Int. J. Advanced Networking and Applications
Volume: 02 Issue: 01 Pages: 458-463 (2010)

Performance Evaluation of Sequential and Parallel Mining of Association Rules using Apriori Algorithms

Puttegowda D
Department of Computer Science, Ghousia College Of Engineering, Ramanagarm
Email: puttegowda_d@yahoo.com
Rajesh Shukala
Department of Computer Science, Corporate Institute of Science and Technology, Bhopal
Email: rkumar_dmh@rediffmail.com
Deepak N A
Department of Computer Science, Ghousia College Of Engineering, Ramanagarm
Email: deepak_n_a@rediffmail.com

----------ABSTRACT----------
The information age has seen most of the activities generating huge volumes of data. The explosive growth of business, scientific and government databases sizes has far outpaced our ability to interpret and digest the stored data. This has created a need for new generation tools and techniques for automated and intelligent database analysis. These tools and techniques are the subjects of the rapidly emerging field of data mining. One of the important problems in data mining is discovering association rules from databases of transactions where each transaction consists of a set of items. The most time consuming operation in this discovery process is the computation of the frequency of the occurrences of interesting subset of items (called candidates) in the database of transactions. To prune the exponentially large space of candidates, most existing algorithms consider only those candidates that have a user defined minimum support. Even with the pruning, the task of finding all association rules requires a lot of computation power and memory. Parallel computers offer a potential solution to the computation requirement of this task, provided efficient and scalable parallel algorithms can be designed. In this paper, we have implemented Sequential and Parallel mining of Association Rules using Apriori algorithms and evaluated the performance of both algorithms.

Keywords:- Association Rules; Apriori algorithms; minimum support; computation power; performance
---
Date of Submission: March 27, 2010 Date of Acceptance: May 29, 2010

1. Introduction

Organizations, which have accumulated great volumes of data, are often interested in obtaining different types of information from it. Using Data Mining, they want to find out how best to satisfy their customers, how to allocate their resources efficiently, and how to minimize losses. One can draw several such meaningful conclusions by using Data Mining Techniques. However, the amount of data involved can often be immeasurable in size. For instance, a credit card company accumulates thousands of transactions every day. Over a period of a year, the company will have to deal with millions of records. In this paper, we are applying Association Rules on a library database, which is one kind of large transaction database. Since the computational complexity of the mining algorithms can vary, we attempt to improve their performance through parallel processing.

1.1 Data Mining

Data Mining (sometimes called data or knowledge discovery) is the process of analyzing data from different perspectives and summarizing it into useful information - information that can be used to increase revenue, cuts costs, or both. Data Mining software is one of a number of analytical tools for analyzing data. It allows users to analyze data from many different dimensions or angles, categorize it, and summarize the relationships identified. Technically, Data Mining is the process of finding correlations or patterns among dozens of fields in large relational databases. A Data Mining approach can generally be categorized into one of the following six-types: Classification, Regression, Time Series, Clustering, Association Rules and Sequence Discovery.

1.2 Association Rule

Data Mining is motivated by the decision support problem faced by most of the large retail organizations. Recently in [3, 5], introduced a class of regularities Association Rules and gave an Algorithm for finding such rules. An Association Rules is an expression X→Y, where X and Y are set of keywords. An intuitive meaning of such rule is that the transactions of the database, which contain X tends to contain Y. An example of such rule

might be that 98% of customers that purchase tires and automobile accessories also have automotive services carried out.

The following is a formal statement of the problem:

Let I={i_1, i_2,…….i_m} be a set of literal, called keyword set, and *d* be a set of transactions, where each transaction T is in keyword such that T⊆ I unique. A set of keyword such that X⊂I is a called keyword set. We say that a transaction T contains the set of keywords X, where X⊆T. An association is an implementation of the form X→Y, where it holds in the transaction set *d* with confidence *c*, if c% of transaction in *d* that contains X also contains Y. The rule X→Y has support *s* in the transaction set *d* containing X∪Y. Negative or missing keywords are not considered in this approach. Given a set of transaction *d*, the problem is to generate all Association Rules using user specified minimum Support and Confidence.

1.3 Motivation for Parallel Mining of Association Rules

With the availability of inexpensive storage and the progress in data capture technology, many organizations have created ultra large databases of business and scientific data, and this trend is expected to grow even more. A complementary technology trend is the progress in networking, memory and processor technologies that has opened up the possibility of accessing and manipulating this massive database in reasonable amount of time. The promise of Data Mining is that it delivers technology that will enable the development of new bread of decision support application.

The following factors will influence the parallel mining of Association Rules

- Very large data sets.
- Memory limitations of sequential computers cause sequential algorithm to make multiple expensive input/output passes over data.
- Need for scalable and efficient Data Mining computation
- Handle larger data for greater accuracy in a limited amount of time.
- Aspiration to gain competitive advantage.

In the recent past there has been considerable research in designing Data Mining algorithm. However, the work so far has been mainly concentrated on designing serial algorithm. Since databases to be mined are often very large (measured in gigabytes and even terabytes), parallel algorithms [2] would be required.

1.4 Parallel Mining of Rules

For parallel mining of Association Rules [2], we investigate and study the behavior implications of spectrum trade off between computation, synchronization and memory usage. Especially transaction databases partition data into a number of subgroups and attempts to utilize the main memory of the system efficiently. Each partition could be assigned to a processor after generating rules; each processor may independently make the decision to terminate or to process the next pass of the algorithm.

2. Sequential Mining of Association Rules

2.1 Problem Description

In this, we consider the problem of mining Association Rules [1] that satisfy user specified minimum Support and minimum Confidence level. Given a set of transactions, where each transaction involves a set of keywords, an association rule is an expression of the form X⇒Y, where X and Y are subsets of keywords. A rule X⇒Y holds in the transaction database d with confidence C if C% of transactions in *d* which contain X also contain Y. The rule has support S in the transactions set if S% of transactions in *d* contain X∪Y.

Confidence emphasizes the strength of implication and the Support level indicates the frequency of with which patterns occur in the database. It is often desirable to pay attention to only those rules, which have reasonably large Support. Such rules with high Confidence and strong Support are referred to as strong rules. The task of mining Association Rules is essentially to discover strong Association Rules in large databases. The problem of mining Association Rules has been decomposed into the following two steps.

1. Discover large keyword sets, i.e., the set of keyword sets that have Support above a predetermined minimum Support S.
2. Use these large keyword sets to generate the Association Rules for the database.

Note that the overall performance of mining is determined by the first phase. After large keyword sets are identified, the corresponding Association Rules can be derived in a straightforward manner.

2.2 Support

Let X be a subset of keywords in K, then the Support for X in *d* is the number of transactions d_i, which contain all the keywords appearing in X. K is a subset of the set {k_1,k_2…….,k_m}

$$\text{Support}(X) = \delta d_{i,x}$$

$$\text{Where } \delta d_{i,x} \begin{cases} 1, & \textit{if all keywords in X appear in entry d i} \\ 0, & \text{otherwise.} \end{cases}$$

2.3 Confidence of a Rule

Given rule in the form R_i : X⇒Y, where

X, Y $\subset$ K, X $\cap$ Y=$\varnothing$. We have the confidence level of the rule as

Confidence (R_i) = Support (X$\cup$ Y)/ Support (X)

We are often interested in rules R_i which satisfy the property. Confidence (R_i) $\geq$ C.

2.4 Association Rule

Given a set of records, each of which contains some number of keywords from a given collection, produce dependency rule, which will predict the occurrences of the keywords.

Eg:{Data Mining, Knowledge Discovery}$\Rightarrow$ {Expert systems}

Have 80% confidence and 10% supports.

This means that in a transaction database with n records. 0,1, .. n transactions contain the 3 keywords " Data mining " "Knowledge Discovery " and "Expert Systems". Further, there are 0.125n records, which contain the keywords "Data Mining" and "Knowledge Discovery ".

2.5 Discovering large keyword sets

Discovering large keyword sets make multiple passes over the data. In these passes, count the Support of individual keyword sets and determine which of them are large i.e., have minimum Support. In each subsequent pass, start with seed set of keyword sets found in the previous pass. Use this seed set for generating new and potentially large keyword sets called candidate keyword sets and count the actual support for these candidate keyword sets during the pass over the data. At the end of the pass, determine which of the candidate keyword sets are actually large and then they become the seed for the next pass. This process continues until no new large keyword sets are found.

We summarize in the Table 1 the notation used in the algorithms.

K– Keyword set	A keyword set of K keywords
L_k	Set of large K keyword sets (those with minimum support) Each member of this set has two fields 1) Keyword Set. 2) Support count.
C_k	Set of candidate K keyword sets (potentially large keyword set) each member of this set has two field 1) Keyword set 2) Support count.

Table 1 Notation for sequential algorithm

2.6 Apriori Algorithm

The basic "Apriori" Algorithm is summarized below:

L_1 ={frequent 1 keyword set};
K=2;
while (L_{k-1} is not empty)
{
C_k =Apriori_ Gen (L_{k-1});
for all transactions t in T
{
if support [subset (C_k, t)] $\geq$ minimum support
$L_k = L_k \cup C_k$;
}
}

Answer = $\cup_k L_k$

2.7 Procedure Apriori_Gen()

The Apriori_Gen() function takes as argument L_{k-1}, the set of all large (k-1) keyword sets, it returns the superset of the set of all large K keyword sets. The Ariori_Gen () performs two phases, Join and Prune operations.

1) Join operation

Apriori_Gen (L_{k-1})
{
Insert into C_k
select p.keyword$_1$……..p.keyword$_{k-1}$,…..q.keyword$_{k-1}$ from L_{k-1} of p, L_{k-1} of q
where p.keyword$_1$ = q.keyword$_1$, ……..
p.keyword$_{k-2}$ = q.keyword$_{k-2}$,
p.keyword$_{k-1}$ < q.keyword$_{k-1.}$
}

2) Prune operation

Delete all keyword sets C$\in$ C_k such that some (k-1)
Subset of C is not in L_{k-1},
for all keyword set C $\in$ C_k
{
for all (k-1) subset S of C
{
if (S $\notin$ L_{k-1}) then
Delete C from C_k
}
}

We need to show that $C_k \geq L_k$. Clearly any subset a large keyword set must also have minimum Support. Hence if we extend each keyword set in $L_{k-1,}$ we would be left with superset of the keyword sets in $L_{k.}$

The join operation is equivalent to extending L_{k-1} with each keyword set in the database and then deleting those keyword set for which L_{k-1} keyword set obtained by deleting the (k-1)th keyword set is not in $L_{k-1.}$ The condition p keyword$_{k-1}$ < q.keyword$_{k-1}$ simply ensures that no duplicates are generated. Thus after the join operation

$C_k \supseteq L_k$. By similar reasoning, the prune step, whose (k-1) subset are not in L_{k-1}, also does not delete any keyword set that could be L_k.

2.8 Discovering of Association Rules

To generate rules, for every large keyword set 1 we find all the non-empty *l*, for every such subset *a*, we output a rule of the form $a \Rightarrow (l\text{-}a)$, if the ratio of support (*l*) to support (*a*) is at least minimum confidence, we consider all subsets of *l* to generate rules with multiple consequent. Generating the subsets of large keyword sets in a recursive depth-first fashion will improve the above said procedure.

Algorithm

for all keyword l_k, $k \geq 2$
call Gen_rules (l_k, l_k)

//The Gen_rules(l_k, l_k) generates all valid association rules

Gen_rules (l_k: large k keyword set, a_m : large m keyword set)
A= { (m-1)-keyword sets $a_{m-1}/a_{m-1} \subset a_m$ }
for all $a_{m-1} \in A$
{
Confidence =support (l_k)/support (a_{m-1})
if (confidence≥ minimum confidence)
{
output rule $a_{m-1} \Rightarrow (l_k - a_{m-1})$
if (m-1 > 1)
Gen_rule (l_k, a_{m-1})
}
}

2.9 Summary

We have explained both phases of Apriori algorithm. The first phase generates large keyword sets and the second phase generates Association Rules. Efficient counting of large keyword sets is thus the focus of most prior work. Complexity of the sequential Apriori algorithm, space and time complexity as well as all possible rules are emphasized is as follows.

Assume that given n transactions and m different keywords.

1) Number of possible Association Rules: O ($m\ 2^{m-1}$)

2) Computation complexity: O ($nm2^m$).

Exponential behavior of the computation complexity for given number of transactions.

3) Memory complexity: O ($n\ 2^m$).

To store all large keyword set, amount of space required is also exponential. It scans the database for every passes of the algorithm.

3. Parallel Mining Of Association Rules

3.1 Dominate Group formation

The success of computerized data management has resulted in the accumulation of huge amounts of data in several organizations. There is a growing perception that the analysis of these databases can turn this passive data into actionable information. The recent emergence of Data Mining or knowledge discovery in databases is a testimony to this trend. Data Mining involves the development of tools that can extract patterns from a large database.

Partitioning of data is an important data-mining problem and can be considered as follows. The input data, also called training set, consists of multiple examples (records), each having multiple keywords. The objective of partitioning or formation of dominant group is to analyze the input data and develop an accurate description or model for each group using the features present in the data.

We summarize the algorithm for generation of dominant groups as follows :

f_i = name of the user
g = number of transactions
ki = set of keywords
while (f_i exists in database)
{
for all f_i in database;
for all g;
search and count (ki , i);
fi ' s keyword matched ≥ 30% or user request % age
increment g ;
}

3.2 Parallel Apriori Algorithm

The algorithm assumes shared-nothing architecture where each of processor has private memory and a private disk. The processors are connected by a communication network and can communicate only by passing messages. The communication primitives used by our algorithms are part of the MPI (Message Passing Interface) [6, 7] communication library supported on the IBM-SP and are keywords set for a message passing communication standard currently under discussion.

Data is evenly distributed on the disks attached to the processors. Each processor's disk has roughly an equal number of transactions. We do not require transactions to be placed on the disks in any special way. We can achieve the parallelism of Apriori Algorithm in different ways; at instructional level or at data level or control level.

Volume: 02 Issue: 01 Pages: 458-463 (2010)

We are following data level parallelism. Using given database generates the dominant group and also divides the database into N partitions. Each partition will be assigned to a processor. Data level parallelism of Apriori algorithm [2, 4] addresses the problem of finding all frequent keyword sets and the generation of the rules form frequent keyword set. Refer to table 2 for a summery of notations used in the algorithm description. We are using superscripts to indicate processor id or rank and subscripts to indicate the pass number (also the size of keyword set).

K keyword set	A keyword set having k keyword set
P^i	Processor with id or rank I
D^i	The keyword set local to processor
N	Number of processor
L_k	Set of frequent k keyword set (those with minimum support) each member of this set has two fields 1) keyword set 2) support count.
C_k	Set of candidate k keyword set (potentially frequent keyword set) each member of this set has two fields 1) keyword set 2) support count.

Table 2 Notation used in parallel algorithm

Given database is divided into N number of partitions.

Our proposed data level parallelism approach used irredundant computations in parallel. We have avoided the communication between the child or slave processors.

1. Each processor P^i receives a 1/N part of the database from the parent or master processors, 0<i<N.
2. Processor P^i performs a pass over data partition D^i and develops local support count for candidates in C_k.
3. Each processor P^i now computes L_k from C_k.
4. Each processor P^i independently makes the decision to terminate or continue to next pass.

3.3 Parallel Rule generation

Parallel implementation of the second phase is to generate rules from frequent keyword sets. Generating rules is much less expensive than discovering frequent keyword sets, as it does not require examination of data. Given the frequent keyword set *l*, rule generation examines each non empty subset a, and generates rule $a \Rightarrow (l-a)$ with support (*l)* and confidence =support (*l*)/ support *(a).*

This computation can efficiently be done by examining the largest subsets of *l* first and processing to smaller subsets, using algorithm. All processor submit the number of rules generated by them to the master or parent processor.

4. Results and Discussion

4.1 Results from sequential Apriori Algorithm

Response time of the sequential Apriori Algorithm for various number of Transactions is as shown in the table 3 and Fig 1 which summarizes the time analysis with a support of 20% and confidence 40%.

No. of Transactions (N)	Response Time (T in m sec)
1000	16
2000	63
4000	264
6000	809
8000	1044
10000	1293
12000	1565
14000	1787

Table 3 Time taken from sequential Apriori Algorithm

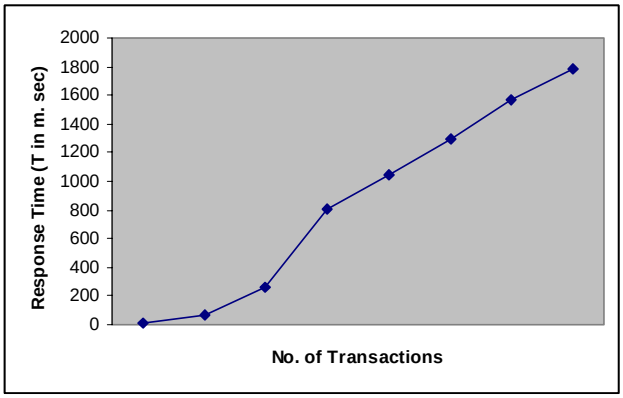

Fig 1 Response time for Sequential algorithm

4.2 Results from Parallel Apriori Algorithm

4.2.1 Response time of the parallel Apriori Algorithm

Table 4 and Fig 2 summarizes time analysis with a support of 25% and numbers of transactions are 14000.

No. of Processors (P)	Response Time (T in m sec)
2	116.75
3	115.56
4	88.23
5	68.17
6	54.22
7	47.64
8	46.64

Table 4 Time taken from Parallel Apriori Algorithm

Volume: 02 Issue: 01 Pages: 458-463 (2010)

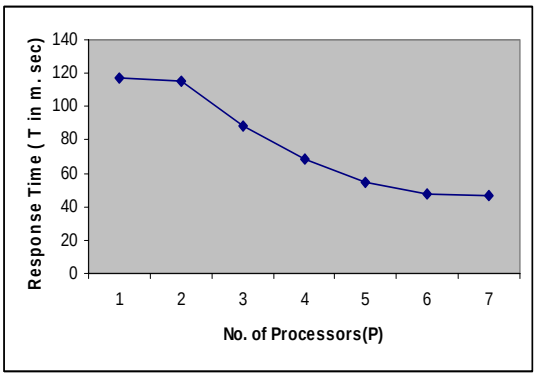

Fig 2 Response time for Parallel algorithm.

4.3 Discussion

The table 3 shows the response time using Sequential Apriori Algorithm for various numbers of transactions with support of 20% and confidence 40%. It shows increase in response time as the number of transaction database.

Table 4 show the scale up results for transactions of 14000 and with a support of 25%. It has been observed that, parallel Apriori Algorithm provides excellent scale up with given number of transactions and keywords.

5. Conclusions and Future Enhancements

In this work, we have studied and implemented the Apriori Algorithm for sequential as well as Parallel Mining of Association Rules

Data mining is becoming more and more of a interest for "ordinary" people. In this paper, we have argued that to make data mining practical for them, data mining algorithms have to be efficient and data mining programs should not require dedicated hardware to run. On these fronts, we can conclude from this thesis that:

- Parallelization is a viable solution to efficient data mining.
- Data parallelism exists in most data mining algorithms.

On implementing the Apriori Algorithm for sequential Mining of Association rules, it has been observed that, it is difficult to handle large amount of data for greater accuracy in a given amount of time. The parallel mining of Association Rules has been generated using apriori algorithm for large transaction database. It has been found that parallel apriori algorithm provides excellent scale up with a given number of transactions and keywords.

5.1 Future Enhancement

- A challenge for data mining in general is how to develop a data mining model so that growth and change in data requires minimal additional mining. We need to tackle this challenge in the context of parallel data mining.
- Visual data mining is a novel approach to deal with the growing flood of information. By combining traditional data mining algorithms with information Visualization Techniques to utilize the advantages of both approaches

References

[1]. R. Agrawal, R.Srikant, "Fast Algorithm for Mining Association Rules", proceedings of 20th VLDB conference PP 478-499 September 1994.

[2]. R. Agrawal, J.C. Shafer "Parallel Mining of Association Rules" IEEE Transactions on Knowledge and Data Engineering, Volumes 8, Number 6, PP 962-969, December 1996.

[3]. R. Agrawal, T.Imielinki, A.Swami "Data base Mining: A performance perspective", IEEE transactions on knowledge and Data Engineering, PP 962-969, December 1993.

[4]. R.Agrawal, J.C. Shafer, Manish Mehta. "SPRINT: A scalable parallel classifier for Data Mining", proceedings of 22nd VLDB conference Mumbai, India, September 1996.

[5]. R.Srikant, R.Agrawal. "Mining Generalized Association Rules", proceedings of 21st VLDB conference Zurich, Switzerland 1995.

[6]. William Gropp, Ewing Lusk, users Guide for MPICH a portable implementation of MPI Technical reports ANL-96/6, Argonne National Laboratory 1996.

[7]. William Gropp and Ewing Lusk " Installation Guide for MPICH a portable implementation of MPI ", Technical report ANL-96/5, Argonne National Laboratory, 1996.

Authors Biography

Mr. Puttegowda D, M.Tech (CSE) working as Senior Lecturer in the Department of computer science, Ghousia College of Engineering, Ramanagarm, Karnataka, currently pursuing research work on Video Mining in NAL research centre, Bangalore, affiliated to University of Mysore and has five years of teaching experience and has published one international paper in the field of Data Mining.

Mr. Deepak N. A, working as Assistant Professor in the department of computer science, Ghousia College of Engineering Karnataka, currently pursuing research work in Mysore University, and has nine years of teaching experience in the field of computer networks, operating systems, database management systems and image processing, and has published two national papers and one international paper in the field of image processing.