ISSN No. 0976-5697

Volume 8, No. 5, May-June 2017

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

MapReduce with Hadoop for Simplified Analysis of Big Data

Ch. Shobha Rani
Research Scholar
Department of Computer Science
Kakatiya University, Warangal, Telangana

Dr. B. Rama
Assistant Professor
Department of Computer Science
Kakatiya University, Warangal, Telangana

Abstract: With the development of web based applications and mobile computer technology there is a rapid growth of data, their computations and analysis continuously in the recent years. Various fields around the globe are facing a big problem with this large scale data which highly supports in decision making. The traditional relational DBMS's were unable to handle this Big Data. The most classical data mining methods are also not suitable for handling this big data. Efficient algorithms are required to process Big Data. Out of the many parallel algorithms MapReduce is adopted by many popular and huge IT companies such as Google, Yahoo, FaceBook etc. In Big data world MapReduce has been playing a vital role in meeting the increasing demands on computing resources affected by voluminous data sets. MapReduce is a popular programming model suitable for Big Data Analysis in distributed and parallel computing. The high scalability of MapReduce is one of the reasons for adapting this model. Hadoop is an open source; distributed programming framework with enables the storage and processing of large data sets. [1] In this paper we try to focus especially on MapReduce with Hadoop for the analytical processing of big data.

Keywords: Big Data, Hadoop, MapReduce, BigData Analytics.

I. INTRODUCTION

In the current era, enormous data is being generated day by day continuously. With the rapid expansion of data, we are moving from the Petabyte to exabytes and zettabytes age. At the same time, new technologies progressing with high speed make it possible to organize and manipulate the voluminous amounts of data presently being generated. With this trend there exists a greater demand for new data storage and analysis methods. [2] Especially, the real world aspects of extracting knowledge from huge data sets have become utmost important.

"Big Data" is the biggest observable fact that has captured the attention of the modern computing industry today since the expansion of Internet globally. Big Data is gaining more popularity today is because of the technological revolutions that have emerged are providing the capability to process data of multiple formats and structures without worrying about the constraints associated with traditional systems and database platforms.

II. IMPORTANCE OF BIG DATA

Big Data can be defined as large volumes of data which is either structured or unstructured and generated at high speeds globally by various new technological devices.

Big Data includes the data that is generated every second by sensors, mobiles, and consumer-driven data from social networks. Big Data is evolving from various facets within organizations legal, sales, marketing, procurement, finance, and human resources departments etc.

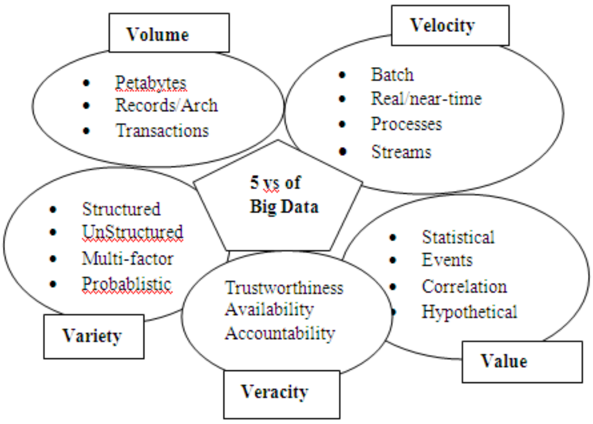


Fig. 1 : 5v's of big data

Along with the three V's, there also exists ambiguity, viscosity, and virality.

- ✓ **Ambiguity**— which comes into existence if the metadata lags behind the clarity of data in Big Data. For example, in a graph, 1 and 0 can depict degree or can depict status as true and false.
- ✓ **Viscosity**—measures the resistance (slow down) to flow in the volume of data. Resistance can manifest in dataflow, business rules, and even be a limitation of technology. For example, social network predictions come under this category, where a number of enterprises just cannot understand what impact is there on business and how it resists the usage of the data in many cases.
- ✓ **Virality**—measures and describes how quickly data is shared in a people-to-people (peer) network.

Big Data is non relational

MapReduce is complementary to DBMS phenomenon, not a competing technology. [3]

- Parallel DBMS are for efficient querying of large data sets. [4]

- Big Data exits mostly in real-manner rather than the traditional Data Warehouse applications.[8]
- Traditional DW architectures (like Exadata, Teradata) are not well suited for Big data applications.
- The architectures like Shared nothing and massively parallel processing are very well suited for big data applications.
- MR-style systems are suitable for complex analytics and especially ETL tasks.
- Parallel DBMS require data to fit into the traditional relational representation of rows and columns.
- In contrast, the MapReduce architecture does not require that data files must and should stick to a particular schema such as the relational data model. That is, the MR programmer can structure their data in any manner or even to have no structure at all. As MR supports both structured and unstructured data this is possible.

Big Data Pillars

- Big Table – consisting of relational tables.
- Big Text – comprising of text in the form of structured, semi-structured data, natural language, and semantic data. [4]
- Big Metadata – collects and stores the data about data stored in big data.
- Big Graphs – Graphs include connections between objects, their semantic discovery, and the degree of separation, linguistic analytics, and subject predicates.

III. MAPREDUCE

MapReduce is a an emerging programming paradigm which is designed for processing extremely large volumes of data in parallel mode by splitting the job into various independent tasks.[3] A MapReduce program in general is a combination of a Map() function and a Reduce() function. The job of Map() is to perform filtering and sorting operations as such, sorting customers by first name into queues, by generating one queue for each name and the Reduce() performs a summary/aggregate operations like counting the number of customers in each queue, thereby yielding the name counts. [3] The "MapReduce System" well known as MapReduce "framework" or “architecture” demonstrates the processing with the distributed servers, running the various tasks in parallel, managing all communications and data transfers between the various parts of the system, and providing for redundant data and fault tolerance.[3]

MapReduce is a framework for processing voluminous data splitted and distributed across huge datasets using a large number of computers (nodes). The group of nodes collectively treated as a cluster, if all nodes are with similar hardware configurations working on the same local network or else the nodes are treated as a grid, if they are geographically shared and distributed with varying hardware specifications. Processing may occur on the data that is stored either in system log files (unstructured) or in a database (structured). MapReduce takes advantage of locality of data, to minimise the data transfer distance.

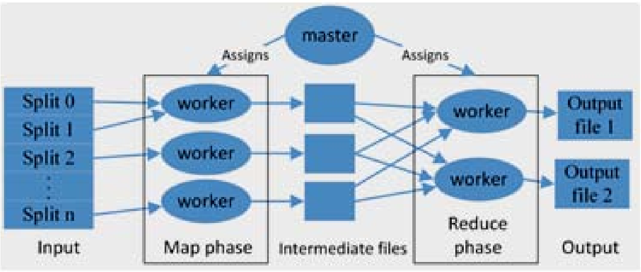


Fig. 2 : Map Reduce workflow

Map Phase: In the map phase, the master node takes the input, divides it into smaller sub-tasks, and distributes them to worker nodes. A worker node may do this again repeatedly, leading to a multi-level tree structure. The worker node processes the smaller task only, and passes the intermediated result back to its master node.

Reduce Phase: During the reduce phase, the master node collects all the intermediated outputs of all the sub-tasks generated by various worker nodes and combines them in some way to form the final output – the solution to the problem it was originally trying to solve.

a) **Input reader**: The input reader splits the input file into appropriate sizes (in practice typically 64 MB to 512 MB as per HDFS) and one split is assigned to one Map function by MapReduce framework. The input reader takes input from stable storage (typically as in our case Hadoop distributed file system) and generates the output as key/value pairs.

b) **Map function**: Each Map function takes a series of key/value pairs generated by the input reader, processes each, and in turn produces zero or more output key/value pairs.[5] The input and output types of the map can be and often are different from each other.

c) **Partition function**: Each Map function output is assigned to a particular reducer by the application' partition function for sharing purposes. The partition function is given as input the key and the number of reducers and it return the index of desired reduce.

d) **Comparison function**: The input for every Reduce is fetched from the machine where the Map run and sorted using comparison function.

e) **Reduce function**: The frame work calls the applications Reduce function for each unique key in the sorted order. It also iterates through the values that are associated with that key and produce zero or more outputs.[6]

f) **Output writer**: It writes the output of the Reduce function to stable storage, usually a Hadoop distributed file system.

Performance:

MapReduce programs do not produce the output with high speed. The main benefit of this programming model is to make use of the optimized shuffle operation of the platform, and the only task of the programmer is to write the Map and reduce functions of the program. While execution, the author of a MapReduce program needs to shuffle the intermediate results.[8] However, the partition function and the amount of data generated by the Map function highly influence the performance of the program. In addition to the partitioner, the Combiner function helps to reduce the amount of data written to storage (disk), and transmitted over the network.

IV HADOOP

Apache Hadoop[1] which is an open-source software in Java is used majorly for distributed storage and processing of extremely large data sets on computer clusters. Apache Hadoop mainly coupled with a storage part (Hadoop Distributed File System (HDFS)) and a processing part (MapReduce). Splitting up of files into large blocks and distributing them on the nodes in the cluster is take care by Hadoop only. It is not the job of the programmer to do the distribution over the cluster, Hadoop itself looks into it. In Hadoop the data processing is done with MapReduce, by transferring the program code to the nodes in parallel based on the data requirements of each node i.e, the process code travels to the node.[7]

The Hadoop framework is encapsulated with the following modules: [1]

- Hadoop Common
- Hadoop Distributed File System (HDFS
- Hadoop MapReduce.

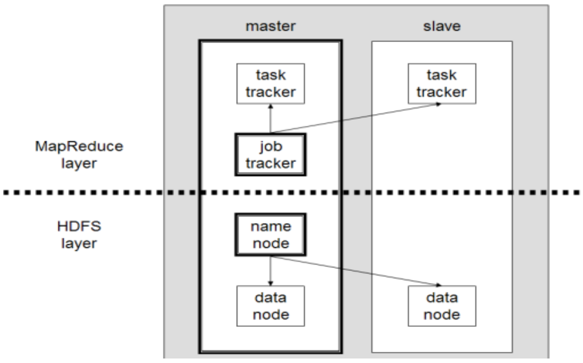


Fig.3 HDFS architecture

V MAPREDUCE IMPLEMENTATION

While designing the MapReduce programs the user may not specify the mappers since it depends on the file size and the block size, where as the number of reducers can be configured by user based on number of mappers. In general the Partitioner decides to choose reducers or else Hadoop takes over the job. With the help of the combiner the network traffic will be highly reduced.

If map() is not defined by the user then the output of Record reader is sent to identity mapper(without any logic) then to reduce without any reducer defined in the program then the output of the identity reducer is stored in the data node itself and is not sent to HDFS.

When multiple mappers are running there may be a situation where some mappers may be running very slow, Hadoop then identifies such slow running jobs and triggers the same job to other data node, this concept is called as Speculator execution in Hadoop.

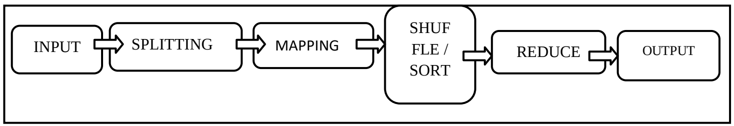


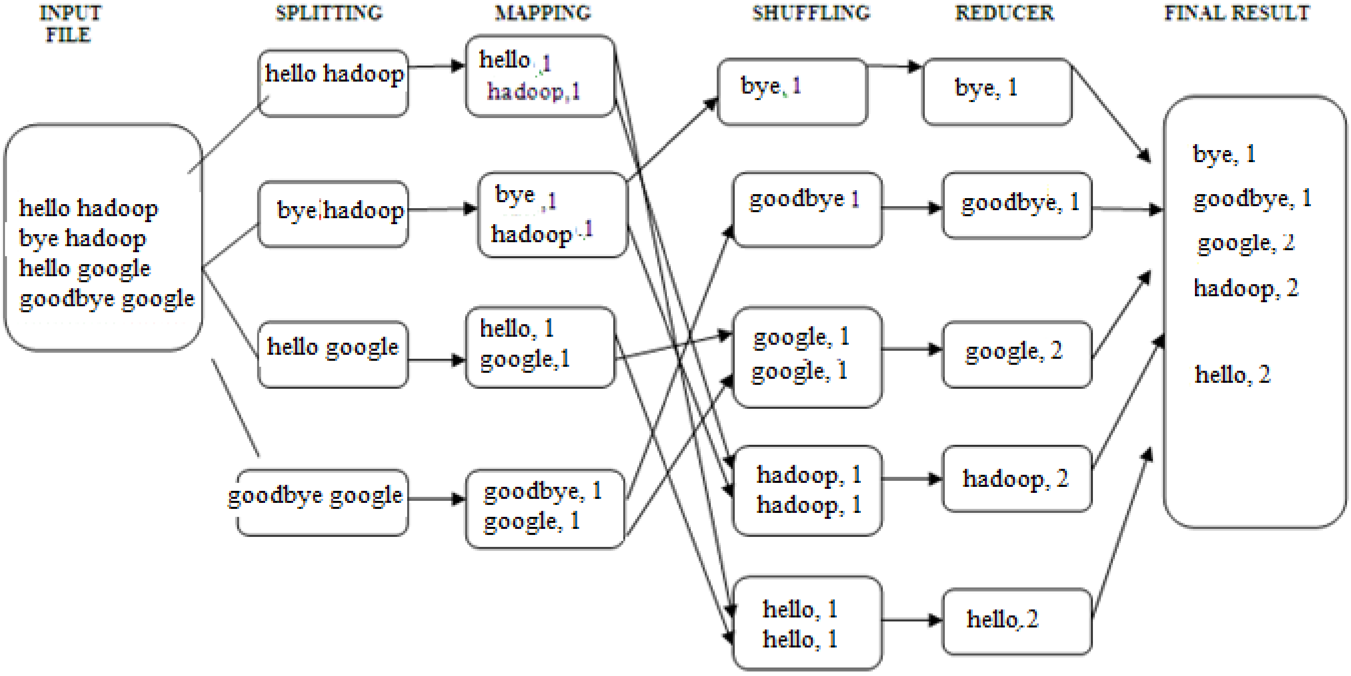
Fig .4. MapReduce Program execution sequence

If we consider a sample input file consisting the text as :

hello hadoop bye hadoop

hello google goodbye google

The internal execution process will be as follows:



VII CONCLUSION

With the invent of new technologies emerging at a rapid rate, one must be very careful to understand the global competition and the big data analysis that support decision making. This paper analyzes the concept of big data analysis and how it can be simplified as from existing traditional relational database technologies. This paper clearly specifies the Hadoop environment, its architecture and how it can be implemented using MapReduce along with various functions. As Big Data Analysis is still in its infancy stage we are sure that this paper helps the researchers to better understand the concepts of Big Data its processing and analysis. Big Data will definitely bring a major social change. Though programming languages like R, SPSS are evolving for Big Data analytics further research is still required to ensure integrity, security for the large data sets being processed. Big Data Analytics should be exploited for sustainable and unbiased society.

REFERENCES

[1] http://hadoop.apache.org,2010

[2] V. Patil, V.B. Nikam, "Study of Mining Algorithm in cloud computing using MapReduce Framework", Journal of Engineering, Computers & Applied Sciences (JEC&AS) Vol.2, No.7, July 2013.

[3] https://en.wikipedia.org/wiki/MapReduce

[4] D. Usha, A.P.S. AslinJenil, " A Survey of Big Data Processing in Perspective of Hadoop and Mapreduce**",** International Journal of Current Engineering and Technology, Vol.4, No.2,April 2014.

[5] S. Ghemawat et al ."The Google File System." ACM SIGOPS Operating Systems Review, 37(5):29–43, 2003.

[6] J. Dean and S. Ghemawat, "Mapreduce: Simplified data processing on large clusters," in Proceedings of OSDI'04: Sixth Symposium on Operating System Design and Implementation, December 2004.

[7] T. White. Hadoop: "The Definitive Guide.",Yahoo Press,2010.

[8] Russom,P "Big Data Analytics", TDWI Best Practices Report, pp.1-40, 2011.