 2647–2662, 2020.
15. George Gatuha and Tao Jiang, “**Smart frequent itemsets mining algorithm based on FP-tree and DIFFset data structures**”, *Turkish Journal of Electrical Engineering & Computer Sciences*, vol. 25, pp. 2096-2107, 2017.
16. Van Quoc Phuong Huynh and Josef Küng, “**FPO Tree and DP3 algorithm for distributed parallel frequent**

Itemsets mining”,*Expert Systems With Applications*, vol.140, 112874, 2020.

17. Agrawal R. and Shafer John C., "**Parallel mining of association rules**", *IEEE Transactions on Knowledge & Data Engineering*, vol. 8, Issue 6, pp. 962-969, 1996.
18. David Cheung, Vincent T.Y. Ng, Ada W. Fu and Yongjian Fu, "**Efficient mining of association rules in distributed databases**”, *IEEE Transactions on Knowledge and Data Engineering*, vol. 8, no. 6, pp. 911-922, 1996.
19. Chen Lin and Junzhong Gu, “**PFIN: A parallel frequent itemset mining algorithm using nodesets**”, *International Journal of Database Theory and Application*, vol. 9, no.6, pp. 81-92, 2016.
20. Liao, Jinggui, Yuelong Zhao and Saiqin Long., "**MRPrePost—A parallel algorithm adapted for mining big data**", *IEEE Workshop on Electronics, Computer and Applications(IWECA)*, 2014.
21. Nader Aryabarzan, Behrouz Minaei-Bidgoli and Mohammad Teshnehlab, “**negFIN: An efficient algorithm for fast mining frequent itemsets**”,*Expert System and Applications*, vol. 105, pp. 129-143, 2018.
22. NVS Pavan Kumar, JKR Sastry and K Raja Sekhara Rao, “**On mining incremental databases for regular and frequent patterns**”, *International Journal of Emerging Trends in Engineering Research*, vol. 7, no. 9, pp. 291–305, 2019.
23. Jun Liu, Yuan Tian, Yu Zhou, Yang Xiao and Nirwan Ansari, “**Privacy preserving distributed data mining based on secure multi-party computation**”, *Computer Communications*, vol. 153, pp. 208-216, 2020.
24. A. Pavithra,and S. Dhanaraj,“**Comparative study of effective performance of association rule mining in different databases**”,in*Proc.of International Conference on Data Science and AnalyticsICDSA 17*, 2017
25. NVS Pavan Kumar, JKR Sastry and K Raja Sekhara Rao, “**Mining negative frequent regular itemsets from data streams**” *International Journal of Emerging Trends in Engineering Research*, vol. 7(8), pp. 85 – 98, 2019.
26. V. Devasekhar, and P. Natarajan, “**Multi-agent distributed data mining: challenges and research directions**”*International Journal on Emerging Technologies*, vol. 11(4), 233–239, 2020.
27. Ammar Alhaj Ali, Pavel Varacha,Said Krayem,Petr Zacekand Andrzej Urbanek,“**Distributed data mining systems: techniques, approaches and algorithms**”, *MATEC Web of Conference 210, 04038, CSCC,* 2018.
28. **Fimidataset repository**: Online Portal http://fimi.ua.ac.be/data/