Mining Data Using Various Association Rule Mining Algorithms in Distributed Environment Using MPI

[1]Riddhi N. Bhoraniya, [2]Arjun V. Bala
[1]PG Student, [2]Assistant Professor
[1] Department of computer engineering,
[2]Darshan institute of Engineering and Technology, Rajkot,Gujarat, India.

***Abstract* - Data mining combines machine learning, statistics and visualization techniques to discover and extract knowledge. In order to improve the efficiency of mining algorithm for the large data sets we are implementing Distributed Data Mining (DDM). In distributed association rule mining algorithm, one of the major challenges is to reduce the communication overhead. Data sites are required to exchange lot of information in the data mining process which may generates massive communication overhead. Message passing interface (MPI) is a technique to exchange information among a number of communicating nodes. Here we apply association rule mining algorithms like TopKRules and TNR algorithm in distributed environment using MPI for mining data within less communication overhead.**

***Index Terms* - Data Mining, Distributed Data Mining, Association rule mining, Message Passing Interface (MPI).**

I. INTRODUCTION

Data mining is a powerful new technology that can help us to focus on the most important information available in our data warehouses. Among different approaches, association rule mining is one of the popular techniques for mining data. In this technique, an interrelation among different items in data is discovered by determining frequent large itemsets which are repeated more than a threshold number of times in the database.

Data mining process can be characterized as centralized and scattered based on the location of data. In case of centralized data mining process, data is resided into a single site whereas in scattered process data is resided into multiple sites. The data may be owned by each site separately or an enormous amount of data may be distributed into multiple data sites [1].

In order to perform many calculations simultaneously on different system Distributed computing is used. The basic principle of Distributed Computing is that a big task is divided into smaller subtasks and then every subtask can be computed in parallel on different machines. We can use the concept of distributed computing in data mining to increase the performance of the mining algorithm because in distributed computing we can make use of resources of all the computers at the same time. We can use Message Passing Interface (MPI). MPI is a message passing interface that can be useful for performing distributed processing. MPI allow the execution of programs written in C, C++, FORTRAN77 and FORTRAN90 in distributed fashion by calling the appropriate library routines. In order to achieve high-performance distributed computing, there is a need to design an algorithm that communicate by MPI between the hosts, make full use of the resources of the workstations [2].

II. OBJECTIVE

Creating a framework which can support association rule mining algorithms like TopKRules and TNR in Distributed Environment using Message Passing Interface (MPI).

This work proposes an association rule mining algorithm which minimizes the communication overhead between the participating data sites.

III. BACKGROUND

A. Association Rule Mining

Association Rule problem was first of all stated by R. Agrawal , that the conventional statement of association rule mining problem was discovering the interesting association or correlation relationships among a large set of data items [3].

Association rule mining finds the interesting or correlation relationship among a large set of data items. The typical example of association rule mining is the market basket analysis. Frequent itemset mining leads to the discovery of associations and correlation among items in large transactional or relational data sets. In brief, an association rule is an expression A $\Rightarrow$B, where A and B are sets of large items. Generally association rule mining is done in two steps.

Finding all frequent itemsets: The items which are frequently or habitually purchased together in one transaction are called the frequent itemsets.

Generate strong association rules from the frequent itemsets: the rules which are generated must satisfy both minimum support and minimum confidence.

There are two central basic measures for association rules, *support*(*s*) and *confidence*(*c*). Since the database is large and users worry about only those frequently purchased items, usually thresholds of support and confidence are predefined by users to crash those rules that are not so interesting or useful. The two thresholds are called *minimal support* and *minimal confidence*

respectively, additional constraints of interesting rules also can be specified by the users. The two basic parameters of Association Rule Mining (ARM) are: support and confidence [4].

Let us take **I = *i1, i2, ….im*** as a set of items. Take D , the data relevant to task, as a set of transactions where in each transaction T is a set of items such that T⊆I. Here, each and every transaction has a unique identifier, TID. Let A be a set of items. A transaction T is said to contain A if and only if A⊆T. An association rule is an implication of the form A⟹B, where A⊂T, B⊆T and A∩B=∅. The rule A⟹B holds in the transaction set D with support **s**, where **s** is the percentage of transactions in D that contain A∪B. The rule A⟹B is said to have confidence **c** in the transaction set D, if there are c percentage of transactions in **D** that contains **A** and also contains **B**,

- Support: Support of a particular itemset is the percentage of the total transactions having that itemset.
  Support (A⟹B) = prob{A ∪ B}
- Confidence: Confidence is the probability of occurring (such as buying) items together.
  Confidence (A⟹B) = prob{B/A}

Many algorithms for generating association rules were presented in past time. A large number of association rule mining algorithms have been developed with different mining efficiencies. Any algorithm should find the same set of rules though their computational efficiencies and memory requirements may be different.

B. TopKRules Algorithm

TopKRules is an algorithm for discovering the **top-k association rules** appearing in a transaction database [5].

In other association rule mining algorithms requires to set a minimum support (*minsup*) parameter that is hard to set (usually users set it by trial and error, which is time consuming). **TopKRules** solves this problem by letting users directly indicate *k*, the number of rules to be discovered instead of using *minsup*.

TopKRules takes three parameters as input:

- a transaction database,
- a parameter *k* representing the number of association rules to be discovered (a positive integer),
- a parameter *minconf* representing the minimum confidence that the association rules should have (a value in [0,1] representing a percentage).

TopKRules outputs the **top-k association rules**.

TopKRules relies on a new approach called rule expansions and also includes several optimizations that improve its performance. Experimental results show that TopKRules has excellent performance and scalability, and that it is an advantageous alternative to classical association rule mining algorithms when the user wants to control the number of association rules generated.

C. TNR (Top-k Non-Redundant) Algorithm

TNR is an algorithm for discovering the **top-k non-redundant association rules** appearing in a transaction database. It is an approximate algorithm in the sense that it always generates non-redundant rules. But these rules may not always be the top-k non-redundant association rules. TNR uses a parameter named **delta**, which is a positive integer >=0 that can be used to improve the chance that the result is exact (the higher the delta value, the more chances that the result will be exact) [6].

TNR takes four parameters as input:

- a transaction database,
- a parameter *k* representing the number of rules to be discovered (a positive integer >= 1),
- a parameter *minconf* representing the minimum confidence that association rules should have (a value in [0,1] representing a percentage).
- a parameter *delta* (a positive integer >=0) that is used to increase the chances of having an exact result (because the TNR algorithm is approximate).

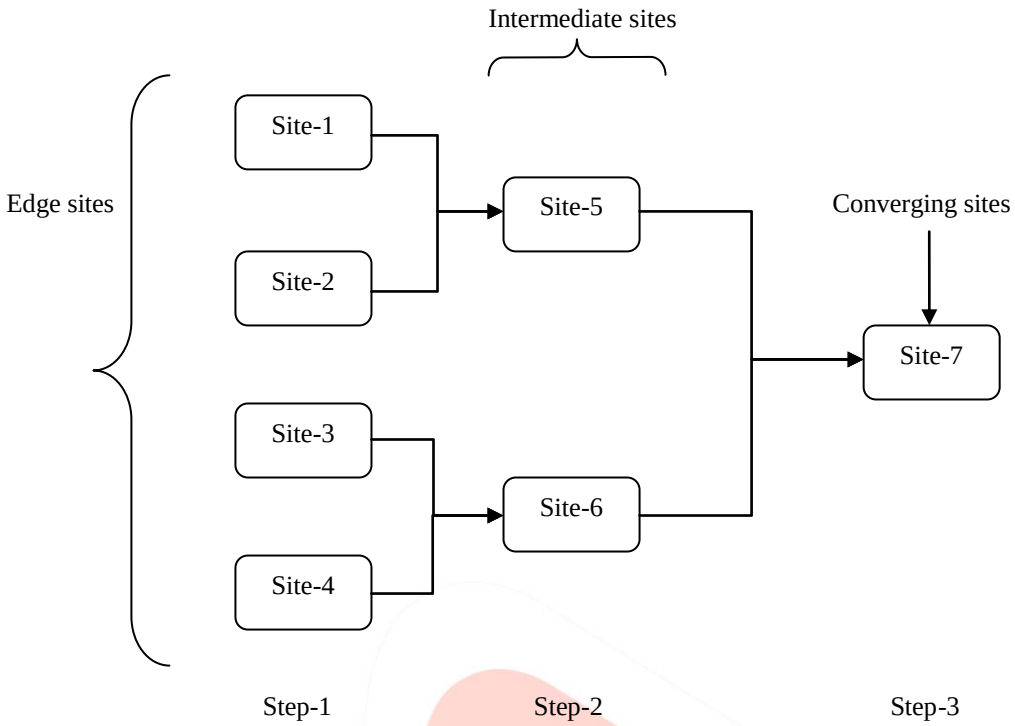
TNR outputs an approximation of the *k* most frequent **non redundant association rules** having a confidence higher or equal to *minconf*.

TNR is an efficient algorithm. It is based on the TopKRules algorithm for discovering **top-k association rule**. The main difference between TNR and TopKRules is that TNR includes additional strategies to eliminate redundancy in results, and that TNR is an approximate algorithm, while TopKRules is not.

D. Message Passing Interface (MPI)

Message Passing Interface (MPI): This is a technique to exchange information between a number of communicating nodes. It is especially suitable for mathematical functions like: summation or accumulation of a particular number which is to be calculated and scattered among nodes. This allows nodes to swap the information without spreading; therefore, it reduces the communication overhead and communication round considerably. Detail of MPI can be found in R. Agrawal and J. Shafer (1996) and Argonne National Laboratory (MPI) [7].

Let us consider 8 sites 1, 2,….7 have their own count values to be summed and shared between themselves. One simple solution is to broadcast everyone's count to others in the general medium. This solution would require massive data transmission over the medium. Also, broadcasting is avoided due to many other reasons if alternatives are realistic. MPI in this case provide best solution.



Step-1: site-1 and site-2 transmit their counts to site-5 and site-3 and site-4 transmit their counts to site-6 respectively.
Step-2: site-5 and site-6 transmit their counts to site-7.
Step-3: finally site-7 has all the counts of all sites. Now site-7 is able of calculating the sum of counts.

Participating sites in the MPI technique can be divided into three categories: edge sites (in this example 1, 2, 3 and 4) are those which do not receive from others, intermediate sites (in this example 5 and 6) are those sites which receives from other sites and converging site (in this example 7) receives all information from others and accumulates.

IV. PROPOSED SYSTEM ALGORITHM

Input: number of transactions
Output: top-k association rule for all sites

Begin: At Master site

1. Broadcast RunTopKRule command
2. While (Received message is not end)
3. Wait for bit-vector from all the site
4. Generate cumulative bit-vector
5. Broadcast cumulative bit-vector
6. End while

End

Begin: at all sites

1. Wait for RunTopKrule command
2. Calculate m using m = t/(c+1),
   where t=total transactions and c = number of item
3. Run TopKRule algorithm
4. Generate bit-vector
5. Send bit-vector to master site using MPI
6. Wait for master's cumulative vector.
7. If $t \leq m*i$
8. Goto step 3
9. Else
10. Send End message to master

End

Description of algorithm

There would be different algorithms at master site and at rest of the local sites. At all the local sites, there are two options form. either a static m would be provided, or m is calculated using the formula m=t/(c+1). Here't' represents the total number of transactions, and c represents total number of items.

Once the sites receive RunTopKRule command from the master, they start running the TopKRule algorithm locally. Thereafter a Bit-vector would be generated. This Bit-vector is sent back to master. For this purpose, Message Passing Interface is used.

At master site, initially, RunTopKRule command will be broadcasted to all the local sites in the network. Then master check for message end from all site and if any message is not received then execute next step where master site will wait for receiving the Bit-vector from the local sites, and will generate Cumulative Bit-Vector which will again be broadcasted to all the sites.

The TopkRule algorithm runs in a loop at all local sites, and every time Bit-Vector and Cumulative Bit-Vectors will be generated at local sites, and master site respectively. Cumulative bit-vector is generate with binary OR operation between available bit-vectors.

At the end of every iteration, a condition is required to be checked, i.e. $t \leq m*i$.

As far as this condition is found to be true, the loop will continue, but once it becomes false, the algorithm will check if all item-set is completed, and will terminate.

V. CONCLUSION

Data mining predict future trends and behaviors, which helps to make proactive, knowledge-driven decisions. Association rule mining is an active data mining research area. *Distributed data mining* has therefore emerged as an active subarea of data mining research most ARM algorithms focus on a sequential or centralized environment where no exterior communication is required. *Distributed ARM* algorithms, on the other hand, aim to generate rules from different data sets spread over different geographical sites; hence, they need external communications throughout the entire process. DARM algorithms must reduce communication expenses so that generating global association rules costs less than combining the participating sites' data sets into a centralized site.

Apply TopKRules and TNR association rule mining algorithms in distributed environment using MPI we can get number of association rules or non-redundant rules with minimum communication overhead.

VI. ACKNOWLEDGMENT

Special thanks to Darshan Institute of Engineering and Technology for supporting in this research. I am grateful to Prof. Arjun V. Bala for helpful discussion and comments.

REFERENCES

[1] Md. Golam Kaosar, Zhuojia Xu and Xun Yi :"Distributed Association Rule Mining with Minimum Communication Overhead".
[2] Hitesh Ninama : "distributed data mining using message passing interface".
[3] R.Agrawal, T.Imielinski, and A.Swami, 1993. "Mining association rules between sets of items in large databases", in proceedings of the ACM SIGMOD Int'l Conf. on Management of data, pp. 207-216.
[4] Qiankun Zhao and Sourav S. Bhowmick :"Association Rule Mining: A Survey"
[5] Philippe Fournier-Viger and Vincent S. Tseng :"Mining Top-K Association Rules".
[6] Philippe Fournier-Viger and Vincent S. Tseng :"Mining Top-K Non-Redundant Association Rules".
[7] R. Agrawal and J.C. Shafer (1996) :"Parallel Mining of Association Rules", Knowledge and Data Engineering, IEEE Transactions on Volume 8, Issue 6, pp. 962-969.
[8] Byung-Hoon Park and Hillol Karguota: "Distributed Data Mining: Algorithms ,Systems and Applications".
[9] Mafruz Zaman Ashrafi, David Taniar, Kate Smith: " An Optimized Distributed Association Rule Mining Algorithm" IEEE distributed systems online vol. 5, no. 3,pp1-18,march 2004
[10] R. Agrawal and J. Shafer (1996): "Parallel Mining of Association Rules: Design, Implementation and Experience", Research Report RJ 10004, IBM Almaden Research Center, San Jose, Calif.
[11] Surbhi Bhatnagar,"Algorithm for finding association rules in distributed databases",IEEE,2012
[12] Wenliang Cao,"Research of the mining algorithm based on distributed database",IEEE,2011