Review of Big Data & Hadoop

Rupali Nehe

Computer Department, Pune University, India
Pratibha College of Commerce and Computer Studies
rups.nehe@gmail.com

Rutuja Chavan

Computer Department, Pune University, IndiaPratibha
College of Commerce and Computer Studies
rutuja.shinde81@gmail.com

Abstract - ***People and devices are constantly generating data, while streaming a video, active in social media, playing games, search any location using GPS. Therefore the rate of data growth is increasing more and more which results in a very large volume of data and make them difficult to capture, manage, process or analyzed. Big Data is huge in Variety, Velocity and volume. Technologies such as MapReduce & Hadoop are used to extract value from Big Data. Hadoop is an open source software project that enables the distributed processing of large data sets across clusters of commodity servers. Hadoop is well adopted, standard-based, open source software framework build on the foundation of Google's MapReduce. Using MapReduce programming paradigm the big data is processed. This paper presents an overview on Big Data, introduction to Hadoop and its components***.

Keywords: big data, Hadoop, Map Reduce, HDFS, YARN.

I. INTRODUCTION

To understand 'Big Data', we first need to know what **'data'** is. Oxford dictionary defines **'data'** as The quantities, characters, or symbols on which operations are performed by a computer, which may be stored and transmitted in the form of electrical signals and recorded on magnetic, optical, or mechanical recording media. " Data that has extra-large Volume, comes from Variety of sources, Variety of formats and comes at us with a great Velocity is normally refer to as Big Data. 'Big Data' is also a **data** but with a **huge size.** In short, such a data is so large and complex that none of the traditional data management tools are able to store it or process it efficiently. Big data can be structured, unstructured or semi-structured, which is not processed by the conventional data management methods. Data can be generated on web in various forms like texts, images or videos or social media posts. Structured data means Any data that can be stored, accessed and processed in the form of fixed format is termed as a 'structured' data. Eg Data stored in a relational database management system is one example of a 'structured' data. Un-structured data poses multiple challenges in terms of its processing for deriving value out of it. Typical example of unstructured data is, a heterogeneous data source containing a combination of simple text files, images, videos etc. Semi-structured data can contain both the forms of data. We can see semi-structured data as a structured in form but it is actually not defined with e.g. a table definition in relational DBMS. Example of semi-structured data is a data represented in XML file. There are certain characteristicsof big data

Volume -The volume is related to the size of data. At present data is in pettabytes and in near future it will be of zettabytes. Variety-The data is not coming from single source it includes semi structured data like web pages, log files etc, raw, and structured and unstructured data. Velocity–The velocity is related to the speed of data coming from different resources. The speed of incoming data is not limited and is also not constant.

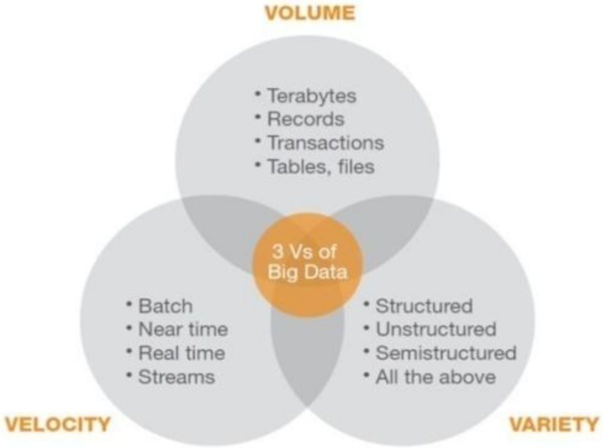


Figure 1 Characteristics of big data

II. HADOOP COMPONENT

Hadoop is an Apache open-source software framework written in java for storing and processing huge data sets on a large cluster of commodity hardware. Hadoop delivers distributed processing power at a remarkably low cost, making it an effective complement to a traditional enterprise data infrastructure. It is designed to scale up from a single computer machine to thousands of machines, with a very high degree of fault tolerance.

Hadoop framework includes following four modules:

Common utilities-These are Java libraries and utilities required by other Hadoop modules. These libraries contain the necessary Java files and scripts required to start Hadoop.

Hadoop YARN- This is a framework for job scheduling and cluster resource management.

Hadoop Distributed File System (HDFS)- A distributed file system that provides high-throughput access to application data.

Hadoop Map Reduce- a programming model for large scale data processing.

Hadoop consists of collection of additional software packages that can be installed on top of or alongside Hadoop, such as Apache Pig, Apache Hive, Apache HBase, Apache Spark etc**.**

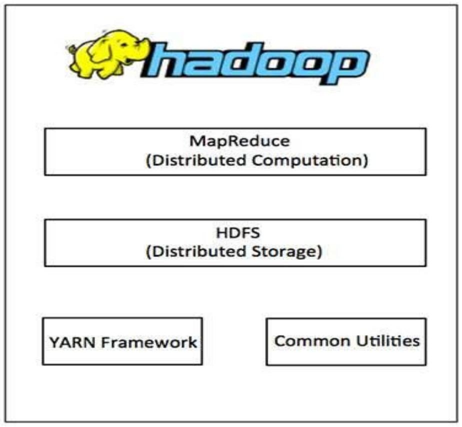


Figure 2.Hadoop Framework

1. HDFS Architecture: HDFS is highly fault tolerant and designed using low-cost hardware. HDFS stores very large amount of data and provides easier access. These files are stored in redundant fashion to rescue the system from possible data losses in case of failure. HDFS follows the master-slave architecture Name node, Data nodes. The system having the name node acts as the master server and it does the following tasks:

- Manages the file system namespace.
- Regulates client's access to files.
- It also executes file system operations such as renaming, closing, and opening files and directories.

For every node (Commodity hardware/System) in a cluster, there will be a data node. These nodes manage the data storage of their system.

- Data nodes perform read-write operations on the file systems, as per client request.
- They also perform operations such as block creation, deletion, and replication according to the instructions of the name node.

Hadoop creates clusters of machines and coordinates work among them.

Clusters can be built with inexpensive computers. If one fails, Hadoop continues to operate the cluster without losing data or interrupting work, by shifting work to the remaining machines in the cluster. HDFS manages storage on the cluster by breaking incoming files into pieces, called "blocks," and storing each of the blocks redundantly across the pool of servers. In other words, the minimum amount of data that HDFS can read or write is called a Block. The default block size is 64MB, but it can be increased as per the need to change in HDFS configuration. In the common case, HDFS stores three complete copies of each file by copying each piece to three different servers.

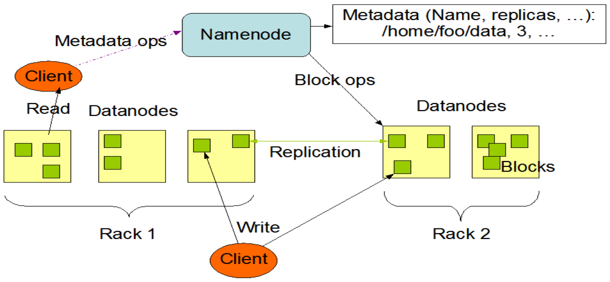


Figure 3 HDFS architecture

2. HBase: HBase adds a distributed, fault-tolerant scalable database, built on top of the HDFS file system, with random real-time read/write access to data. Each HBase table is stored as a multidimensional sparse map, with rows and columns, each cell having a time stamp. It also provides row-level atomicity guarantees, but no native cross-row transactional support. From a data model perspective, column-orientation gives extreme flexibility in storing data and wide rows allow the creation of billions of indexed values within a single table.

HBase has its own Java client API, and tables in HBase can be used both as an input source and as an output target for MapReduce jobs through TableInput/TableOutputFormat. There is no HBase single point of failure.

In addition to HBase, other scalable random access databases are now available. HadoopDB is a hybrid of MapReduce and a standard relational db system. HadoopDB uses PostgreSQL for db layer (one PostgreSQL instance per data chunk per node), Hadoop for communication layer, and extended version of Hive for a translation layer.

3. HIVE: Hive is a data warehousing solution developed on top of Hadoop to meet the big data challenges of storing, managing and processing large data sets without having to write complex Java based MapReduce programs. Hive is a familiar programming model for big data professionals who know SQL but do not have a good grip in programming. Hive is not a relational database or an architecture for online transaction processing. It is particularly designed for online analytical processing systems (OLAP). Hive works in terms of tables. There are two kinds of tables you can create: managed tables whose data is managed by Hive and external tables whose data is managed outside of Hive. Another option Hive provides for speeding up queries is bucketing.

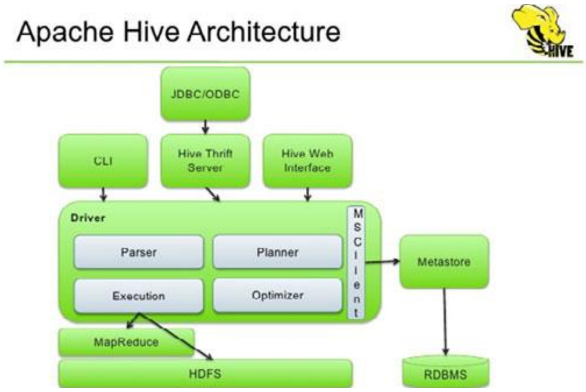
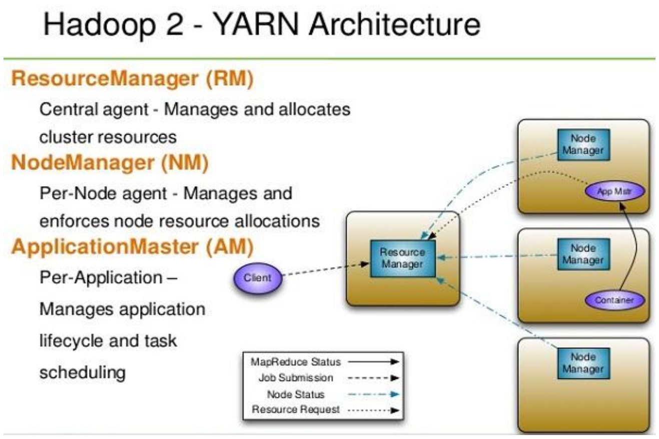


Figure 4.Hive Architecture

4. Pig - Dataflow Language on Hadoop: Pig was initially developed at Yahoo Research around 2006 but moved into the Apache Software Foundation in 2007. Apache Pig is a high level procedural dataflow language on top of Hadoop for processing and analyzing big data without having to write Java based MapReduce code. Apache Pig has RDBMS like features- joins, distinct clause, union, etc. For crunching large files containing semi-structured or unstructured data.
Apache Pig Components

1, Pig Latin: It is a SQL like data flow language to join, group and aggregate distributed data sets with ease.
2. Pig Engine: Pig engine takes the Pig Latin scripts written by users, parses them, optimizes them and then executes them as a series of MapReduce jobs on a Hadoop Cluster.

5. YARN: Hadoop is one of the widely-adopted cluster computing frameworks for processing of the Big Data. Although Hadoop arguably has become the standard solution for managing Big Data, it is not free from limitations. MapReduce has reached scalability limit of 4000 nodes [21]. Another limitation is Hadoop's inability to perform fine-grained resource sharing between multiple computation frameworks. To solve these limitations, the open source community proposed the next generation MapReduce called YARN (Yet Another Resource Negotiator).YARN is included in the latest Hadoop release and its goal is to allow the system to serve as a general data-processing framework. The earlier version of Hadoop did not have YARN but it was added in the Hadoop 2.0 version to increase the capabilities YARN’s basic idea is to split up the two major functionalities of the Job Tracker, resource management and job scheduling into separate daemons. The idea is to have a global ResourceManager and per application ApplicationMaster. The ResourceManager arbitrates resources among all the applications in the system and it has two components: Scheduler and Applications Manager.


Hadoop 2 - YARN Architecture

ResourceManager (RM)
Central agent - Manages and allocates cluster resources

NodeManager (NM)
Per-Node agent - Manages and enforces node resource allocations

ApplicationMaster (AM)
Per-Application – Manages application lifecycle and task scheduling

Figure 4.YARN architecture

III. MAP REDUCE

Hadoop MapReduce is a software framework for distributed processing of large data sets on compute clusters of commodity hardware. The framework allows the specification of an operation to be applied to a huge data set, divide the problem and data, and run it in parallel. There are two functions in MapReduce as follows:

Map – the function takes key/value pairs as input and generates an intermediate set of key/value pairs map(key1,value) -> list<key2,value2>

That is, for an input it returns a list containing zero or more (key, value) pairs:

- The output can be a different key from the input
- The output can have multiple entries with the same key

Reduce – the function which merges all the intermediate values associated with the same intermediate key reduce (key2, list<value2>) -> list<value3>

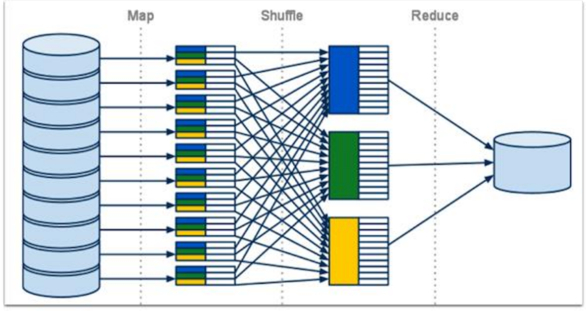


Figure 5: MapReduce Architecture

In Hadoop, these kinds of operations are written as MapReduce jobs in Java. There are a number of higher level languages like Hive and Pig that make writing these programs easier. The outputs of these jobs can be written back to either HDFS or placed in a traditional data warehouse.

IV. CONCLUSION

The paper describes the concept of Big Data along with Volume, Velocity and variety of Big Data. Hadoop MapReduce is a software framework for distributed processing of large data sets on compute clusters of commodity hardware. The MapReduce programming model has been successfully used at Google for many different purposes the model is easy to use, even for programmers without experience with parallel and distributed systems, The paper describes Hadoop which is an open source software used for processing of Big Data. The paper describes Hadoop components such as Apache Pig, Apache Hive, and Apache HBase.

REFERENCES

1. Jonathan Paul Olmsted "Scaling at Scale: Ideal Point Estimation with 'Big-Data" Princeton Institute for Computational Science and Engineering 2014.
2. Jonathan Stuart Ward and Adam Barker "Undefined By Data: A Survey of Big Data Definitions" Stamford, CT: Gartner, 2012.
3. Balaji Palanisamy, Member, IEEE, Aameek Singh, Member, IEEE Ling Liu, Senior Member, IEEE" Cost-effective Resource Provisioning for MapReduce in a Cloud"gartner report 2010, 25
4. Mrigank Mridul, Akashdeep Khajuria, Snehasish Dutta, Kumar N " Analysis of Bidgata using Apache Hadoop and Map Reduce" Volume 4, Issue 5, May 2014 .
5. Shadi Ibrahim⋆ _ Hai Jin _ Lu Lu "Handling Partitioning Skew in MapReduce using LEEN" ACM 51 (2008) 107–113.
6. Kenn Slagter · Ching-Hsien Hsu "An improved partitioning mechanism for optimizing massive data analysis using MapReduce" Published online: 11 April 2013.
7. Shiplap and M. Kaur,"Big Data and Methodology-A review", International Journal of Advanced Research in Computer Science and Software Engineering, Vol.3(10), Oct 2013, pp. 991-995.
8. Varade M and Jethani V," Distributed Metadata Management Scheme in HDFS", Intern- ational Journal of Scientific and Research Publications, Vol 3(5),2013.
9. Yahoo Team,"The Pig Experience", VLDB,2009.
10. V. S. Patil and P. D. Soni,"HADOOP SKELETON & FAULT TOLERANCE IN HADOOP CLUSTERS ",International Journal of Application or Innovation in Engineering & Management (IJAIEM) Vol . 2(2), Feb 2013,pp. 247-250
11. Hadoop Wiki," Apache Hadoop", Accessed: http://wiki.apache.org/hadoop.
12. Jeffrey Dean and Sanjay Google, Inc." MapReduce: Simplified Data Processing on Large Clusters"
13. Kyuseok Shim Seoul National University shim@ee.snu.ac.kr "MapReduce Algorithms for Big Data Analysis"
14. Sanjeev Dhawan1, Sanjay Rathee2, Faculty of Computer Science & Engineering, Research Scholar " Big Data Analytics using Hadoop Components like Pig and Hive" AIJRSTEM 13- 131; © 2013, AIJRSTEM.
15. Vasiliki Kalavri, Vladimir VlassovKTH The Royal Institute of Technology Stockholm, Sweden kalavri@kth.se "MapReduce: Limitations, Optimizations and Open Issues". TrustCom/ISPA/IUCC,Page1031-1038,IEEE,(2013)
16. Apache Hive. Available at http://hive.apache.org
17. Apache HBase. Available at http://hbase.apache.org