ISSN : 2347 - 8446 (Online)
ISSN : 2347 - 9817 (Print)

Big Data and Hadoop

[I]Dr. Rakesh Rathi, [II]Sandhya Lohiya
[I]Assistant Professor, [II]PG Scholar
[I,II]Dept. of Computer Engineering, Govt. Engineering College, Ajmer, Rajasthan, India

Abstract
Today's world is a world of large data, ranging from some petabytes to zetabytes. This kind of large data also called as Big Data and 80% of the world's data is now in unstructured formats, which is created and held on the web. Over the next decade there will be 45 times more data than today. Many applications are required to deal with large data because the results should be obtained within a particular time limit. Hadoop is the solution of all the problems that arises due to massive amount of data which includes audio, video, text, images and etc. It is the open source framework developed by apache.

Index Terms
Big Data, Hadoop, Petabytes, Zetabytes

I. Introduction

Today is the world of computer, technologies and computing devices because new technologies have made computer systems faster and more affordable. Due to the growing number of labs, departments, and institutions high-performance parallel systems are required, such as clusters. Because a single computing system has limited computing resources, so to satisfy the increasing computational demands there is one method to utilize computing resources across multiple distributed computing systems. Because of the sharing of computing resources that are owned and managed by geographically separated different institutions, creates new challenges in software portability and compatibility, resource availability and accessibility, and security and authorization. These challenges are addressed in a field called Grid computing [1]. Now in today's grid sharing is not limited to the resources but it covers the data also. Many frameworks are developed for this kind of challenge but today's data are generating in the size of petabytes to zetabytes. All these kind of data are in unstructured form, which is created and held on the web. This large amount of data is referred as Bigdata. Approximately Over the next decade there will be 44 times more data than today. It's also clear that every data storages even in grid has some limited capacity.
So, there is a challenge to handle this kind of massive data because the result should be in time and in proper format. Hadoop is the solution of Bigdata problem. It is an open source framework developed by apache.
The rest of this article is as follows. In Section 2, some things about Bigdata Section 3, briefly describes the framework architecture of Hadoop in Section 4, Summary. In section 5 acknowledgement and next is references..

II. About Bigdata

Big data [2-4] is a term applied to data sets whose size is beyond the ability of commonly used software tools to capture, manage, and process the data within a tolerable elapsed time.
The usual big data characteristics are:

- **Volume:** there is a lot of data to be analyzed and/or the analysis is extremely intense; either way, a lot of hardware is needed.
- **Variety:** the data is not organized into simple, regular patterns as in a table; rather text, images and highly varied structures—or structures unknown in advance—are typical.
- **Velocity:** the data comes into the data management system rapidly and often requires quick analysis or decision making.

The part of Big Data that relates to the expected explosive growth and creation of new value is the unstructured data mostly arising from the external sources. It will be a great challenge of handling this kind of massive data. Because of that, many applications are required to deal with large data and the results should be obtained within a particular time limit. As unstructured data is increasing rapidly on web, besides handling, managing of these data there are many other problems related to BigData like Resource sharing, Openness, Concurrency, Scalability, Fault tolerance, Heterogeneity, Transparency and hide the details and complexity of accomplishing above challenges from the user is the biggest challenge.
To deal with this large volume of data, it is required that the problem should be divided into parts and these parts should be distributed to multiple machines to handle in parallel. Whenever multiple machines are used in cooperation, the probability of failures rises. In a distributed environment, failures are an expected and common. Networks can experience partial or total failure if switches and routers break down. Due to the network congestion problem data may not arrive at a particular point in time. Individual compute nodes also have failures like overheat, crash, experience hard drive failures, or run out of memory or disk space. Data may be corrupted, or maliciously or improperly transmitted. Multiple implementations of client software may speak slightly different protocols from one another. Clocks may become desynchronized, lock files may not be released, parties involved in distributed atomic transactions may lose their network connections part-way through, etc. But the distributed system should be able to recover from the component failure or transient error condition and continue to make progress. Of course, actually providing such resilience is a major software engineering challenges.
So, there is a need to design such platform that provides high levels of reliability, efficiency, availability and scalability. Hadoop [5, 6] is one such architecture which exploits above mentioned features.

III. Hadoop

Hadoop is an open source software framework used for processing very large amounts of data in parallel across a cluster of servers. It is a large-scale distributed batch processing infrastructure designed to efficiently distribute large amounts of work across a set of machines. Hadoop is a popular implementation of the MapReduce framework. According to Mike Olson [7] the underlying technology was invented by Google back in their earlier days so they could

usefully index all the rich textural and structural information they were collecting, and then present meaningful and actionable results to users. Google's innovations were incorporated into Nutch, an open source project, and Hadoop was later spun-off from that. Yahoo has played a key role developing Hadoop for enterprise applications.

1. Hadoop Architecture

A. Hadoop Node Types:

Hadoop has a variety of node types within each Hadoop cluster; these include DataNodes, NameNodes, and EdgeNodes [5]. Hadoop's architecture is modular, allowing individual components to be scaled up and down as the needs of the environment change. The base node types for a Hadoop cluster are:

- **NameNode** –The NameNode is the central location for information about the file system deployed in a Hadoop environment. Files and directories are represented on the NameNode by inodes, which record attributes like permissions, modification and access times, namespace and disk space quotas. The file content is split into large blocks (typically 128 megabytes) and each block of the file is independently replicated at multiple DataNodes (typically three). An environment can have one or two NameNodes, configured to provide minimal redundancy between the NameNodes. The NameNode[13] is contacted by clients of the Hadoop Distributed File System (HDFS) to locate information within the file system and provide updates for data they have added, moved, manipulated, or deleted. HDFS keeps the entire namespace in RAM.
- **DataNode** –DataNodes make up the majority of the servers contained in a Hadoop environment. Common Hadoop environments will have more than one DataNode, and oftentimes they will number in the hundreds based on capacity and performance needs. Each block replica on a DataNode is represented by two files in the local host's native file system. The first file contains the data itself and the second file is block's metadata including checksums for the block data and the block's generation stamp. The size of the data file equals the actual length of the block and does not require extra space to round it up to the nominal block size as in traditional file systems. Thus, if a block is half full it needs only half of the space of the full block on the local drive. The DataNode [13] serves two functions: It contains a portion of the data in the HDFS and it acts as a compute platform for running jobs, some of which will utilize the local data within the HDFS.
- **EdgeNode** –The EdgeNode is the access point for the external applications, tools, and users that need to utilize the Hadoop environment. The EdgeNode sits between the Hadoop cluster and the corporate network to provide access control, policy enforcement, logging, and gateway services to the Hadoop environment. A typical Hadoop environment will have a minimum of one EdgeNode and more based on performance needs.

The Hadoop framework transparently provides both reliability and data motion to applications. Hadoop implements a computational paradigm named MapReduce where the application is divided into many small fragments of work, each of which may be executed or re-executed on any node in the cluster. Hadoop consists of two key components: the Hadoop Distributed File System (HDFS) and a data processing and job scheduling technique called MapReduce, originated by Google.

B. HDFS

Hadoop provides a distributed file system and a framework for the analysis and transformation of very largedata sets using the MapReduce [8]. An important characteristic of Hadoop is the partitioning of data and computation across thousands of hosts, and executing application computations in parallel close to their data. A Hadoop cluster scales computation capacity, storage capacity and IO bandwidth by simply adding commodity servers. Hadoop clusters at Yahoo! span 25 000 servers, and store 25 petabytes of application data, with the largest cluster being 3500 servers. One hundred other organizations worldwide report using Hadoop. Hadoop comes with a distributed File system called HDFS, which stands for Hadoop Distributed File system. HDFS is Hadoop's flagship File system but Hadoop actually has a general purpose File system abstraction. However, the differences from other distributed file systems are significant. HDFS is highly fault-tolerant and is designed to be deployed on low-cost hardware. HDFS provides high throughput access to application data and is suitable for applications that have large data sets. HDFS stores file system metadata and application data separately. As in other distributed file systems, like PVFS [8] Lustre and GFS [9], HDFS stores metadata on a Dedicated server, called the NameNode. Application data are stored on other servers called DataNodes. All servers are fully connected and communicate with each other using TCP-based protocols.

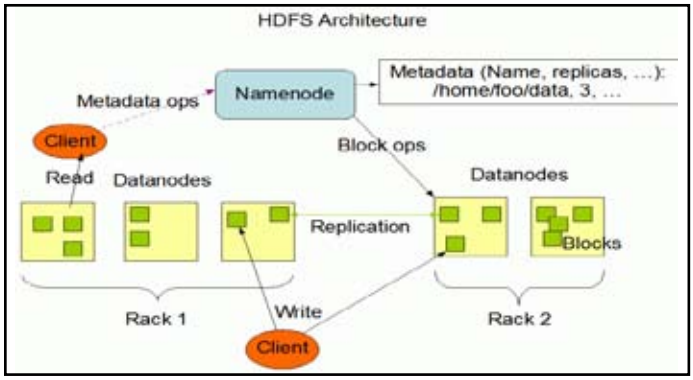

Fig. 1: HDFS Architecture

The above diagram shows the following things.

i. HDFS Design:

- It is based on the design of the Google File System (GFS).
- Individual files are broken into blocks of a fixed size(A problem divided into sub problems).
- Blocks are stored across a cluster of one or more DataNode machines (Data are replicated).
- One file can be composed of several blocks not necessarily stored on the same machines.
- DataNodes that hold each block are chosen randomly across several machines.

ii. HDFS [11] Architecture:

- Master-Worker architecture
- HDFS Master "Namenode"

1. Manages the file system namespace
2. Controls read/write access to files
3. Manages block replication

ISSN : 2347 - 8446 (Online)
ISSN : 2347 - 9817 (Print)

4. Checkpoints namespace and journals namespace changes for reliability
- HDFS Workers "Datanodes"

1. Serve read/write requests from clients
2. Perform replication tasks upon instruction by Namenode

C. MapReduce

Hadoop, a popular implementation of the MapReduce framework is commonly installed on a shared hardware controlled by virtual machine monitors. MapReduce is a programming model and an associated implementation for processing and generating large data sets. Users specify a map function that processes a key/value pair to generate a set of intermediate key/value pairs, and a reduce function that merges all intermediate values associated with the same intermediate key. The basic idea is to partition a large problem into smaller subproblems. The map phase in MapReduce roughly corresponds to the map operation in functional programming, whereas the reduce phase in MapReduce roughly corresponds to the fold operation in functional programming. Actually, It is (M/R) is a technique for dividing work across a distributed system. This takes advantage of the parallel processing power of distributed systems, also reduces network bandwidth as the algorithm is passed around to where the data lives, rather than a potentially huge dataset transferred to a client algorithm. Developers can use MapReduce for things like filtering documents by tags, counting words in documents, and extracting links to related data. The MapReduce framework helps developers divide a query into steps, divide the dataset into chunks, and then run those step/chunk pairs in separate physical hosts.

There are two steps in a MapReduce query:

- Map – data collection phase. Map breaks up large chunks of work into smaller ones and then takes action on each chunk.
- Reduce – data collation or processing phase. Reduce combines the many results from the map step into a single output.

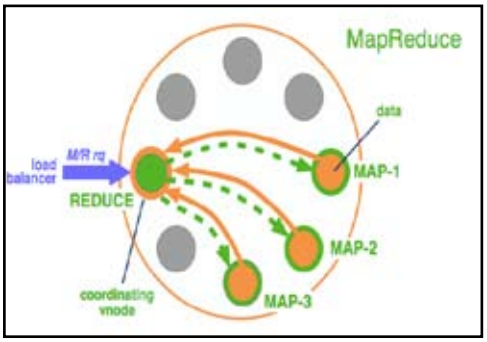

Fig. 2 : MapReduce

On the basis of the framework and performance of the MapReduce have following features [12]:-

1. MapReduce Features
- Complete solution for distributed computing
- Simple, but powerful interface
- Implementation within hours, not weeks
- Detects machine failures and redistributes work
- Avoids data loss due to hard disk failures (together with distributed file system)

2. Architecture(Fig.3.3):
- Dedicated master process identifies worker processes/ machines for map and reduce
- Master partitions input file into M partitions
- Partitions assigned to map workers
- Map workers output to R files on local hard disks (by hash code), master notified
- Each reduce worker reads one output file from the map workers for each input partition (by RPC) & sorts them (many output keys per file!)
- Each reduce worker aggregates data per key

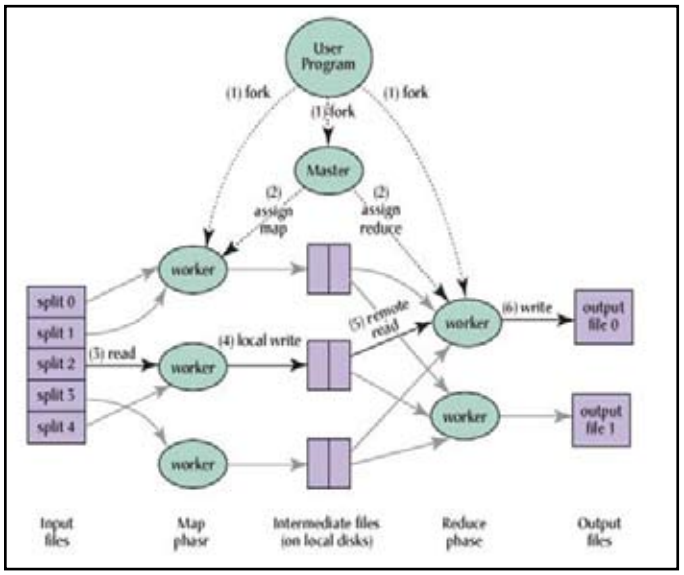

Fig. 3 : Simplified view of MapReduce

D. File I/O Operation And Block Management

The namenode is in periodic communication with the datanodes to ensure proper replication of all the blocks: if there aren't enough replicas due to disk or machine failures or to connectivity losses due to networking equipment failures, the namenode directs the creation of additional copies; if there are too many replicas due to repaired node rejoins the cluster, extra copies are discarded.

To create a new file and write data to HDFS [15], the application client first contacts the namenode, which updates the file namespace after checking permissions and making sure the file doesn't already exist. The namenode allocates a new block on a suitable datanode, and the application is directed to stream data directly to it. From the initial datanode, data is further propagated to additional replicas. In the most recent release of Hadoop as the release 0.20.2, files are immutable they cannot be modified after creation.

IV. Summary

The benefits of using Hadoop in compare to other distributed systems are its flat scalability curve. When execute Hadoop on a limited amount of data and on a small number of nodes may not demonstrate particularly stellar performance as the overhead involved in starting Hadoop programs is relatively high. Other parallel and distributed programming paradigms like MPI (Message Passing Interface) may perform much better on two, four, or perhaps a dozen machines. Alhough the effort of coordinating work among a small number of machines may be better-performed by such systems, the price paid in performance and engineering effort increases non-linearly. A program written in distributed frameworks other than Hadoop may require large amounts of refactoring when scaling from ten to one hundred or one thousand machines. This may involve having the program be rewritten several times; fundamental elements of its design may also put an upper bound on the scale to which the application

can grow.

V. Acknowledgment

This paper describes the term BigData, it's explosive nature which is increasing rapidly in today's scenario, the framework architecture to handle this massive amount of data known as Hadoop , it's strength and weakness. Because this is a survey report, I studied the many research reports, papers, and thesis. I refer those research papers in my survey. So, I extremely thankful to all, who directly or indirectly supported me for this survey.

References

[1]. *Kesselman, C., and I. Foster, The Grid: Blueprint for a New Computing Infrastructure, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1998 ISBN 1-55860-475-8.*

[2]. *Gali Halevi, Dr. Henk Moed: "The Evolution of Big Data as a Research and Scientific Topic-Overview of the Literature". Research Trends Issue 30 September 2012.*

[3]. *Jeff Zakrzewski Vice President – Sogeti USA-" Taming the Beast of Big Data" Local Touch, Global Reach, Infotec 2012.*

[4]. *Julia Lane "Big Data Science Metrics and the black box of Science Policy" Research Trends Issue 30 September 2012*

[5] *Joey Jablonski: "Introduction to Hadoop -A Dell Technical White Paper" Dell Hadoop Introduction Series, 2011*

[6]. *Apache Hadoop. http://hadoop.apache.org/*

[7]. *Mike Olson: "Reasons To Use Hadoop-Open source technology can help organizations manage large-scale data" Ventana Research, 2011-04-06, Article ID: V11-08 .*

[8]. *J. Dean, S. Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters," In Proc. of the 6th Symposium on Operating Systems Design and Implementation, San Francisco CA, Dec. 2004.*

[9]. *P. H. Carns, W. B. Ligon III, R. B. Ross, and R. Thakur. "PVFS: A parallel file system for Linux clusters," in Proc. of 4th Annual Linux Showcase and Conference, 2000, pp. 317–327.*

[10]. *M. K. McKusick, S. Quinlan. "GFS: Evolution on Fast-forward," ACM Queue, vol. 7, no. 7, New York, NY. August 2009.*

[11]. *Jothi Padmanabhan: "Introduction to Grid Computing via MapReduce & Hadoop" Yahoo! Bangalore, Dec 5, 2010*

[12]. *Ralf Schenkel: "MapReduce" Max Planck Institute Informatik By MCI-Cluster of Excellence.*

[13]. *Konstantin Shvachko, Hairong Kuang, Sanjay Radia, Robert Chansler: "The Hadoop Distributed File System" IEEE, 2010.*

[14]. *Vaibhav N. Keskar, Amit A. Kathade, Gopal S. Sarda and Amit D. Joshi : "An Automation Tool for Single-node and Multi-node Hadoop Cluster". Journal of Artificial Intelligence, 6: 82-88, 2013 .*

[15]. *Shvachko, K. and Kuang, H., 2010, The Hadoop Distributed File System, Mass Storage Systems and Technologies (MSST), IEEE 26th Symposium.*

[16]. *An Introduction to HDFS Federation, http://hortonworks.com/blog/an-introduction-to-hdfs-federation/*

Author's Profile

1. First Author

Dr. Rakesh Rathi, Assistant Professor, Department of Computer Science, Govt.. Engineering College, Ajmer-305001 Rajasthan Technical University, Kota.

2. Second Author

Sandhya Lohiya, M.Tech Scholar, Govt. Engineering College, Ajmer -305001, Rajasthan Technical University, Kota.
E-mail Id : sandhyalohiya@gmail.com