ISSN 2278-3091
Volume 8, No.4, July – August 2019
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse64842019.pdf
https://doi.org/10.30534/ijatcse/2019/64842019

Mining Distributed Databases for Negative Associations from Regular and Frequent Patterns

NVS Pavan Kumar[1], Dr.JKR Sastry[2], Dr. K Raja Sekhara Rao[3]
[1]Koneru Lakshmaiah Education Foundation, Vaddeswaram, Andhra Pradesh, India, nvspavankumar@kluniversity,in
[2]Koneru Lakshmaiah Education Foundation, Vaddeswaram, Andhra Pradesh, India, drsastry@kluniversity,in
3Usharama institute of Engineering and Technology, Andhra Pradesh, India, krr_it@yahoo.co.in

ABSTRACT

Most of the transactional data stored in distributed databases, especially when region wise business conducted. Extensive knowledge about the business is hidden in these databases, the discovery of which will give immense power to the decision-makers. Most of the decisions are taken around frequently occurring patterns and the positive associations that exist among those patterns. Mining a single database can yield the patterns of sales happening at different distributed locations. The regularity of the occurrence of sales pattern is equally as important as frequent patterns. Negative associations also reveal interesting discovery about the business in addition to positive associations and yet time negative associations among the regularly and frequently occurring patterns having a heavy bearing on the Business. It is also important to study the occurrence of the patterns at all the distributed locations and determine the regularity or global nature of the patterns**.**

In this paper, Algorithms presented using which mining of negative association that exist among regular and frequent patterns mined presented considering both local and Global Locations.

Key words: Mining distributed databases, negative associations, Regular, frequent patterns, and Local and Global patterns, Maximum regularity and minimum support

1. INTRODUCTION TO DISTRIBUTED DATABASES

Data mining algorithms deal predominantly with Relational databases and simple data formats, typically flat files, or delimited file. There is an increasing amount of focus on mining complex and advanced data types such as object-oriented, spatial, and temporal data. Another aspect of this growth and evolution of data mining systems is the move from stand-alone systems using centralized and local computational resources towards supporting increased levels of distribution. As data mining technology matures and moves from a theoretical domain to the practitioner's arena, there is an emerging realization that distribution is a factor to be accounted for especially when pattern mining undertaken.

Databases in today's information age inherently distributed. Organizations that operate in global markets need to perform data mining on distributed data sources which are either homogeneous or heterogeneous and require cohesive and integrated knowledge from this data. The distributed database management systems characterized by geographical separation of users from the data sources. This inherent distribution of data sources and large volumes of data involved inevitably leads to exorbitant communications costs. Therefore, the traditional data mining model involving the co-location of users, data, and computational resources is inadequate when dealing with distributed environments.

The development of the distribution of the data leads to the emergence of distributed data mining. The need to address specific issues associated with the application of data mining in distributed computing environments is the primary objective of distributed data mining. Broadly, data mining environments consist of users, data, hardware, and the mining software, and this includes both the mining algorithms and any other associated programs. Distributed data mining addresses the impact of the distribution of users, software, and computational resources on the data mining process.

In the DDM literature, one of two assumptions commonly adopted as to how data is distributed across sites either homogeneously or heterogeneously. Both viewpoints adopt the conceptual viewpoint that the data tables at each site are partitions of a single global table. In the homogeneous case, the global table horizontally partitioned. The tables at each site are subsets of the global table. All the tables at all locations have the same attributes. In the heterogeneous case, the table is vertically partitioned. Each site contains a collection of Tables that have different columns. However, each tuple at each site is assumed to contain a unique identifier to facilitate joining the like tables placed at different locations. The key column attribute can also be used to join different tables existing in the same location.

The enormity and high dimensionality of datasets typically available as input to the problem of association rule discovery make it an ideal problem for solving multiple processors in parallel. The primary reasons are the memory and CPU speed limitations faced by single processors. Thus it is critical to design efficient parallel algorithms to do the task.

Another reason for the parallel algorithm comes from the fact that many transaction databases are already available in parallel

databases or distributed at multiple sites, to begin-with. The cost of bringing them all to one side or one computer for the serial discovery of association rules can be prohibitively expensive. For compute-intensive applications, parallelization is an obvious means for improving performance and achieving scalability. A variety of techniques may be used to distribute the workload involved in data mining over multiple processors. The data distributed either horizontally or vertically or sometimes, both types of distributed are adapted based on the way data generated at different locations. The companies that sell products in different regions adapt the database regionally.

Frequent pattern mining or regular pattern mining or generation leads to valuable hidden knowledge in the distribution in the form of association rules. In the beginning, these pattern mining techniques started with multiple scans of a database with fewer amounts of transactions of the typical market basket analysis database as the prototype. Later the database size increased in due course and required more efficient methods introduced came into the picture.

Scalability is the major obstacle in these algorithms to handle huge volumes of transactional databases. The concept of distributed memory arises at this point and in many ways, experimented. One fundamental approach of these is with multiple processors having both individual memory with each processor and a shared pool of memory accessible by all the processors to communicate with each other. There is a hurdle of increased I/O and suffers from bottlenecks relating to the availability of bandwidth for communicating the data.

In reality, the data gets distributed into different locations attached to a single processor. The data, as such, can be portioned either horizontally or vertically. Transactions happen at each location and therefore, are stored at the locations where they captured. Transactions that happen at different locations are captured and stored at respective locations. The transactions contained at each location shall have the items that can be sold at any of the locations, making it difficult to find the regularity, frequency, and maximally.

Most of the focus was on discovering the frequent patterns, and the positive association that exists among the frequent patterns, Regular occurrence of the patterns is equally important in addition to frequent patterns as they reveal interesting trends are happening in the business or interesting behavior exhibited by the Users. Sometimes negative associations also exist among the regular and frequent itemsets. Ignorance of negative associations sometimes shall have a devastating effect.

In a distributed system, the processing of the transactions undertaken at geographically distributed locations for finding regular and frequent patterns that have negative association among the patterns. Finding the patterns locally and accumulating them centrally to find the negatively associated patterns is a complex process. Thus processing should take place at distributed locations for local pattern-finding and processing done at the central location for finding the negative associations. Thus both the processing and data are distributed at different locations.

2. PROBLEM DEFINITION

Thus the problem is to find regular, frequent itemsets that are negatively associated when the transactional data distributed over several servers located at different sites, and all the servers are connected through a network as shown in Figure. 1

3. RELATED WORK

Right from Apriori by [Srikanth and Agarwal 1993] [1], many have presented that data existed at distributed locations and parallel algorithms required for processing huge databases. Agarwal, R et al., [2] presented two algorithms, namely Apriori and AprioriTiD, for discovering association rules that exist among the item sets in a large database. These algorithms differ in the way candidate itemsets are counted and generated. They have presented the features of an algorithm achieved by combining the two algorithms that they have presented. They have shown that the hybrid algorithm scales–up linearly concerning the number of transactions contained in the database.

Another typical approach by [Savasere, 1995] [3] partition algorithm with two passes logically divides the transactional database into n number of different partitions to convert into the vertical format, i.e., item wise transactions. Candidates for the second pass prepared by merging frequent local item set to formulate global frequent item set.

Another milestone in this area by [Park, 1995] [4] proposed a level-wise pruning approach on Apriori algorithm using dynamic hashing techniques and pruning. In this approach, approximate support of n+1 itemset was pre-computed in n itemset counting. This kind of process leads to reducing the size of the database at each level by pruning n itemset level that does not appear in n+1 level.

[Muller 1995] [5] Discussed in his thesis about the partitioning and Apriori. They worked on sequential patterns and sequential association rules in the name of sequential efficient association rules. In this algorithm, the hash tree replaced with the candidate prefix tree. Hence it generates a candidate set with length k+1, k+2, etc. instead of length k.

[Brin 1997] [6] Developed dynamic itemset counting DIC similar to Apriori. The database divided into n-partitions suitable for memory uses local support counts. All these results later developed to generate global support count and prune the itemset of less support count than the user given threshold. This approach identifies many positive local frequent itemset but not frequent global itemset when the database is not homogeneous. They have also proposed a random partitioning algorithm to avoid partitioning skew.

[R. Agrawal, H. Mannila 1997] [7] have presented in their book, different mining algorithms for mining frequent itemsets from distributed databases, using the support as an interesting measure. They have not as such touched on the issue of regularity and negative associations.

[Zaki 1997]. [8] Presented excellent maxClique algorithm improvised from many such algorithms. The major strength of all the algorithms Eclat, MaxClat Clique, and MaxClique, etc. lies in the vertical format of the database they used. In this approach, simple intersection and union operations give us frequent itemset by calculating the support count of k-itemset in an easier way. MaxClique finds the maximal patterns directly with few intersection operations. The above algorithms use the combination of the two fundamental approaches bottom-up and hybrid in finding frequent itemset.

Parallel algorithms are the alternative solution for the sequential bottleneck of the current methods. Also, it provides scope for scalability to any extent and improved response time for large datasets. This kind of process enhances the minimization of I/O operations too. Nonuniform memory access NUMA is developed based on the two fundamental memory sharing mechanisms, namely Dynamic memory mechanism DMM and Shared memory Systems SMP for hybrid and hierarchical mechanisms. Static load balancing cannot support the heterogeneous systems with multi-user systems because these systems require dynamic load balancing. Whenever load on a processor is high in the multiprocessor system, with a minimal additional cost of data movement is to be allocated to a less loaded pre-processor.

[Bandon 2004] [9] Implemented Apriori algorithm on a rooted directed tree rather than a hash tree. He has given importance to implementation details and developed to the implementation details and developed a fast algorithm. They followed the same procedure of dividing a large database into N partitions instead of loading into limited main memory, which earlier versions of Apriori did. They are distributed these partitions on to N nodes to compute local k-item set based on local support count. Thus the results are concluded with a global support count by pruning all the k-item set that does not satisfy the global support count.

[Shengnan Cong 2005] [10] Presented selective sampling technique in their framework to resolve the load balancing problem while finding sequential patterns in large databases based on pattern growth algorithm. They experimented both on frequent patterns, and sequential patterns with the framework developed using divide and conquered algorithm. The projected databases are independent in both FP growth and Prefix Span algorithms. The Estimated mining time of a data set achieved by running a small sample set of data from the set. Random sampling is done by selecting a fraction of the sequence of transactions and tested with the same fraction of support threshold. Both the methods may not be suitable in generating a better heuristic. The proposed selective sampling discards some of the most frequent and non-frequent item set from the sample. Also, similar discarding of a few elements from the tail of the sequences results in finding a good sample to run the algorithm.

[Yenbin Ye 2006] [11] Proposed Bandon's version of the Apriori algorithm to mine frequent using parallel input of very large transactional database on a parallel computer**.** Most of the work is focussed on frequent items till 2009 and no focus made till such time on regular items which are those that occur in regular intervals. Regular items are the most important than the frequent items, the occurrence of which has no time limitation. [Tanbeer 2008] [15] are the first group of authors who have focussed on regular itemsets. They have proposed a "Regular pattern tree," which is a tree structure to discover regular patterns. The algorithm scans the database twice. In the first set, regularity and support values of the item sets are determined, and in the second scan, a regular Pattern tree constructed. The process adopted by them is similar to cyclic and periodic patterns.

Mining distributed databases most of the times, require parallel processing done on a single server or distributed over several servers. High performing mining carried through parallel processing. [Hu J. and Yang-Li X 2008] [13] Have carried a detailed survey and determine all the methods that have been proposed in the literature and figured out the defects and disadvantages of those existing algorithms. The main drawback is a known consideration of the existence of multiple levels within the datasets. They have developed a frequent pattern forest that connects up all the frequent patterns through a tree structure. They have proposed to construct multiple trees with the established interconnection between the tress. They have also proposed a parallel mining algorithm that considered parallel data and parallel tasks combined. The FP-forest method considers both the issues of parallel data and parallel tasks.

Data divided among different distributed processors situated on different nodes; each of the distributed databases is known as sub-database. An FP structure is built on each of the nodes separately. One of the nodes in the distributed set-up synchronizes the tress resident on different nodes. An MF-tree assigned to the mining task is initiated at a particular node dynamically. The tasks at each of the node undertake the mining of association rules. The tasks implanted frequent growth mining techniques using a Depth-first search technique to scan over the trees.

[GUO Yi-mig 2010] [14] presented a vertical format based algorithm to find all frequent itemset with a single scan of the database using AND operations of the transaction ids in follow up data mining process in contrast with Apriori algorithm which requires multiple database scans. The key point of algorithm efficiency lies in judging the ability of non-frequent itemset to formulate candidate sets. Also, all the Tids with k-itemset allows us to find support degree of k+1 itemset without another scan of the database. The AND operation of Tids will result in the support count to prune non-frequent item set.

[Vijay Kumar 2011, [Vijay Kumar 2012], [15, 16] developed many algorithms towards regular itemset mining. The regular itemset describes the pattern occurrence frequency as well as occurrence behavior. The measure used for regularity is the period of two consecutive occurrences of an itemset in the database. While using a vertical format, the algorithm can efficiently identify the frequency and regularity of a pattern.

[Philippe Fournier-Viger, 2014] [17] Presented FHN algorithm to find high utility patterns with positive and negative unit

profits. In this, they used different strategies without generating candidate sets. Unlike earlier algorithms to find frequent high items with less profit, these algorithms focus on the less frequent items with high profit and negative profit itemset. This method requires two scans of the database. In the first scan, all patterns above *minutil* (minimum utility threshold) defined. Later in the second scan, the estimated utility co-occurrence structure is built in ascending order of the total order of itemset, and the procedure continues to find positive High Utility Itemset. This kind of process is then modified to find a negative item based on the threshold by making the second term of pruning zero without affecting the rest of the process.

Frequently occurring itemsets represents correlations among the itemsets having items identified with different weights. Generally speaking, no item in the data set assigned a different weight. All the items in the data set shall have the same weight. [Luca Cagliero 2014] [18] have opined that discovering rare data correlations is more important than discovering frequent correlations. They have presented a method aimed at discovering rare and weighted itemsets/patterns — a quality measure which is interesting proposed by the authors. The mining process proposed by them considers the measures for identifying the rare patterns that are interesting.

[Zengyon He 2016] [19] Discussed recently disproportionate frequencies in a different class labeled data sets. Discriminative property of the earlier algorithms suffered from redundancy. To remove the redundancy, they introduced a CDPM (Conditional Descriptive Pattern Mining) algorithm and successfully proved the valuable results.

[Shehal J Patil 2016] [20] Described infrequent itemset mining in their algorithms by assigning weights to the patterns. The proposed algorithms to find itemset with a frequency less than or equivalent to the maximum user given threshold. They successfully find minimal frequent MIWI, and weighted item set IWI the two major categories of infrequent item set. Support count and correlation infrequent itemset depend on the direct association like itemset mining in market basket analysis, whereas indirect association also is a vital measure in association rule mining. Text mining could be a better example of this because for many words; there are synonyms and antonyms on different occasions in a similar meaning possess indirect association. Pattern growth algorithms are used in this process to satisfy downward closure property by adding higher weight items in decreasing fashion. The concept of minimal, infrequent weight item set is to generate minimal patterns and stops as and when the first infrequent itemset found.

[Philippe Fournier-Viger 2017] [21] Published a survey paper on sequential pattern mining. In this, they discussed right from what is a sequence to today's methods in an orderly way and is very much useful for further research. In this, they discussed several algorithms and their behavior. As a part of it, they described SPTID is similar to partition method whereas SPEAR algorithm is to SEAR. But SPINC Algorithm reduces the complexity of the second scan. All these based on partition techniques with various approaches. According to them, algorithms that follow a depth-first search or breadth-first search with different measures to define the support threshold are effective. Also, many approaches that follow horizontal databases as well as vertical databases exist. There are few efforts towards negative pattern mining draw our interest, namely Negative-GSP, PNSP, e-NSP, etc. There is much scope to find negative patterns in all directions and requires massive effort to bring hidden knowledge into the mainstream.

To improve the performance and efficiency of the algorithm in distributed databases, we follow the parallel and distributed approach. There exists n number of systems namely s= {s0, s1, s2…sn-1} in the homogeneous system. Each site corresponds to a portion of the database dbi where i is in [0.. n-1]. At every portion of the database db= {db0, db1, db2…dbn-1} local regularity threshold, λ_l applied to find the positive regular items in the distribution. All these values are tabulated into a header table (table-3, table-4) including periods and regularities as shown in the tables. Maximum of these local regularities becomes the global regularity of the given item set. This global regularity must satisfy the formula $Reg(x) < \lambda_{max\_reg}$ to declare it a positive regular pattern. The process of finding negative regular item set begins at this point by inspecting the intersection of the transactions of positive regular item set. If there are no common transactions between any two positive regular itemsets, then they could be declared as negative regular item set or contradicting patterns.

[Yonxi Wu 2017] [22] Developed algorithm to improve performance and efficiency of the algorithm in distributed databases we follow parallel and distributed approach. There exists n number of systems namely s= {s0, s1, s2…sn-1} in the homogeneous system. Each site corresponds to a portion of the database dbi where i is in [0.. n-1]. At every portion of the database db= {db0, db1, db2…dbn-1} local regularity threshold, λ_l am applied to find the positive regular items in the distribution. All these values are tabulated into a header table (table-3, table-4) including periods and regularities as shown in the tables. Maximum of these local regularities becomes the global regularity of the given itemset.

This global regularity must satisfy the formula $Reg(x) < \lambda_{max\_reg}$ to declare it a positive regular pattern. The process of finding negative regular itemset begins at this point by inspecting the intersection of the transactions of a positive regular itemset. If there are no common transactions between any two positive regular itemsets, then they could be declared as negative regular itemset or contradicting patterns and then to find nonoverlapping patterns in the sequence database — meaningful subsequences obtained by combining flexible wild cards or gap constraints for specific requirements. Earlier methods do not support Apriori property ends with finding too general or restrictive patterns due to the variation in the support ratio of a pattern and its subpatterns. They developed and tested their NOSEP algorithm for non-overlapping sequences over DNA sequences. This kind of process is an Apriori property based method combines gap constraints and yields better results than state of the art mining techniques.

Many algorithms approach presented in the literature for mining frequent items sets suitable for different applications.

Not much emphasis is given for regular and frequent itemsets considering the negative associations [23] [24] [25] [26] [27] [28][29][30][31]]32][33].

4. COMPARATIVE ANALYSIS

Comparison of exiting algorithms has been made considering various aspects considered for generating negative associations considering regular and frequent itemsets to assess the adequacy of those algorithms**.** Table 1 shows the comparison, from the table it can be seen that none of the existing algorithms are dealing with the most important aspects of the negative associations that include regularity, positive/negative associations, frequency and interestingness measures.

5. INVESTIGATIONS AND FINDINGS

5.1 Architectural design of Experimental distributed database

IBM supplied 100,000 records related to sales transactions. Out of these 50,000 records are placed on one server, and the remaining 50,000 records placed on the second server. An Algorithms is developed and placed at each server, which can process the negatively associated patterns existing at each of the locations. The process placed in the client collects the negatively associated patterns at each of the location and process further to get overall negatively associated patterns. The architecture of the distributed processing implemented to determine overall patterns shown in Figure 1.

5.2 Algorithmic Approaches for finding negative associations from distributed databases considering the regularity of the item sets

A parallel and distributed approach proposes to improve the performance and efficiency of the algorithm. There exists n number of systems namely s= {s0, s1, s2…sn-1} in the homogeneous system. Each site corresponds to a portion of the database dbi where i is in [0.. n-1]. At every portion of the database db= {db0, db1, db2…dbn-1} local regularity threshold, λ_l applied to find the positive regular items in the distribution. All these values are tabulated into a header table (table-3, table-4) including periods and regularities as shown in the tables. Maximum of these local regularities becomes the global regularity of the given item set.

This global regularity must satisfy the formula $Reg(x) < \lambda_{max\_reg}$ to declare it a positive regular pattern. The process of finding negative regular item set begins at this point by inspecting the intersection of the transactions of positive regular item set. If there are no common transactions between any two positive regular itemsets, then they could be declared as negative regular item set or contradicting patterns.

In chapter-3 we have shown the way negative patterns or the patterns that show negative associations considering {Regular), {Regular, Frequent}, {Regular, Frequent, Maximal} could be generated considering standalone database. In this chapter-4, we present below the way {Regular), {Regular, Frequent}, {Regular, Frequent, Maximal} patterns that show the negative associations generated considering distributed Database. A sample of records considered for experimentation and reporting.

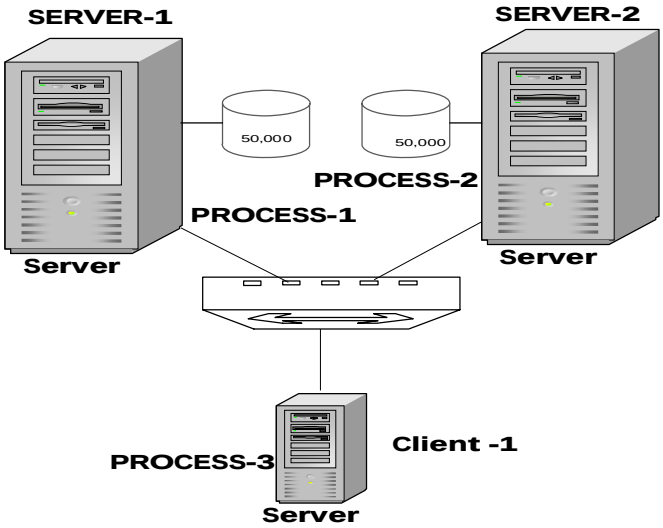

Figure 1 : Distributed database and Mining Systems

5.3 Algorithmic Approaches for finding negative associations from distributed databases considering the regularity and frequency of the itemsets

Mining Negative patterns based on regularity and frequency at Location – 1

Algorithm

1. Read the support value that dictates the threshold value of the frequency of the patterns and also the regularity as defined by the user
2. Read the data in flat file / DBMS Table into an Array as shown in Table 2
3. Convert the data in table 2 into the vertical format as shown in Table 3
4. Prune the Initial Irregular Items and non-frequent items
5. Repeat the following process

Consider the current Item

Select next item and prune it if it is not regular or frequent and go to the next item (self-Loop).

If the next item is regular and frequent, get the intersection of the transactions of the current item and next item

If the intersection is null, then enter the current and next items into a negative set array

If the intersection is not null, then get the common elements and see if the count of elements is > regularity threshold decided by the user

If the common elements satisfy the regularity and frequency constraint, then add the common elements into a vertical table as a new row as they are regular and frequent at the end of the vertical table.

If the common elements do not satisfy the regularity or frequency constraint, then ignore them

If all the elements in the vertical table are exhausted, then convert the next item which is next to the current item as the current item and then LOOP

If all the elements in the vertical table are not exhausted, then move to the next item and LOOP When no item left in the vertical table, then the process terminates

Experimentation with sample data and results

The proposed algorithm applied to the IBM supplied data set, and the results obtained are detailed below:

Step-1

Transaction IDS added to the IBM supplied data. Sample first 20 records of IBM supplied data shown in Table 2. Here frequency is the count of transactions in which the item set appears.

Step-3

Find the first regular item by pruning all the previous items whose regularity is > User given Maximum Regularity threshold (γ_{min_reg}) and also the item that satisfies the minimum support value. Here regularity implies the relative occurrence of the Item, computed as the distance between two successive transactions. Consider (γ_{min_reg}) = 4 and the minimum frequency by 3 for sample data. The First regular and frequent Item is called Previous-Item. The list of items that will be leftover in the vertical format table shown in Table 4.

Table 2: Transaction data at Location-1

TrId	Item Set
1	2 5 7
2	1 2 3 4 5
3	3 6 8 9
4	2 5 7 9
5	1 2 3 4 5
6	3 8 9
7	2 5 7 9
8	1 2 3 4
9	3 6 8 9
10	2 9
11	1 2 3 4 5
12	3 8 9

Step-2

Convert the Transaction filled IBM data to Vertical format. Table 3 shows the transactional data in a vertical format. In the vertical data format, for each of the Item, in the data repository, the transactions that contain the Items are found and mapped. The frequency of an item is the count of transactions in which the item appears.

Table 3: Transaction Data in the vertical format at Location-1

Itemset	Trids
1	2 5 8 11
2	1 2 4 5 7 8 10 11
3	2 3 5 6 8 9 11 12
4	2 5 8 11
5	1 2 4 5 7 11
6	3 9
7	1 4 7
8	3 6 9 12
9	3 4 6 7 9 10 12

Step-4

Consider each item starting from Previous-item and repeat the following procedure.

1. Consider the next item and let that be current-item
2. Find if the current-item is regular and frequent. If the current-item is not regular or frequent prune it.
3. If the current- item is regular and frequent, find Intersection of the transactions of the current item with the previous-item.
4. If the intersection is null, then add the Item set into the negative item-set list.
5. If the intersection is not null, find the regularity considering the common elements.
6. If the regularity is < (γ_{min_reg}), then add a previous item and the current item set along with its related transaction as an additional record to the vertical database since they are regularly and frequently associated.
7. If the next item is not the lost entry in the vertical table, Make Current Item as the next Item and loop.
8. If the next item is the last in the Vertical table, then Previous Item = Previous Item +1 and then Loop.

Table 4: Pruning the Items in the vertical table till the first regular and frequent Item is traced max regularity =4 support count =4

#	Itemset	Trids	Periods	Max Regularity	Support Count
1	1	2 5 8 11	3 3 3	3	4
2	2	1 2 4 5 7 8 10 11	1 2 1 2 1 2 1	2	8
3	3	2 3 5 6 8 9 11 12	1 2 1 2 1 2 1	2	8
4	4	2 5 8 11	3 3 3	3	4
5	5	1 2 4 5 7 11	1 2 1 2 4	4	6
6	8	3 6 9 12	3 3 3	3	4
7	9	3 4 6 7 9 10 12	1 2 1 2 1 2	2	7
8	2 5	1 2 4 5 7 11	1 2 1 2 4	4	6

After this step completed, Table 5 shows the pruned items, and the negatively associated items shown in Table 6**,** and positively associated items shown in Table 7.

Table 5: List of the Pruned item list

Itemset	Trids	Periods	Max Regularity	Support Count
6	3 9	3 6 3	6	2
7	1 4 7	1 3 3 5	5	3

Table 6: Negatively Associated Item set

#	Itemset1	Itemset2
1	1	8
2	1	9
3	2	8
4	4	8
5	4	9
6	5	8
7	8	2 5
8	1 4	8 9

Table 7: Positively Associated Item set

#	Itemset	Trids	Periods	Max Regularity	Support Count
1	1	2 5 8 11	3 3 3	3	4
2	2	1 2 4 5 7 8 10 11	1 2 1 2 1 2 1	2	8
3	3	2 3 5 6 8 9 11 12	1 2 1 2 1 2 1	2	8
4	4	2 5 8 11	3 3 3	3	4
5	5	1 2 4 5 7 11	1 2 1 2 4	4	6
6	8	3 6 9 12	3 3 3	3	4
7	9	3 4 6 7 9 10 12	1 2 1 2 1 2	2	7
8	2 5	1 2 4 5 7 11	1 2 1 2 4	4	6

Pseudocode

```
Algorithm Boolean Regularity (is, TIds, λmax_reg,m)

{  // let TIdsfirst and TIdslast are the first and last transactions of is
    is_First =TIdsfirst-0;
    if ( is_First> □nax_reg )          return F;
    is_Reg=is_First;
    for p= TIdsnext  to TIdIslast
       {   is_next= TIdIs pnext - TIdIsp;
           if(is_next)>is_Reg then
           is_Reg=is_next;
        }
     Is_next=m- TIdslast;
     If (is_next)>is_Reg then  is_Reg=is_Next;
     if (is_Reg }> λmax_reg) return F;
     return T;
}

Algorithm Vertical_Conversion (Tfirst, Tlast)

 {

for each item ik in I
{
   VDBik = {∅};
   for each tp  in Tfirst to Tlast do
          {
            for each item ik in I
                   {
              if( ik ⊆ p)  VDBik = VDBik ∪ tP;
                   }
          }
 }

Algorithm PD_PRISM(Tfirst, Tlast, VDBS, nS)
{
  for each ik in I
     {
          for each j ⊆ k
             {
                  if(Regularity (ij, TIdk, □max_reg,nS)= = F)  prune ik ;
                  else
                  VDB= VDB ∪ { ik , TIdIk };
                  n=n+1;
              }
     }

Algorithm PD_Result(Ifirst,Ilast)
{
          Result ={∅}
          for all ik in ifirst to ilast
          {

              for all ij in inext to ilast
                   {

                     if (Ikj = VDB)   continue;
                     if(j ⊆ k)  continue;
                     if (TIdIk∩ TrIdIj) = = {;} )
                    Result= Result ∪ { ik , ij }
                     return ;
                   }
          }
}

Algorithm PD_NPRISM(Tfirst, Tlast, nS)
{
          // nS is the size of sight S
           Vertical_Conversion (Tfirst, Tlast);

          for k=ifirst to ilast
          {
           if  ( Regularity( k, TIdk, □max_reg, nS ) = = T)  break;
          else
```

```
                    prune k;
            }
    // To find remaining regular items
            for  j= knext to n
            {
            if  ( Regularity( j, TIdj, □max_reg ,nS) = = F) prune j;
            else
            {
                                TIDIkj = TIDIk ∪ TIDIj ;

                        If (  Regularity (ikj, TIdIkj, λmax_reg,nS) = = T)

                        if (Ikj = VDB) continue;

                         else

                         VDB=VDB ∪ { ikj , TIdIkj }
            }
          }
        PD_Result(Ifirst,Ilast);
}
```

Mining Negative patterns based on regularity and frequency at Location 2

Algorithm

1. Read the support value that dictates the threshold value of the frequency of the patterns and also the regularity as defined by the user
2. Read the data in flat file / DBMS Table into an Array as shown in Table 8
3. Convert the data in table 7 into the vertical format as shown in Table 8
4. Prune the Initial Irregular Items and non-frequent items
5. Repeat the following process
   Consider the current Item
   Select next item and prune it if it is not regular or frequent and go to the next item (self-Loop).
   If the next item is regular and frequent, get the intersection of the transactions of the current item and next item
   If the intersection is null, then enter the current and next items into a negative set array
   If the intersection is not null, then get the common elements and see if the count of elements is > regularity threshold decided by the user
   If the common elements satisfy the regularity and frequency constraint, then add the common elements into a vertical table as a new row as they are regular and frequent at the end of the vertical table.
   If the common elements do not satisfy the regularity or frequency constraint, then ignore them
   If all the elements in the vertical table are exhausted, then convert the next item which is next to the current item as the current item and then LOOP
   If all the elements in the vertical table are not exhausted, then move to the next item and LOOP
   When no item left in the vertical table, then the process terminates

Experimentation with sample data and results

The proposed algorithm applied on the IBM supplied data set, and the following results applied

Step-1

Transaction IDS added to the IBM supplied data. Sample first 20 records of IBM supplied data shown in Table 8. Here frequency is the count of transactions in which the item set appears

Table 8: Sample Transaction ID supplied IBM Data at Location 2

#	Trid	Itemset
1	1	1 2 4 6
2	2	3 5 7 8 9
3	3	1 2 4 5
4	4	2 5 6 8 9
5	5	1 4 5
6	6	2 5 7 8 9
7	7	1 4 5
8	8	2 3 5 6 8 9
9	9	1 3 4 5
10	10	2 3 5 7 8 9
11	11	1 2 3 4 5
12	12	2 3 5 7

Step-2

Convert the Transaction filled IBM data to Vertical format. Table 9 shows the data in vertical format. In the vertical data format, for each of the Item, in the data repository, the transactions that contain the Items are found and mapped. The frequency of an item is the count of transactions in which the item appears.

Table 9 : Sample Transaction Data in vertical format for Location 2

#	Itemset	Trids
1	1	1 3 5 7 9 11
2	2	1 3 4 6 8 10 11 12
3	3	2 8 9 10 11 12
4	4	1 3 5 7 9 11
5	5	2 3 4 5 6 7 8 9 10 11 12
6	6	1 4 8
7	7	2 6 10 12
8	8	2 4 6 8 10
9	9	2 4 6 8 10

Step-3

Find the first regular item by pruning all the previous items whose regularity is > User given Maximum Regularity threshold (γ_{min_reg}) and also the item that satisfies the minimum support value.

Here regularity implies the relative occurrence of the Item, computed as the distance between two successive transactions. Consider (γ_{min_reg}) = 4 and the minimum frequency by 3 for sample data. The First regular and frequent Item is called Previous-Item. The list of items that will be leftover in the vertical format table shown in Table 10

Table 10 : Pruning the Items in the vertical table until the first regular and frequent Item traced

#	Itemset	Trids	Periods	Max Regularity	Support Count
1	1	1 3 5 7 9 11	2 2 2 2 2	2	6
2	2	1 3 4 6 8 10 11 12	2 1 2 2 2 1 1	2	8
3	4	1 3 5 7 9 11	2 2 2 2 2	2	6
4	5	2 3 4 5 6 7 8 9 10 11 12	1 1 1 1 1 1 1 1 1	1	11
5	8	2 4 6 8 10	2 2 2 2	2	5
6	9	2 4 6 8 10	2 2 2 2	2	5

Step-4

Consider each item starting from Previous-item and repeat the following procedure.

1. Consider the next item and let that be current-item
2. Find if the current-item is regular and frequent. If the current-item is not regular or frequent prune it.
3. If the current- item is regular and frequent, find Intersection of the transactions of the current item with the previous-item.
4. If the intersection is null, then add the Item set into the negative item-set list.
5. If the intersection is not null, find the regularity considering the common elements.
6. If the regularity is < (γ_{min_reg}), then add the previous item and the current item set along with its related transaction as an additional record to the vertical database since they are regularly and frequently associated.
7. If the next item is not the lost entry in the vertical table, Make Current Item as the next Item and loop.
8. If the next item is the last in the Vertical table, then Previous Item = Previous Item +1 and then Loop.

After this step completed, Table 11 shows the pruned itemsets and the negatively associated items shown in Table 12, and positively associated items detailed in Table 13.

Table 11: List of the Pruned item list

Itemset	Trids	Perio ds	Max Regularity	Support Count
5	1 2 4 5 7 11	1 1 2 1 2 4 1	4	6
6	3 9	3 6 3	6	2
7	1 4 7	1 3 3 5	5	3

Table 12: Negatively Associated Item set max regularity =3 support count =5

SL. No	Itemset1	Itemset2
1	1	8
2	1	9
3	4	8
4	4	9
5	8	1-4
6	9	1-4

Table 13: Positively Associated Item set

#	Itemset	Trids	Periods	Max Regularity	Support Count
1	1	1 3 5 7 9 11	2 2 2 2 2	2	6
2	2	1 3 4 6 8 10 11 12	2 1 2 2 2 1 1	2	8
3	4	1 3 5 7 9 11	2 2 2 2 2	2	6
4	5	2 3 4 5 6 7 8 9 10 11 12	1 1 1 1 1 1 1 1 1	1	11
5	8	2 4 6 8 10	2 2 2 2	2	5
6	9	2 4 6 8 10	2 2 2 2	2	5
7	1 4	1 3 5 7 9 11	2 2 2 2 2	2	6
8	2 5	3 4 6 8 10 11 12	1 2 2 2 1 1	2	7

Pseudocode

Algorithm Boolean Regularity (is, TIds, □max_reg,m)

```
{  // let TIdsfirst and TIdslast are the first and last transactions of is
     is_First =TIdsfirst-0;
     if ( is_First> λnax_reg )  return F;
     is_Reg=is_First;
     for p= TIdsnext  to TIdIslast
         {   is_next= TIdIs pnext - TIdIsp;
             if(is_next)>is_Reg then
             is_Reg=is_next;
         }
      Is_next=m- TIdslast;
      If (is_next)>is_Reg then  is_Reg=is_Next;
      if (is_Reg }> λmax_reg) return F;
      return T;
}
```

```
Algorithm Vertical Conversion (Tfirst, Tlast)

 {

for each item ik in I
{
   VDBik = {∅};
   for each tp  in Tfirst to Tlast do
          {
            for each item ik in I
                  {
              if( ik ⊆ p)  VDBik = VDBik ∪ tP;
                  }
          }
 }

Algorithm PD_PRISM(Tfirst, Tlast, VDBS, nS)
{
  for each ik in I
     {
          for each j ⊆ k
             {
                 if(Regularity (ij, TIdk, □max_reg,nS)= = F)  prune ik ;
                 else
              VDB= VDB ∪ { ik , TIdIk };
          n=n+1;
           }
    }

Algorithm PD_Result(Ifirst,Ilast)
{
          Result ={∅}
          for all ik in ifirst to ilast
          {

              for all ij in inext to ilast
                    {

                      if (Ikj = VDB)   continue;
                      if(j ⊆ k)  continue;
                      if (TIdIk∩ TrIdIj) = = {;} )
                   Result= Result ∪ { ik , ij }
                   return ;
                  }
          }
}

Algorithm PD_NPRISM(Tfirst, Tlast, nS)
{
          // nS is the size of sight S
           Vertical_Conversion (Tfirst, Tlast);

          for k=ifirst to ilast
          {
           if  ( Regularity( k, TIdk, λmax_reg, nS )= = T)  break;
          else
                    prune k;
          }
  // To find remaining regular items
          for  j= knext to n
          {
          if  ( Regularity( j, TIdj, λmax_reg ,nS) = = F) prune j;
          else
                    {
                              TIDIkj = TIDIk ∪ TIDIj ;

                    If (  Regularity (ikj, TIdIkj, λmax_reg,nS) = = T)

                     if (Ikj = VDB) continue;

                          else

                              VDB=VDB ∪ { ikj , TIdIkj }
                    }
          }
       PD_Result(Ifirst,Ilast);
}
```

Composing the Negative Global patterns at the Client

At each of the individual distributed locations, negatively associated item sets generated, and the same are combined to make an overall negative pattern set. The same set of patterns can appear at both the locations or a different set of patens may exist at each location. The combined patterns list shown in Table 14

Algorithm

1. Select the negative pattern item pattern at server - 1 if this pattern is not available in Server-2 to add the patterns to the global pattern list.
2. If the same negative pattern is available at both the servers, add the pattern in server-1 into global pattern list and ignore the pattern in Serv-2 list.
3. Select the negative pattern item pattern at server - 2 if this pattern is not available in Server-1 to add the patterns to the global pattern list.

The above algorithm applied, and the Global negative patterns arrived at are shown in Table 14.

Table 14: Negatively Associated Item set

#	Itemset1	Itemset2	Action
1	1	8	Retained
2	1	9	Retained
3	2	8	Retained
4	4	8	Retained
5	4	9	merged
6	5	8	Retained
7	8	2 5	Retained
8	1 4	8 9	Retained
1	1	8	Merged
2	1	9	Merged
3	4	8	Merged
4	4	9	Merged
5	8	1-4	Retained
6	9	1-4	Retained

6. DATA ANALYSIS AND INTERPRETATIONS

IBM supplied 1,00,000 data records about commercial transactions conducted by a mal out of which two sets of 40,000 records have been selected and placed in two different servers. The data at both the servers processed and the Negative regular, frequent patterns determined.

Table 15: Negative, Regular, frequent associations at Location-1 and Location-2

#	Location-1 1-40,000 Records		Location-2 40,001 – 80,000	
	Itemset1	Itemset2	Itemset1	Itemset2
1	5	108	1	122
2	7	723	1	440
3	18	347	28	992
4	18	742	31	665
5	28	624	33	527
6	58	319	43	948
7	66	157	51	517
8	66	464	58	319
9	66	659	58	422
10	66	762	58	639
11	80	224	90	469
12	80	859	97	887
13	80	943	100	336
14	95	314	104	157
15	95	447	104	351
16	95	992	105	181
17	762	823	105	403
18	784	943	105	414
19	801	943	105	628
20	830	839	110	527
21	838	72-541	707	890
22	801	208-888	735	948
23	860	208-888	763	790
24	860	290-888	790	792
25	95	33-217	790	950
26	95	33-283	792	900
27	95	33-346	792	912
28	95	33-515	834	978
29	733	217-515	878	978
30	733	283-346	900	964
31	733	283-515	912	992
32	733	346-515	815	39-704
33	733	217-33-283	815	39-825
34	733	217-33-346	815	704-825
35	733	283-33-346	899	39-704
36	733	283-217-346	912	39-825
37	733	283-217-515	912	704-825
38	733	346-33-515	964	217-346
39	733	346-217-515	964	368-829
40	733	346-283-515	970	217-346
41			815	704-39-825
42			899	704-39-825
43			912	704-39-825

Item sets that are negatively associated when maximum regularity = 1200 and Minimum frequency = 650 at Location-1 mined considering first 40,000 records. Item sets that are negatively associated when maximum regularity = 600 and Minimum frequency = 400 at Location-2 mined considering records 40, 001 – 80,000 records. The details of the Item sets mined shown in Table 15. One can see from the Table that patterns happening at Location-1 are different from the patterns occurring at Location-2. There are a few patterns that happen at both places.

Data analysis is carried to find the behavior of regular, frequent itemsets and Negative frequent regular sets at Location—1 and Location-2 varying Maximum reality from 600 to 1200 at an increment of 200 varying frequency from 550 to 800 at an increment of 50. Figure 2 shows regular, frequent behavior, and Figure 3 show negative regular, frequent behavior when the maximum regularity and Minimum frequency is varied at Location-1.

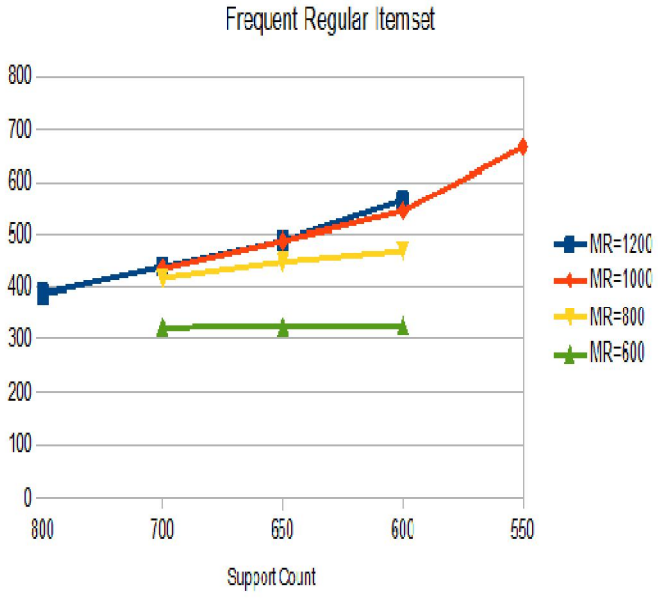

Figure 2: Regular, frequent itemsets at Location-1

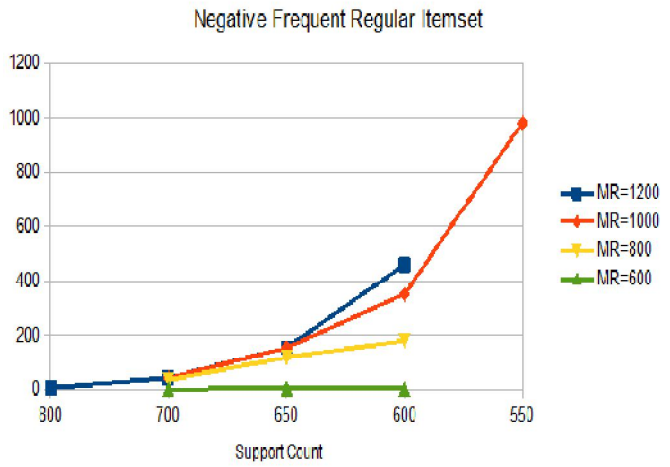

Figure 3: Negative Regular, frequent itemsets at Location-1

Figure 4 shows regular, frequent behavior, and Figure 5 show negative regular, frequent behavior when the maximum regularity and Minimum frequency is varied at Location-2.

One can see from figures 4 and Figures 5 that regular, frequent itemsets do not stabilize while negative regular item sets converge when the Maximum regularity fixed at 600 at Location-2

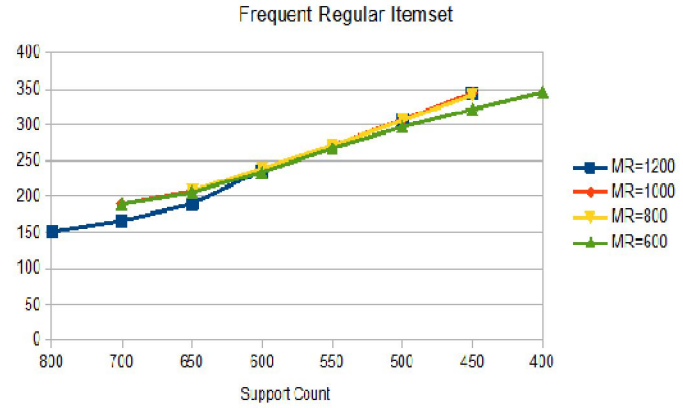

Figure 4: Regular, frequent itemsets at Location-2

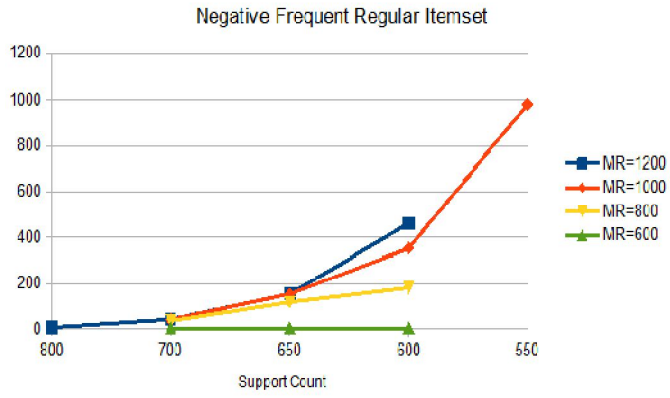

Figure 5: Negative Regular, frequent itemsets at Location-2

Figure 6 show regular, frequent behavior and Figure 7 showed negative regular, frequent behavior when the maximum regularity and Minimum frequency expressed in percentage with MR varied at increment 0.5% and Support % varied 0.13% is at Location-1.

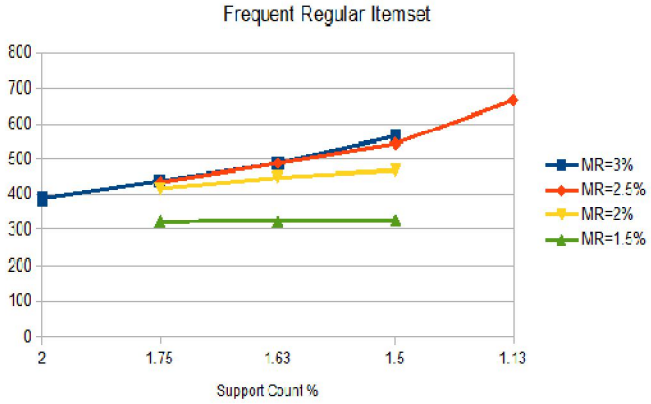

Figure 6: Behavior of regular, frequent itemsets when expressed in % at Location-1

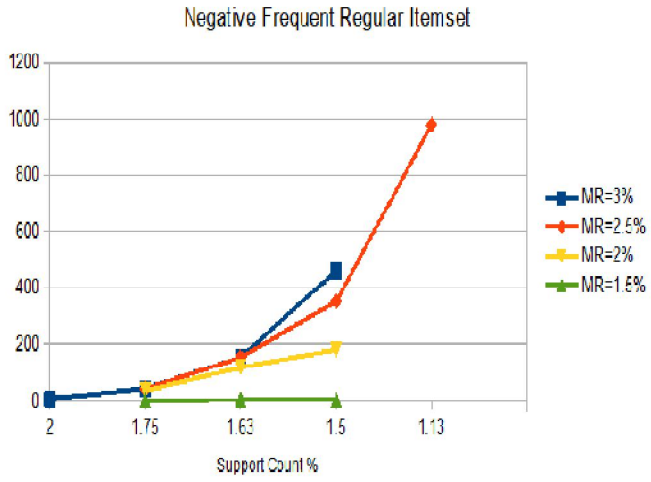

Figure 7: Behaviour of negative regular, frequent itemsets when expressed in % at Location-1

One can see from figures 6 and 7 that both the regular, frequent itemsets and Negative, regular, frequent itemsets converge with MR being 1.5%

8 show regular, frequent behavior and Figure 9 showed negative regular, frequent behavior when the maximum regularity and Minimum frequency expressed in percentage with MR varied at increment 0.5% and Support % varied 0.13% is at Location-2.

One can see from figures 8 and 9 that the both the regular, frequent itemsets and Negative, regular, frequent itemsets converges with MR being 1.5%.

7. CONCLUSION

In many large scale databases, knowledge mining process of finding frequent or regular patterns becomes tedious as the size of the transaction database increases. All the now a day, applications are generating transactions in terabytes and even petabytes. The traditional approach of multiple scans of the database and executing on a single data source makes the task to the highest degree of complexity. Databases for web applications, online eCommerce sites, etc. have been implementing distributed databases in place of the traditional centralized system to handle the hectic traffic. It is wise to implement mining algorithms also in parallel. The performance of parallel processing depends upon the amount of information communicated between the nodes. The process leads to a drastic increase in I/O operations that reduce performance.

Memory distribution methods also suffer from the inter-processor communication as all the processors share the common memory that leads to bottleneck problem. Hence the proposed algorithm successfully finds all the regular negative patterns by finding all possible positive patterns with a single scan of transactions database using vertical format. The algorithm is executed in parallel at each node on the portion of the database to generate regular items for further process of finding overall positive and negative patterns. There is no need to scan the database again for the final process, and hence the I/O operations minimized. Further, this aspect extended with hashing and indexing techniques on the database and additional parameters to strengthen the framework.

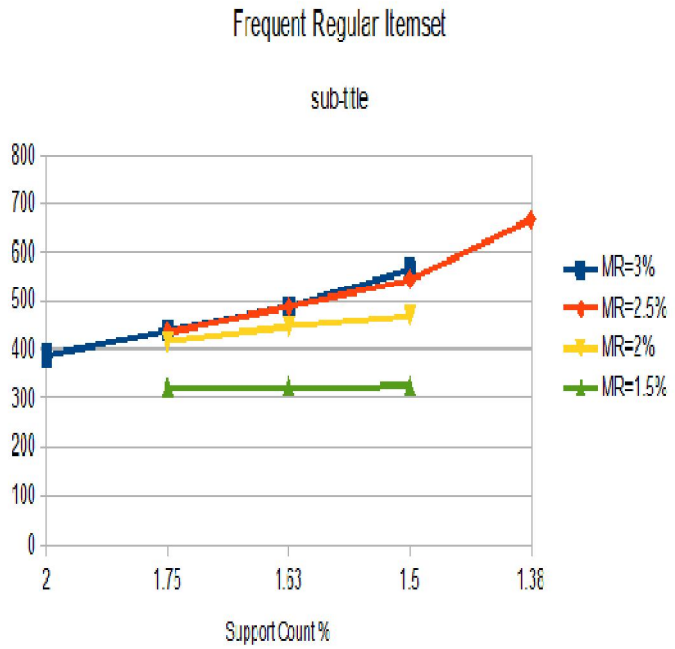

Figure 8: Behaviour of regular, frequent itemsets when expressed in % at Location-2

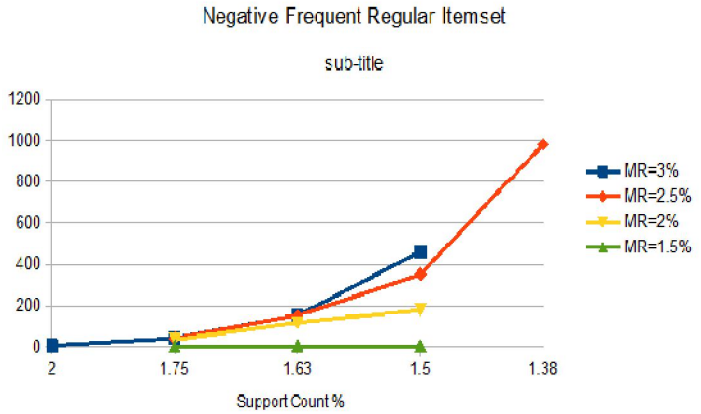

Figure 9: Behaviour of negative regular, frequent itemsets when expressed in % at Location-2

REFERENCES

1. Agrawal R., Imielinski., Swami N, " Mining association rules between sets of items in large databases," In ACM SIGMOD Int.Conf.on Management of Data, pp. 207-216 (1993) https://doi.org/10.1145/170036.170072
2. Agrawal R and Srikanth R." Fast algorithms for mining association rules", In VLDB, pp.489-499 (1994)
3. Savasere A., Omiecinski E., Navathe S.B." An efficient algorithm for mining association rules in large databases" In VLDB pp. 432-444 (1995)
4. J. S. Park, M. Chen and P. S. Yu. "An effective hash-based algorithm for mining association rules" In ACM SIGMOD Intl. Conf. Management of Data, May 1995. https://doi.org/10.1145/223784.223813
5. Mueller. "Fast sequential and parallel algorithms for association rule mining: A comparison." Technical report-TR-3515, University of Maryland, College Park, August 1995.
6. S. Brin, R. Motwani, J. Ullman, and S. Tsur. "Dynamic itemset counting and implication rules for market basket data" In ACM SIGMOD Conf. Management of Data, May 1997. https://doi.org/10.1145/253260.253325
7. R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, Inkeri Verkamo. " Fast discovery of association rules," In U. Fayyad and et al., editors, Advances in Knowledge Discovery and Data Mining, pages 307–328. AAAI Press, Menlo Park, CA, 1997.
8. M. J. Zaki, S. Parthasarathy, M. Ogihara, and W. Li. "New algorithms for fast discovery of association rules" In3rd Intl. Conf. on Knowledge Discovery and Data Mining, August 1997. https://doi.org/10.1007/978-1-4615-5669-5_1
9. F. Bodon, "A Fast Apriori Implementation," In B.Goethals and M. J. Zaki, editors, *Proceedings of the IEEE ICDM Workshop on Frequent Itemset Mining Implementations*, Vol. 90 of CEUR Workshop Proceedings, 2004.

10. Shengnan Cong Jiawei Han Jay Hoe_inger David Padua "A Sampling based Framework for Parallel Data Mining," June 15.17, 2005, Chicago, Illinois, USA. ACM 2005
11. Yanbin Ye, Chia-Chu Chiang, "A Parallel Apriori Algorithm for Frequent Itemsets Mining," in Proceedings of the Fourth International Conference on Software Engineering Research, Management and Applications (SERA'06),2006 https://doi.org/10.1109/SERA.2006.6
12. Tanveer S K., Farhan C.A., Jeong B-S. "Parallel and distributed algorithms for frequent pattern mining in large databases" In IETE technical review, vol.26, pp. 55-66 (2009) https://doi.org/10.4103/0256-4602.48469
13. Hu J. and Yang-Li X. "A fast parallel association rules mining algorithm based on FP-Forest" In 5th international symposium on Neural Networks, pp. 40-49 (2008) https://doi.org/10.1007/978-3-540-87734-9_5
14. GUO Yi-ming WANG Zhi-jun "A vertical format algorithm for frequent mining itemsets," IEEE, 2010 https://doi.org/10.1109/ICACC.2010.5487072
15. G.Vijay Kumar, M.Sreedevi, NVS.Pavan Kumar, "Mining Regular Patterns in Transactional Databases using Vertical Format, International Journal of Advanced Research in Computer Science," 2 (5), Sept-Oct, 2011.
16. G.Vijay Kumar, V.Valli Kumari, "Parallel and Distributed Frequent Regular Pattern Mining Using Vertical format in Large Databases," Fourth International Conference on Advances in Recent Technologies in Communication and Computing (ARTCom2012)
17. Philippe Fournier-Viger, X. Luo, J.X. Yu, and Z. Li (Eds.)" FHN: Efficient Mining of High-Utility Itemsets with Negative Unit Profits" ADMA 2014, Springer International Publishing Switzerland 2014 LNAI 8933, pp. 16–29, 2014. https://doi.org/10.1007/978-3-319-14717-8_2
18. Luca Cagliero and Paolo Garza, Infrequent Weighted Itemset Mining Using Frequent Pattern Growth, IEEE Transactions On Knowledge And Data Engineering, Vol. 26, No. 4, April 2014 https://doi.org/10.1109/TKDE.2013.69
19. Zengyou He, Feiyang Gu, Can Zhao, Xiaoqing Liu, Jun Wu, Ju Wang "Conditional Discriminative Pattern Mining: Concepts and Algorithms," Information Sciences (2016)
20. Snehal J. Patil1, A. S. Dange2 "Infrequent Weighted Itemset Mining Using Frequent Pattern Growth" International Journal of Engineering Science and Computing, August 2016
21. Philippe Fournier-Viger Jerry Chun-Wei Lin, Rage Uday Kiran, Yun Sing Koh, Rincy Thomas," A Survey of Sequential Pattern Mining," Data Science and Pattern Recognition Ubiquitous International, Volume 1, Number 1, February 2017.
22. Youxi Wu, Yao Tong, Xingquan Zhu, Senior Member, IEEE, and Xindong Wu, Fellow, IEEE, "NOSEP: Nonoverlapping Sequence Pattern Mining With Gap Constraints," 2017 IEEE
23. Changala, R., Rajeswara Rao, D., Evaluation and analysis of discovered patterns using pattern classification methods in text mining, ARPN Journal of Engineering and Applied Sciences, 13(11), pp. 3706-3717,2018
24. Kolli, S., Sreedevi, M., Prototype analysis of different data mining classification and clustering approaches, ARPN Journal of Engineering and Applied Sciences,13(9), pp. 3129-3135,2018
25. Deshpande, L., Rao, M.N., Concept drift identification using classifier ensemble approach, International Journal of Electrical and Computer Engineering,8(1), pp. 19-25,2018, https://doi.org/ 10.11591/ ijece.v8i1.pp19-25
26. Changala, R., Rajeswara Rao, D. T., A survey on the development of pattern evolving model for discovery of patterns in text mining using data mining techniques, Journal of Theoretical and Applied Information Technology,95(16), pp. 3974-3981,2017
27. Wagner, S.S., Rajarajeswari, P.., Parallel frequent dataset mining and feature subset selection for high dimensional data on Hadoop using map-reduce, International Journal of Applied Engineering Research, 12(18), pp. 7783-7789,2017
28. Vijay Kumar, G., Krishna Chaitanya, T., Pratap, M., Mining popular patterns from the multidimensional database, Indian Journal of Science and Technology, 9(17),93106 https://doi.org/10.17485/ijst/2016/v9i17/93106
29. Gangadhar, M.N.S., Sreedevi, M., Regular pattern mining on dynamic databases using vertical formate on given user regularity threshold, Journal of Theoretical and Applied Information Technology, 86(3), pp. 360-364, 2016
30. Greeshma, L., Pradeepini, G., Input split frequent pattern tree using MapReduce paradigm in Hadoop, Journal of Theoretical and Applied Information Technology, 84(2), pp. 260-271
31. Greeshma, L., Pradeepini, G., Mining Maximal Efficient Closed Itemsets Without Any Redundancy, Advances in Intelligent Systems and Computing, 433, pp. 339-347, https://doi.org /10.1007/978-81-322-2755-7_36
32. Kyeongjoo Kim, Jihyun Song and Minsoo Lee, Real-time Streaming Data Analysis using Spark, International Journal of Emerging Trends in Engineering Research, Volume 6, No.1, 2018, pp. 1-5 https://doi.org/10.30534/ijeter/2018/01612018
33. Sasi Bhanu J, Sastry JKR, Sunitha Devi B, Chandra Prakash, Career Guidance through TIC-TAC-TOE Game, International Journal of Emerging Trends in Engineering Research, Volume 7, No.6, 2019, pp. 25-31 https://doi.org/10.30534/ijeter/2019/01762019

Table 1: Comparative Analysis of Pattern finding Algorithms

Algorithm Serial Number	Main Author	Interestingness measures					Occurrence Behaviour					Type of Associations		Mining technique used
		Support	Confidence	Correlation	Multi support	Multi Correlation	Regularity	Irregularity/ Rare	Frequent	Maximal	Natural	Positive Associations	Negative Associations	
1	Agarwal R										√			
2	Savasere	√							√					
3	Park, 1995										√			Apriori + hashing
4	Muller								√		√			Apriori + Sequence Tree
5	Brin	√							√					Apriori + DIC
6	Zaki										√			maxClique, shared memory systems
7	Bandon	√							√					Local + Global item sets
8	Shengnan Cong	√							√					FP tree
9	Yenbin Ye								√					Apriori
10	Tanbeer	√						√			√			
11	GUO Yi-mig	√							√					
12	Vijay kumar	√							√	√		√		
13	Philippe Fournier-Viger											√	√	
14	Luca Cagliero			√								√		Raw data correlations
15	Zengyon He				√						√			
16	Shehal J Patil	√			√						√			
17	Yonxi Wu	√					√				√			
15	Pavan NVS	√					√	√	√	√	√	√	√	Veridical Tab