ISSN (Print) : 0974-6846
**Indian Journal of Science and Technology,** *Vol 9(26), DOI: 10.17485/ijst/2016/v9i26/93418, July 2016* ISSN (Online) : 0974-5645

# Big Data Analytics with Apache Hadoop MapReduce Framework

**L . Greeshma* and G. Pradeepini**

Department of CSE, K L University, Vaddeswaram - 522502, Andhra Pradesh, India;
greeshma_l@vnrvjiet.in, pradeepini.gera@gmail.com

## Abstract

Huge amount of data cannot be handled by conventional database management system. For storing, processing and accessing massive volume of data, which is possible with help of Big data. In this paper we discussed the Hadoop Distributed File System and MapReduce architecture for storing and retrieving information from massive volume of datasets. In this paper we proposed a WordCount application of MapReduce object oriented programming paradigm. It divides input file into splits or tokens that is done with help of java.util.StingTokenizer class. Output file is represented in the form of <<key>, value>. The experimental results are conducted on Hadoop framework by loading large number of input files and evaluating the performance of Hadoop framework with respect to MapReduce object oriented programming paradigm. In this paper we have examined the performance of the map task and the reduce task by loading more number of files and read-write operations that are achieved by these jobs.

**Keywords:** Hadoop, HDFS, Job Tracker, MapReduce, NameNode

## 1. Introduction

In the recent years people are frequently using most of the social media networking websites such as Facebook, LinkedIn and Twitter[1]. By using Big Data Analytics we can increase productivity and economy of private and public sector[2]. One of the most widely used data mining functionality is association rule mining for storing massive volume of data and extracting most interestingness patterns or knowledge. Due to the volume, velocity and veracity of Big Data identifying association rules and frequent itemsets is very difficult. Several algorithms are existed for mining association rules[3,4]. Using some interestingness measures such as support, confidence, lift and correlation analysis can identify frequent itemsets. For large-scale datasets, identifying frequent itemsets from a single centralized system does not produce effective results[5]. Thereby we require a decentralized systems or Massive Parallel Processor (MPP). In general divide datasets into data partitions. Later distributing partitions to independent computing nodes. Each node have own RAM, CPUs and often own storage. Control node consolidates partition results into one result set, which is returned to the client. SQL-MPP is well established in commercial applications like Teradata, Netezza etc. whereasNon-SQL MPP is well established in Internet properties such as Google, Amazon, Yahoo, etc. In SMP server, storage capacity is 16TB and in Hadoop cluster it is 64TB.

HDFS is scalable, reliable and manageable solution for working with huge amount of data. HDFS has been deployed in cluster of 10 to 4000 data nodes. It is used in the production at Internet scale companies such as Yahoo, Facebook, Twitter and EBay[6]. It is also deployed in many enterprises including financial companies. It is highly scalable file system of 64 nodes and 120 PB. Essentially scalable by adding commodity servers and disks to scale storage and IO bandwidth of file system. HDFS system supports parallel reading and processing of the data (i.e., read, write, rename and append).

Big Data can be handled by using the Hadoop and MapReduce. The multiple node clusters can be set up using the Hadoop and MapReduce. Different size files can be stored in this cluster[7]. Related work is found in

* *Author for correspondence*

**Table 1.** Literature survey

| Author's Name | Technique | Work on MapReduce | Characteristics | Benefits | Limitations |
|---|---|---|---|---|---|
| Greeshma et. al.[8] | AAEMIs | No | Retrieves relevant itemsets from maximal closed itemsets | With the help of this algorithm it prunes the non-closed itemsets | Not efficient in terms of speed and memory |
| Han et. al.[9] | PFP-Tree | No | An Improved version of parallel growth frequent pattern tree algorithm | Linearly Scalable | Consumes more processing time. |
| K. M. Yu J.Zhou[10] | TPFP-Tree | No | TPFP-Tree uses divide and conquer strategy | Efficient for a medium scale dataset | Does not produce a good result for a massive volume of data, which is distributed across multiple nodes. |
| Greeshma et. al.[11] | ISFP-Tree | Yes | ISFP-Tree is constructed using MapReduce object oriented paradigm | Efficient for a large volume of data | Final output file has to be performed which leads sensitivity per loss data. |
| Author's Name | Technique | Work on MapReduce | Characteristics | Benefits | Limitations |
| Thulasi et. al.[12] | Attribute Selection | Yes | Features selection and decision making models are retrieved using Hadoop framework | Predictive analysis | Not scalable based on scheduling job priorities for a decision making. |

the literature survey and there many models listed in the below Table 1 that have been used for retrieving relevant itemsets with and without MapReduce framework.

## 2. Hadoop, HDFS and MapReduce

HDFS system supports parallel reading and processing of the data (read, write, rename and append). In HDFS the files are broken into blocks and these blocks are typically large of size 64 MB or 128 MB. The blocks are stored as files on the data nodes. The blocks are replicated for reliability and typically block replication factor is 3. In the HDFS cluster there is a node called NameNode that manages file system meta data information. NameNode also manages the mapping of the files to the blocks that belongs to it. A failed disk need not be repaired immediately unlike RAID systems. Typically in HDFS cluster repair is done periodically for collection of failures. The reliability data coupled without any immediate attention to failures and fix problems in the cluster makes HDFS easy to manage.

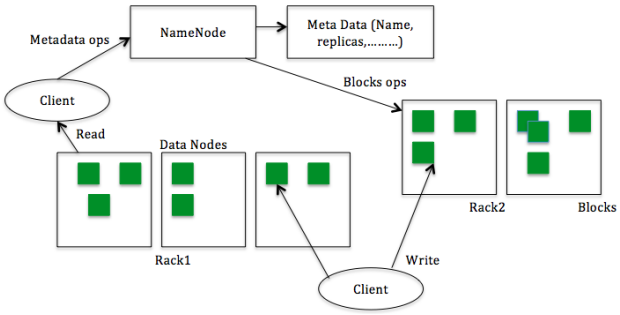


**Figure 1.** HDFS architecture.

MapReduce is at heart of Hadoop. It is programming paradigm that allows for massive scalability across hundreds or thousands of servers in a Hadoop clusters. The MapReduce concept is easy to understand. The term MapReduce actually refers to two separate and distinct that Hadoop program performs. The first is the map job, which takes input data and processes it to produce key/value pairs, which are to be shuffled and sorted and sent as an input to the reduce task. The reduce job takes those key/value pairs and then combines or aggregates them to

produce the final results. As the name MapReduce implies, the reduce job is always performed after the map job. For instance if there are five input files. Each file contains two columns key and value in Hadoop terms.

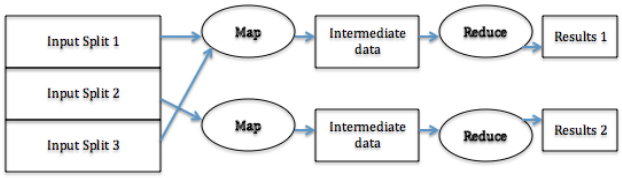


**Figure 2.** MapReduce framework.

## 3. Word Count

WordCount is a simple application of MapReduce object oriented programming paradigm. It evaluates the number of times a key-value pairs which is available in data set. By importing packages such as io.*, conf.*, map.*, util.* from Hadoop API. The Mapper is executed by a map function, which is processed at a time. Input provided can be either a string or a dataset. It divides input file into splits or tokens that is done with help of java.util.StingTokenizer class. Output file is represented in the form of <<key>, value>. A distinct map initiates these key value pairs. An input file containing the following text like:

VNR college workshop VNR Java Lab.

The Reducer for these maps is executed by the Reduce function. It gathers input splits from the map and later it generates intermediate values. Thereby applying shuffling and sorting operation to overcome redundancy. Finally output file is denoted as (< key>, n) where n is denoted as repetition of the token or key that is available in input split file. Reducer output is denoted as follows:

(<VNR>, 2) (<college>, 1) (<Lab>, 1) (<Java>, 1)

## 4. Experimental Setup

The results are experimented on 4GB RAM per node, intel core i5 CPU@ 2.5 GHz, all nodes are connected with 1 Gigabit Ethernet; all clusters are configured on ubuntu 11.10 Linux, Hadoop 2.7.2 and source code is compiled under JDK 1.8.0-65. Each slave contains one data node and task tracker. JobTracker and NameNode are executed on master node.

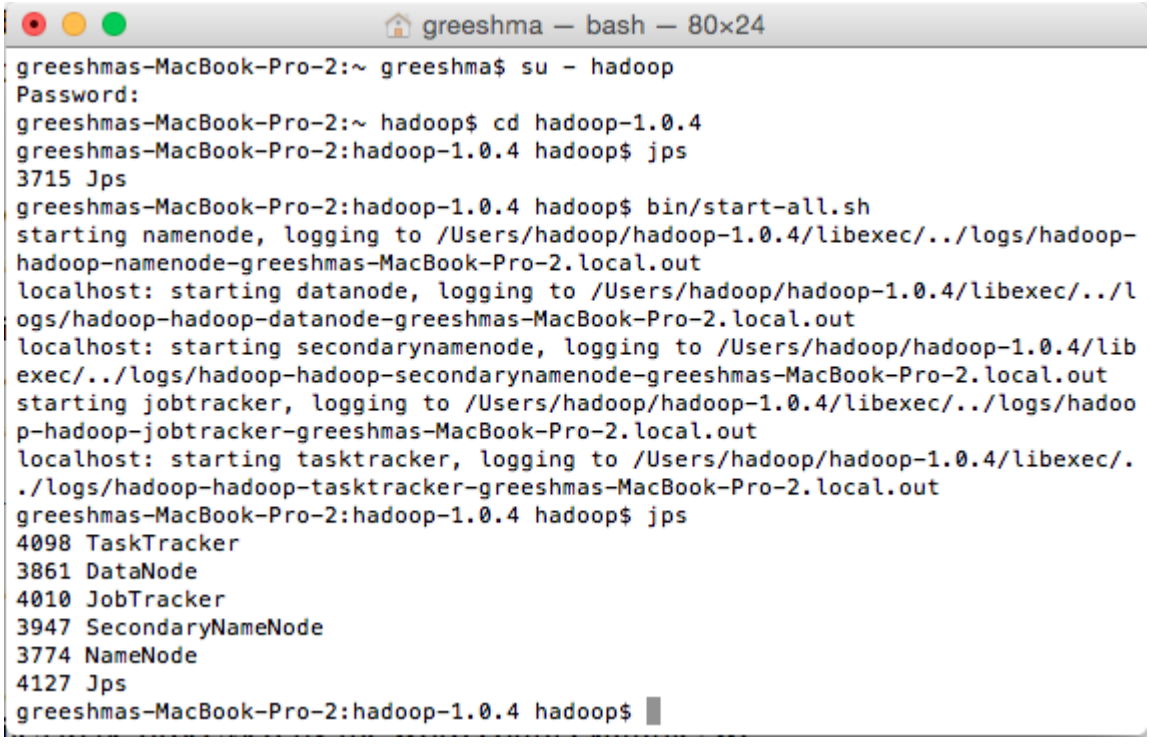


**Figure 3.** Jps output .

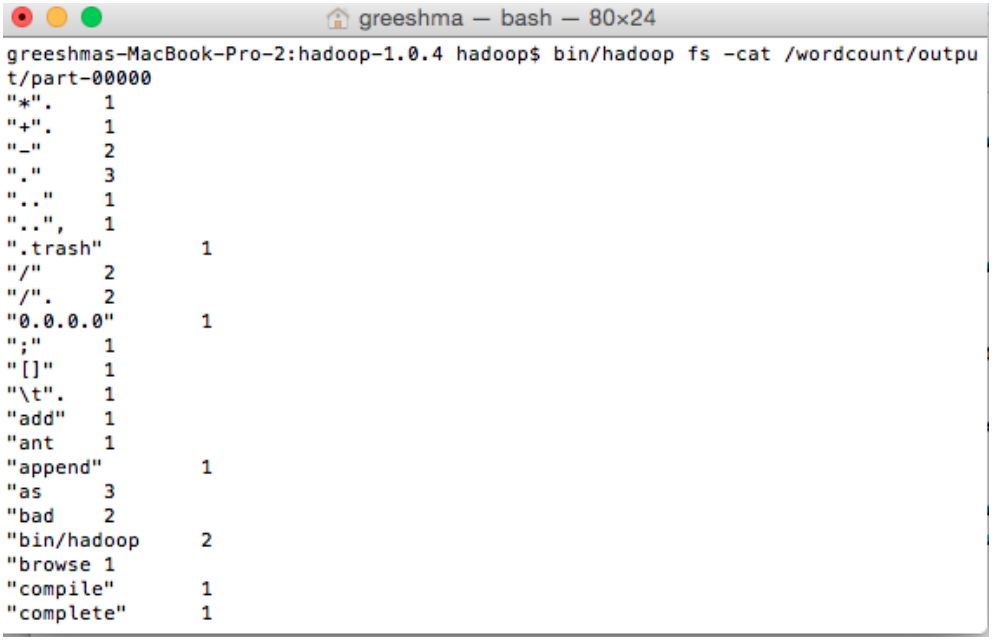


**Figure 4.** WordCount output .

**Table 2.** Experimental results

| | Experiment1 | Experiment2 | Experiment3 | Experiment4 |
|---|---|---|---|---|
| File Bytes Read | 432735 | 1123765 | 653324 | 611225 |
| # MapReduce <Key, value pairs> | 12165 | 22345 | 11321 | 12332 |

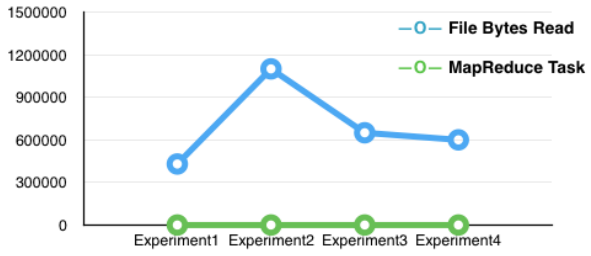


**Figure 5.** File bytes read by MapReduce job.

## 6. Conclusion

In this paper we discussed the Hadoop Distributed File System and MapReduce architecture for storing and retrieving information from massive volume of datasets. The experimental results are conducted on Hadoop framework by loading large number of input files and evaluating the performance of Hadoop framework with respect to MapReduce object oriented programming paradigm. In this paper we have examined the performance of the map task and the reduce task by loading more number of files and read-write operations that are achieved by these jobs.

## 7. References

1. Available from: http://home.web.cern.ch/about/updates/2013/02/cern-data-centre-passes100-petabytes
2. Big Data "has Big Potential to Improve Americans" Lives, Increase Economic Opportunities. Committee on Science, Space and Technology. 2013 Apr. Available from: http://science.house.gov/press-release
3. Shvachko K, Kuang H, Radia S, Chansler R, The Hadoop distributed file system. Mass Storage Systems and Technologies (MSST). IEEE 26th Symposium; 2010 May 3-7.
4. Dhamodaran S, Sachin KR, Kumar R. Big Data implementation of natural disaster monitoring and alerting system in real time social network using Hadoop technology. Indian Journal of Science and Technology. 2015 Sep; 8(22). DOI: 10.17485/ijst/2015/v8i22/79102.
5. Gowthami M, Briskilal J, Jayashree R. Scheduling job queue on Hadoop using hybrid Hadoop fair sojourn protocol. Indian Journal of Science and Technology. 2016 Apr; 9(16). DOI: 10.17485/ijst/2016/v9i16/92231.
6. Somu N, Gangaa A, Sriram VSS. Authentication service in Hadoop using one Time Pad. Indian Journal of Science and Technology. 2014 Apr; 7(S4). DOI: 10.17485/ijst/2014/v7i4/50062.
7. Wang X, Yang C, Zhou J. Clustering aggregation by proba-

bility accumulation. Pattern Recog. 2009; 42(5):668–75.
8. Greeshma L, Pradeepini G. Mining maximal efficient closed itemsets without any redundancy. Springer Third international Conference on Information Systems Design and Intelligent Applications. 2016 Jan; 433:339–47.
9. Han J, Pei J, Yin Y, Mao R. Mining frequent patterns without candidate generation: A frequent-pattern tree approach. Journal of Data Mining and Knowledge Discovery. 2004; 8(1):53–87.
10. Yu KM, Zhou J. Parallel TID-based frequent pattern mining algorithm on PC clusters and grid computing system. Expert System with Applications. 2010; 37(3):2486–94.
11. Greeshma L, Pradeepini G. Input split frequent pattern tree using MapReduce paradigm in Hadoop. JATIT. 2016 Feb; 84:260–71.
12. Bikku T, Rao NS, Akepogu AR. Hadoop based feature selection and decision making models on Big Data. Indian Journal of Science and Technology. 2016 Mar; 9(10). DOI: 10.17485/ijst/2016/v9i10/88905.