ISSN:0975-9646

# Performance Analysis of Association Rule Mining Using Hadoop

K Murali Gopal[1] Ranjit Patnaik[2]
*Dept. of Computer Science and Engineering*
*Gandhi Institute of Engineering and Technology, Gunupur*

**Abstract-**To discover association between different items in large datasets; Association rule mining plays a major role. There are several association algorithms among which the Apriori Algorithm is most suitable one. Actually the Apriori Algorithm is capable of run on single node or computer, due to which it limits the use of this algorithm on large datasets. There have various studies for parallelizing the algorithm. In this paper, Apache G-Hadoop was chosen as the distributed framework to implement the algorithm, to evaluate the performance of the algorithm on G-Hadoop The performance and analysis shows the most suitable platform for distributed association rule mining.

*Keywords: Association rule mining; Hadoop; Apriori Algorithm.*

## 1 Introduction

It is observed that the business intelligence has become an integral part of a successful organization. Organization's growth can be measured by analyzing and making decision based upon the analysis is very important. For Example, in market analysis, association rule mining helps identify what items are purchased together by the customer and generate interesting rules based on transactional data. Many approaches were proposed to mining association rule, but a majority of them are depending on Apriori algorithm as basis. Due to the huge size of dataset, running these algorithms and mining association rules on a single computer is not efficient because it is limited by processor capacity, RAM, storage and various other factors. Hence, it is necessary to develop distributed algorithms to perform association rule mining.

The rapid growth of the Internet and WWW has led to vast amounts of information available on-line In addition, social, scientific and engineering applications have created large amounts of both structured and unstructured information which needs to be processed, analyzed, and linked. Now a days data-intensive computing typically uses modern data center architectures a massive data processing paradigms.

## 2 Background

**What is Hadoop?**

"Hadoop MapReduce is an open source software framework for writing applications which process vast amounts of data (multi-terabyte datasets) in-parallel on large clusters (thousands of nodes) of commodity hardware in a reliable, fault-tolerant manner."

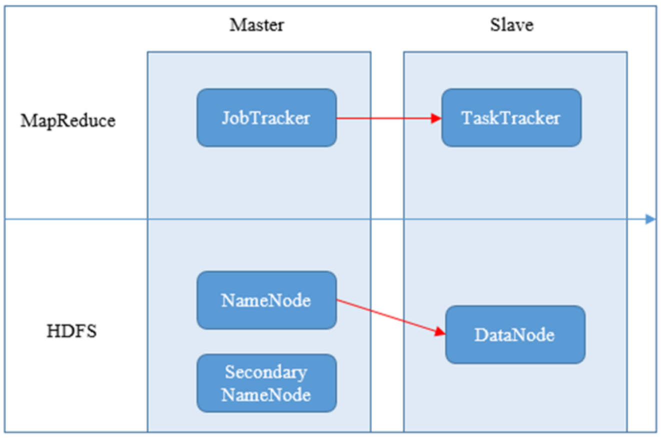
The base Apache Hadoop framework is composed of the following modules:

Hadoop Common – contains libraries and utilities needed by other Hadoop modules;

Hadoop Distributed File System (HDFS) – a distributed file-system that stores data on commodity machines, providing very high aggregate bandwidth across the cluster;

Hadoop YARN – a resource-management platform responsible for managing computing resources in clusters and using them for scheduling of users' applications;

Hadoop MapReduce – an implementation of the MapReduce programming model for large scale data processing.



**Apriori Algorithm:**

Apriori is an algorithm for frequent item set mining and association rule learning over transactional databases. It proceeds by identifying the frequent individual items in the database and extending them to larger and larger item sets as long as those item sets appear sufficiently often in the database. The frequent item sets determined by Apriori can be used to determine association rules which highlight general trends in the database: this has applications in domains such as market basket analysis.

A major drawback of this algorithm is the high I/O costs.

There are three types of Parallel versions of Apriori algorithms:

- Count Distribution: Local processing and global verification
- Data Distribution: Mutual Exclusive Candidate items are distributed.
- Candidate Distribution: Independent Candidate items set for each processor.

The following implements the apriori algorithm:

```
//Map transaction t in data source to all Map nodes;
1.  In each Map node m
        Cm1 = {size 1 frequent items at the node m};
2. In Reduce, compute C1 and L1 with all Cm1; C1 = {size 1 frequent items};
3. min support = num / total items; for example: 33% L1 = {size 1 frequent items min_support};
for (k = 1; Lk!=; null;k++) do begin
4. In each Map node m
// Lmk: Lk mapped to each node m;
// sort to remove duplicated items
Cm (k+1) = Lk join_sort Lmk;
5. In Reduce, use Apriori Property compute Ck+1 with all sorted Cm (k+1); if (k>=3) prune(Ck+1 ); for each transaction t in data source with Ck+1 do
6. In each Map node m increment the count of all candidates in Lm (k+1) that are contained in t end
7. In Reduce, find Lk+1 with Lm(k+1) and
// min_support
Lk+1 = {size k+1 frequent items min_support};
end
return ∪k Lk;
```

## Association Rule Mining

Association rule mining was originally proposed for Market Basket Analysis. By searching for frequent patterns in transactional data sets, interesting associations and correlations between item sets in transactional and relational databases may be discovered.

Development of distributed algorithm for association rule is yet another important angle to address the large centralized data and distributed databases, for mining relations. Many large databases are distributed in nature. The idea of local and global frequent item set was introduced, it finds the local support counts and prunes all infrequent local support counts. After completing local pruning, each site broadcasts messages containing all the remaining candidate sets to all other sites to request their support counts. It then decides whether large itemsets are globally frequent and generates the candidate itemsets from those globally frequent itemsets. The paper focused on communication and synchronization of the systems to reduce the number of candidate itemsets in a distributed data base system. The parallel data mining collected information from each node on each itemset from its local database by hashing method. The information discovered by each node is next shared with other nodes via some communication schemes. It later employed a technique, called clue-and poll, to address the uncertainty due to the partial knowledge collected at each node by selecting a small fraction of the itemsets for the exchange of count information among nodes.

## MapReduce Model

MapReduce is a programming model and an associated implementation for processing and generating large data sets with a parallel, distributed algorithm on a cluster.

MAP(): that performs filtering and sorting

Reduce(): That performs a summary operation

A MapReduce job usually splits the input data-set into independent chunks which are processed by the map tasks in a completely parallel manner. The map, maps the job into key and value. The framework sorts the outputs of the maps, which are then input to the reduce tasks. The input of reduce and output of map must have same type. Typically both the input and the output of the job are stored in a file-system. The output from the 'map' is stored in the temporary file in the HDFS, after completion of the all the map reduce task the file is converted into the permanent one. The framework takes care of scheduling tasks, monitoring them and re-executes the failed tasks.

The MapReduce framework consists of a single master Job Tracker and one slave Task

Tracker per cluster-node. The master is responsible for scheduling the jobs' component tasks on the slaves, monitoring them and re-executing the failed tasks. The number of times a failed job is tried to be executed is configurable. The slaves execute the tasks as directed by the master. The MapReduce framework operates exclusively on <key, value> pairs, that is, the framework views the input to the job as a set of <key, value> pairs and produces a set of <key, value> pairs as the output of the job. It is not necessary that the output of map and output of reduce both be of the same type. They can be of different data types. Input and Output types of a MapReduce job:

(Input) <k1, v1> -> map -> <K2, v2> -> combine -> <K2, v2> -> reduce -> <k3, v3> (output)

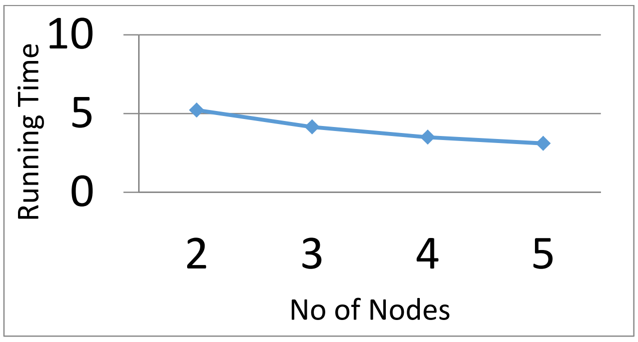
There are various input and output format defined. A programmer can also override these, to have input and output formats, according to the requirement.

## Experimental Setup

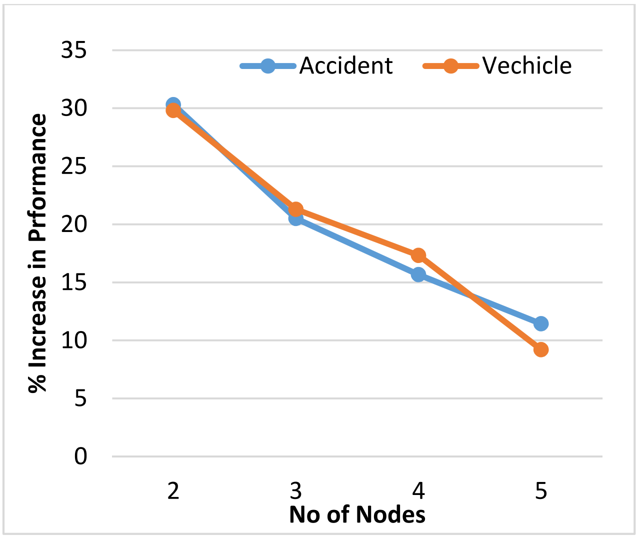
The experiment has been performed based on Count Distribution strategy by implementing Hadoop framework in 7 different computer with i3 processors, 4GB RAM with high speed internet support. The experiment is carried using Accident Data set 2015 as a test cases. The experiment is carried for three times on the above data set as an average is shown as a result of Apriori algorithm on a Hadoop cluster. The accident dataset contain more than 2 lakhs of accident data with 24 attributes.

## Results:

Analysis on number of noes in the cluster and Running Time



Analysis on number of noes in the cluster and % increase in Performance



## CONCLUSION:

The main goal was not to optimize the Apriori algorithm but to test whether it can be implemented on Hadoop with satisfactory results. The experiment shows that increase in nodes, decrease the running time of the program.

## REFERENCES:

[1] Othman Yahya, Osman Hegazy, Ehab Ezat "An Efficient Implementation of Apriori Algorithm Based On Hadoop-MapReduce Model", International Journal of. Reviews in Computing Vol 12, 31st December 2012.
[2] Jongwook Woo "Apriori-Map/Reduce Algorithm", Computer Information Systems Department California State University Los Angeles, C, May 2011.
[3] Xin Yue ,Yang Zhen Liu ,Yan Fu "MapReduce as a Programming Model for Association Rules Algorithm on Hadoop" Department of Computer Science and Engineering University of Electronic Science and Technology of China Chengdu, China, November 2013.
[4] LuJiaheng. Hadoop in Action. Beijing. Machinery Industry Press. 2011.
[5] Tom White. Hadoop: The Definitive Guide. Tsinghua University Press, 2010.
[6] XieYaowei. Savor Hadoop-Hadoop Cluster (Section 8). Shrimp studio. 2012.
[7] Han Jiawei, MichelineKamber, Jian Pei. Data Mining: Concepts and Techniques. Beijing: Machinery Industry Press.2012.
[8] Zhou Shihui. The Research and Application of an Improved Parallel FP-Growth Algorithm Based on Hadoop: [D]. Shandong University .2013.
[9] Sean Owen, Robin Anil, Ted Dunning, Ellen Friedman. Mahout in Action. Manning Publications.2010.
[10] Zhang Xinxia. The Interesting Association Rule Mining Based on Statistical Correlation: [D]. Wuhan University of Science and Technology. 2002.
[11] Ye Fuming. Basic Research on Data Mining Technology. Manufacturing Automation .2011.