Association Rule Mining Algorithm In Parallel And Distributed Data Mining

Rashmi Jha
Research Scholor, B.R.A Bihar University
Under Guidance Of

Dr. Amit Kumar

HEAD, DEPARTMENT OF MATHEMATICS

RPS COLLEGE, CHEKIYAJ (VAISHALI), B.R.A BIHAR UNIVERSITY

Paper Received on: 30/06/2014
Paper Reviewed on: 01/07/2014
Paper Accepted on: 03/07/2014

Abstract

Many current data mining tasks can be accomplished successfully only in a distributed setting. The field of distributed data mining has therefore gained increasing importance in the last decade. The Apriori algorithm has emerged as one of the best Association Rule mining algorithms. Ii also serves as the base algorithm for most parallel algorithms. The enormity and high dimensionality of datasets typically available as input to problem of association rule discovery, makes it an ideal problem for solving on multiple processors inparallel. The primary reasons are the memory and CPU speed limitations faced by single processors. In this paper an Optimized Distributed Association Rule mining algorithm for geographically distributed data is used in parallel and distributed environment so that it reduces communication costs

Keywords*: Data Mining, Association, Knowledge Discovery, Association rules, Apriori algorithm, parallel and distributed data mining*

INTRODUCTION

Association rule mining (ARM) has become one of the core data mining tasks and has attracted tremendous interest among data mining researchers. ARM is an undirected or unsupervised data mining technique which works on variable length data, and produces clear and understandable results. There are two dominant approaches for utilizing multiple processors that have emerged distributed memory in which each processor has a private memory; and shared memory in which all processors access common memory. Shared memory architecture has many desirable properties. Each processor has direct and equal access to all memory in the system. Parallel programs are easy to implement on such a system. In distributed memory architecture each processor has its own local memory that can only be accessed directly by that processor. For a processor to have access to data in the local memory of another processor a copy of the desired data element must be sent from one processor to the other through

message passing. XML data are used with the Optimized Distributed Association Rule Mining Algorithm.

A parallel application could be divided into number of tasks and executed concurrently on different processors in the system [9]. However the performance of a parallel application on a distributed system is mainly dependent on the allocation of the tasks comprising the application onto the available processors in the system.

Modern organizations are geographically distributed. Typically, each site locally stores its ever increasing amount of day-to-day data. Using centralized data mining to discover useful patterns in such organizations' data isn't always feasible because merging data sets from different sites into a centralized site incurs huge network communication costs. Data from these organizations are not only distributed over various locations but also vertically fragmented, making it difficult if not impossible to combine them in a central location. Distributed data mining has thus emerged as an active subarea of data mining research. In this paper an Optimized Association Rule Mining Algorithm is used for performing the mining process.

1. ASSOCIATION RULE MINING ALGORITHMS

An association rule is a rule which implies certain association relationships among a set of objects (such as ``occur together'' or ``one implies the other'') in a database. Given a set of transactions, where each transaction is a set of literals (called items), an association rule is an expression of the form X Y , where X and Y are sets of items. The intuitive meaning of such a rule is that transactions of the database which contain X tend to contain Y.

2.1 Apriori Algorithm

An association rule mining algorithm, Apriori has been developed for rule mining in large transaction databases by IBM's Quest project team [3]. An itemset is a non-empty set of items. They have decomposed the problem of mining association rules into two parts

A) Find all combinations of items that have transaction support above minimum support. Call those combinations frequent item sets.
B) Use the frequent item sets to generate the desired rules. The general idea is that if, say, ABCD and AB are frequent item sets, then we can determine
if the rule AB CD holds by computing the ratio r = support(ABCD)/support(AB). The rule holds only if r >= minimum confidence. Note that the rule will have minimum support because ABCD is frequent. The algorithm is highly scalable [7].The Apriori algorithm used in Quest for finding all frequent itemsets is given below.

procedure Apriori Alg()

```
Begin
L1 := {frequent 1-itemsets};
for ( k := 2; Lk-1 0; k++ ) do {
Ck= apriori-gen(Lk-1) ; // new candidates
for all transactions t in the dataset do {
for all candidates c Ck contained in t do
c:count++
}
Lk = { c Ck | c:count >= min-support}
}
Answer := k Lk
        End
```

It makes multiple passes over the database. In the first pass, the algorithm simply counts item occurrences to determine the frequent 1-itemsets (item sets with 1 item). A subsequent pass, say pass k, consists of two phases. First, the frequent item sets Lk-1 (the set of all frequent (k-1)-item sets) found in the (k-1)th pass are used to generate the candidate item sets Ck, using the apriori-gen() function. This function first joins Lk-1 with Lk-1, the joining condition being that the lexicographically ordered first k-2 items are the same. Next, it deletes all those item sets from the join result that have some (k-1)-subset that is not in Lk-1 yielding Ck. The algorithm now scans the database. For each transaction, it determines which of the candidates in Ck are contained in the transaction using a hash-tree data structure and increments the count of those candidates [8], [14]. At the end of the pass, Ck is examined to determine which of the candidates are frequent, yielding Lk. The algorithm terminates when Lk becomes empty

2.2 Distributed/parallel algorithms

Databases or data warehouses may store a huge amount of data to be mined. Mining association rules in such databases may require substantial processing power [6]. A possible solution to this problem can be a distributed system.[5] . Moreover, many large databases are distributed in nature which may make it more feasible to use distributed algorithms. Major cost of mining association rules is the computation of the set of large itemsets in the database. Distributed computing of large itemsets encounters some new

problems. One may compute locally large itemsets easily, but a locally large itemset may not be globally large. Since it is very expensive to broadcast the whole

data set to other sites, one option is to broadcast all the counts of all the itemsets, no matter locally large or small, to other sites. However, a database may contain enormous combinations of itemsets, and it will involve passing a huge number of messages. A distributed data mining algorithm FDM (Fast Distributed Mining of association rules) has been proposed by [5], which has the following distinct features.

A) The generation of candidate sets is in the same spirit of Apriori. However, some relationships between locally large sets and globally large ones are explored to generate a smaller set of candidate sets at each iteration and thus reduce the number of messages to be passed.
B) After the candidate sets have been generated, two pruning techniques, local pruning and global pruning, are developed to prune away some candidate sets at each individual sites.
C) In order to determine whether a candidate set is large, this algorithm requires only O(n) messages for support count exchange, where n is the number of sites in the network. This is much less than a straight adaptation of Apriori, which requires O(n2) messages.

Distributed data mining refers to the mining of distributed data sets. The data sets are stored in local databases hosted by local computers which are connected through a computer network [15], [16]. Data mining takes place at a local level and at a global level where local data mining results are combined to gain global findings. Distributed data mining is often mentioned with parallel data mining in literature. While both attempt to improve the performance of traditional data mining systems they assume different system architectures and take different approaches. In distributed data mining computers are distributed and communicate through message passing. In parallel data mining a parallel computer is assumed with processors sharing memory and or disk. Computers in a distributed data mining system may be viewed as processors sharing nothing. This difference in architecture greatly influences algorithm design, cost model, and performance measure in distributed and parallel data mining.

2.3 Distributed Algorithms

- Distributed association rule learning
- Collective decision tree learning
- Collective PCA and PCA-based clustering
- Distributed hierarchical clustering
- Other distributed clustering algorithms
- Collective Bayesian network learning
- Collective multi-variate regression

2.4 Parallel Algorithms

The main challenges associated with parallel data mining include

- minimizing I/O
- minimizing synchronization and communication
- effective load balancing [12]
- effective data layout
- deciding on the best search procedure to use
- good data decomposition

2. OPTIMIZED DISTRIBUTED ASSOCIATION RULE MINING ALGORITHM

The performance of Apriori ARM algorithms degrades for various reasons. It requires n number of database scans to generate a frequent n-itemset. Furthermore, it doesn't recognize transactions in the data set with identical itemsets if that data set is not loaded into the main memory. Therefore, it unnecessarily occupies resources for repeatedly generating itemsets from such identical transactions. For example, if a data set has 10 identical transactions, the Apriori algorithm not only enumerates the same candidate itemsets 10 times but also updates the support counts for those candidate itemsets 10 times for each iteration. Moreover, directly loading a raw data set into the main memory won't find a significant number of identical transactions because each transaction of a raw data set contains both frequent and infrequent items. To overcome these problems, we don't generate candidate support counts from the raw data set after the first pass. This technique not only reduces the average transaction length but also reduces the – minimizing/avoiding duplication of work data set size significantly, so we can accumulate more transactions in the main memory. The number of items in the data set might be large, but only a few will satisfy the support threshold..Consider the sample data set in Figure 1a. If we load the data set into the main memory, then we only find one identical transaction (ABCD), as Figure 1b shows. However, if we load the data set into the main memory after eliminating infrequent items from every transaction (that is, items that don't have 50 percent of support and thus don't occur once in every second transaction in this case, itemset E), we find more identical transactions (see Figure 1c). This technique not only reduces average transaction size but also finds more identical transactions. The following gives the pseudo code of ODAM algorithm.

ODAM eliminates all globally infrequent 1-itemsets from every transaction and inserts them into the main memory; it reduces the transaction size (the number of items) and finds more identical transactions. This is because the data set initially contains both frequent and infrequent items. However, total transactions could exceed the main memory limit. ODAM removes infrequent items and inserts each transaction into the main memory. While inserting the transactions, it checks whether they are already in memory. If yes, it increases that transaction's counter by one. Otherwise, it inserts that transaction into the main memory with a count equal to one. Finally, it writes all main-memory entries for this partition into a temp file. This process continues for all other partitions

Transactions	
No.	**Items**
1.	ABCD
2.	BC
3.	AB
4.	ABCDE
5.	ACD
6.	ABE
7.	CDE
8.	AD
9.	BCE
10.	ABCD

Figure 1.(a) An Example Dataset

Transactions	
No.	**Items**
1,10	ABCD
2.	BC
3.	AB
4.	ABCDE
5.	ACD
6.	ABE
7.	CDE
8.	AD
9.	BCE

Figure 1. (b) Identical transactions

Transactions	
No.	**Items**
1,4,10	ABCD
2.9	BC
3,6	AB
5	ACD
7	CDE
8	AD

Figure 1. (c) Transactions after Pruning frequent Items.

4. PERFORMANCE EVALUATION

The number of messages that ODAM exchanges among various sites to generate the globally frequent itemsets in a distributed environment, we partition the original data set into five partitions.

To reduce the dependency among different partitions, each one contains only 20 percent of the original data set's transactions. So, the number of identical transactions among different partitions is very low. ODAM provides an efficient method for generating association rules from different datasets, distributed among various sites.

5. CONCLUSION

In this paper an idea is given to the potential implications of successful forecasting in a business context. There are two major types of predictions: one can either try to predict some unavailable data values or pending trends, or predict a class level for some data. The latter is tried to classification. Once a classification model is built based on a training set, the class label of an object can be foreseen based on the attribute values of the object and the attribute value of the class. Prediction is however more often referred to the forecast of missing numerical values, or increase/decrease trends in time related data. The major idea is to use a large number of past values to consider probable future values.

[1] R. Agrawal, T. Imielinski, and A. Swami, "Mining Association Rules between Sets of Items in Large Databases", In Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, Pages 207–216, Washington, DC, May 26- 28 1993.
[2] Agrawal, R., Imielinski, T., & Swami, A. (1993), "Mining Association Rules between Sets of Items in Large Databases", Proceedings of the ACM SICMOD Conference on Management of Data, pp.207-216, Washington, D.C.
[3] Han, J., Pei, J., & Yin, Y (2000), "Mining Frequent Patterns Candidate Generation" In Proc. 2000 ACM SIGMOD Int.Management of Data (SIGMOD'00), Dallas, TX. [4] Berzal, F., Blanco, I., Sánchez, D. and Vila, M.A."Measuring the Accuracy and Importance of Association Rules: A New Framework" Intelligent Data Analysis, 6:221- 235, 2002.
[5] David A. Clausi, "An Analysis of Co-occurrenc Texture Statistics as a Function of Gray Level Quantization", Can. J. Remote Sensing, 28, No. 1, pp. 45-62, 2002.
[6] Bodon, F. "A Fast Apriori Implementation", In Proc. IEEEICDM Workshop on Frequent Item set Mining Implementations, 2003.
[7] Brijs, T. Vanhoof, K. and Wets, G., "Defining Interestingness for Association Rules", In Int. Journal Of Information Theories and Applications,10:4, 2003.
[8] Tung, A., Lu, H., Han, J., & Feng, L. (2003), "Efficient Mining of Inter transaction Association Rules", IEEE Transaction on Knowledge and Data Engineering, 15(1),43-56.
[9] Xu, Z. & Zhang, S. (2003), "An Optimization Algorithm Base on Apriori for Association Rules",Computer Engineering, 29(19), 83-84.
[10] 4th European Conference of the International Federation for Medical and Biological Engineering ECIFMBE 2008 23–27 November 2008 Antwerp, Belgium, 10.1007/978-3-540-89208-3_144, Jos Vander Sloten, Pascal Verdonck, Marc Nyssen and Jens Haueisen.

Author's Profile

Rashmi Jha was born in Sitamarhi, Bihar, India. She has done M.Phil. in Computer Science from Manav Bharti University, H.P in 2012. Worked as a lecturer in BRBA University in Computer Science Dept.She is currently associated with NIELIT Centre (GCTI) here handling different type of Govt. projects under ministry of Information Technology. She is doing her research work in developing algorithm for data mining