International Journal of Trend in Scientific Research and Development (IJTSRD)
International Open Access Journal
ISSN No: 2456 - 6470 | www.ijtsrd.com | Volume - 2 | Issue – 4

HADOOP: A Solution to Big Data Problems using Partitioning Mechanism Map-Reduce

Jagjit Kaur
Assistant Professor
Chandigarh Group of Colleges, (CGC-COE)
Landran, Mohali, Punjab, India

Heena Girdher
Assistant Professor, Department of Computer Applications, Chandigarh Group of Colleges,
Landran, Mohali, Punjab, India

ABSTRACT
With an increased usage of the internet, the data usage is also getting increased exponentially year on year. So obviously to handle such an enormous data we needed a better platform to process data. So a programming model was introduced called Map Reduce, which process big amounts of data in-parallel on large clusters (thousands of nodes) of commodity hardware in a reliable, fault-tolerant manner. Since HADOOP has been emerged as a popular tool for BIG DATA implementation, the paper deals with the overall architecture of HADOOP along with the details of its various components.

Keywords : *Hadoop, Big data, HDFS, YARN, SAS etc*

INTRODUCTION
Hadoop is open source software for reliable, scalable and distributed computing. It's an Java-based programming framework that supports the processing and storage of extremely large data sets in a distributed computing environment. It is part of the Apache project. Hadoop makes it possible to run applications on systems with thousands of hardware nodes, and to handle thousands of terabytes of data. This approach facilitates:-

- Rapid data transfer
- Cost effective and flexible
- Distribute data and computation
- Tasks are independent
- Simple programming model. The end-user programmer only writes map-reduce tasks.
- Highly scalable storage platform
- This approach lowers the risk of catastrophic system failure and unexpected data loss.

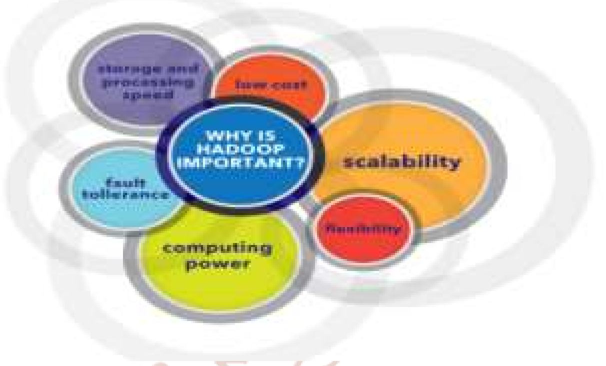


Fig1.1-Various features of Hadoop

Challenges:-

- Security concerns
- Vulnerable by nature
- Not fit for small data
- Potential stability issues

Doug cutting built a parallel Hadoop Distributed File System (HDFS). ***The software or framework that supports HDFS and MapReduce is known as Hadoop***. Hadoop is an open source and distributed by Apache.

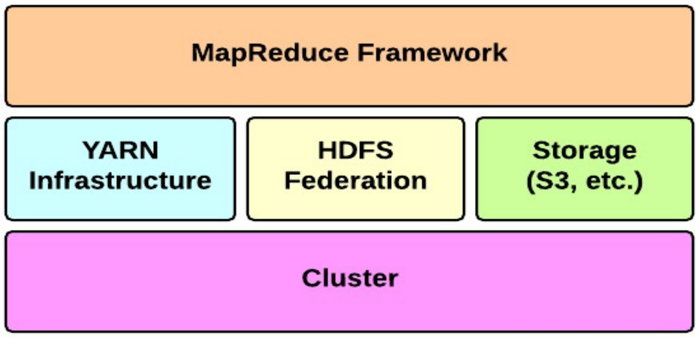
Hadoop Framework :-

Currently, four core modules are included in the basic framework from the apache foundation:

Hadoop Common – The libraries and utilities used by other Hadoop modules.
Hadoop Distributed File System (HDFS) – the Java-based scalable system that stores data across multiple machines without prior organization.

YARN – **(Yet Another Resource Negotiator)** provides resource management for the processes running on Hadoop.

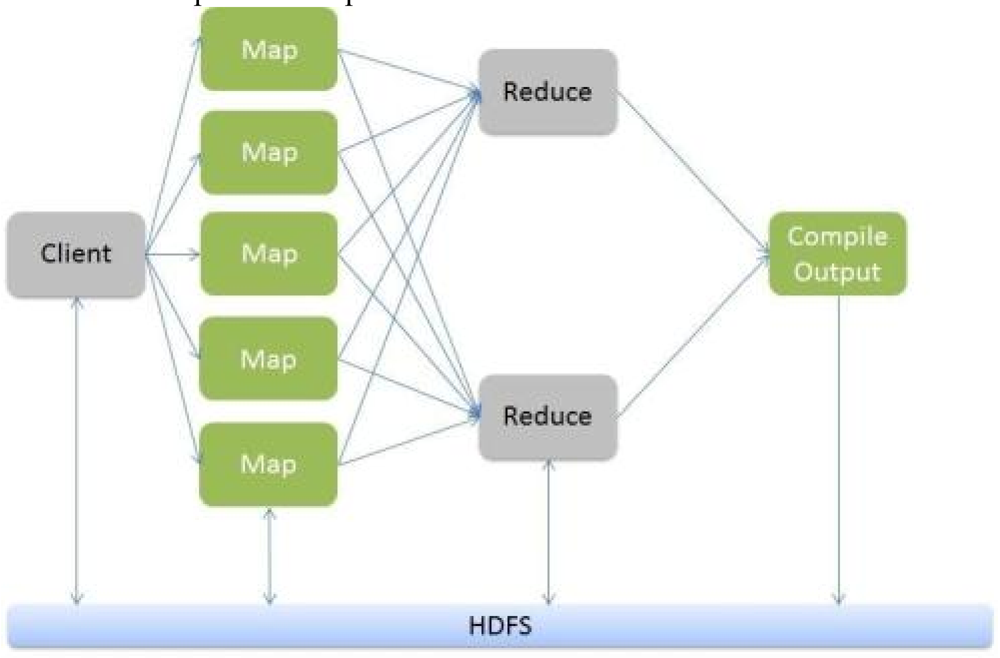
MapReduce – A parallel processing software framework. It is comprised of two steps. Map step is a master node that takes inputs and partitions them into smaller sub-problems and then distributes them to worker nodes. After the map step has taken place, the master node takes the answers to all of the sub-problems and combines them to produce output.



[Fig 1.2-Hadoop Framework Model]

Hadoop Map Reduce:-

MapReduce is mainly used for parallel processing of large sets of data stored in Hadoop cluster. Initially, it is a hypothesis specially designed by Google to provide parallelism, data distribution and fault-tolerance. MR processes data in the form of key-value pairs. A key-value (KV) pair is a mapping element between two linked data items - key and its value.



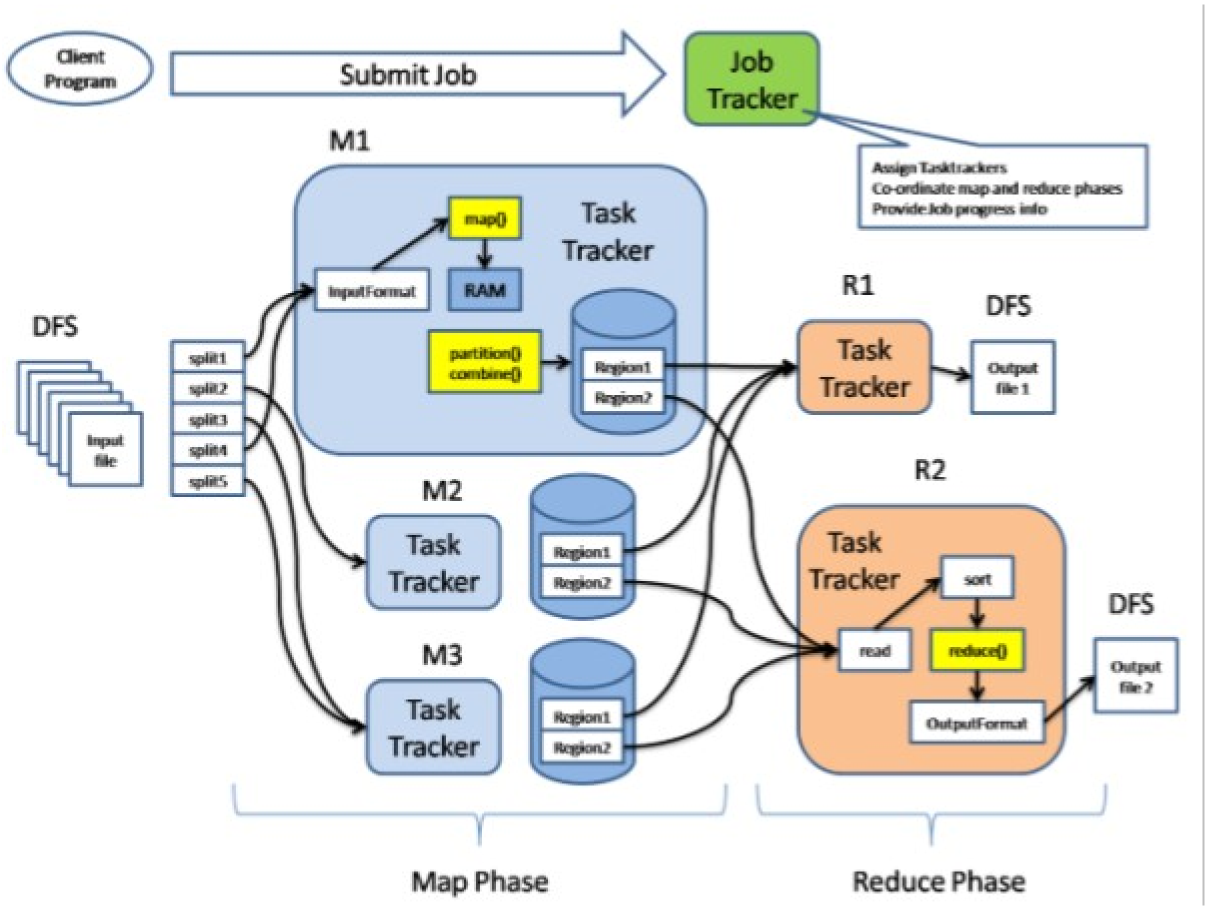
[Fig1.3-Hadoop-Map Reduce Model]

This algorithm divides the task into small parts and assigns those parts to many computers connected over the network, and collects the results to form the final result dataset.

- The first is the map job, which takes a set of data and converts it into another set of data, where individual elements are broken down into tuples (key/value pairs).

- The reduce job takes the output from a map as input and combines those data tuples into a smaller set of tuples. As the sequence of the name MapReduce implies, the reduce job is always performed after the map job.

Hadoop Map Reduce architecture



[Fig-1.4-Hadoop Map reduce Architecture]

Map reduce architecture consists of mainly two processing stages. First one is the map stage and the second one is reduce stage. The actual MR process happens in task tracker. In between map and reduce stages, Intermediate process will take place. Intermediate process will do operations like shuffle and sorting of the mapper output data. The Intermediate data is going to get stored in local file system.

Mapper Phase

In Mapper Phase the input data is going to split into 2 components, Key and Value. The key is writable and comparable in the processing stage. Value is writable only during the processing stage. Suppose, client submits input data to Hadoop system, the Job tracker assigns tasks to task tracker. The input data that is going to get split into several input splits.

Intermediate Process

The mapper output data undergoes shuffle and sorting in intermediate process. The intermediate data is going to get stored in local file system without having replications in Hadoop nodes. This intermediate data is the data that is generated after some computations based on certain logics. Hadoop uses a Round-Robin algorithm to write the intermediate data to local disk. There are many other sorting factors to reach the conditions to write the data to local disks.

Reducer Phase

Shuffled and sorted data is going to pass as input to the reducer. In this phase, all incoming data is going to combine and same actual key value pairs is going to write into hdfs system. Record writer writes data from reducer to hdfs. The reducer is not so mandatory for searching and mapping purpose.

Hadoop - HDFS File System

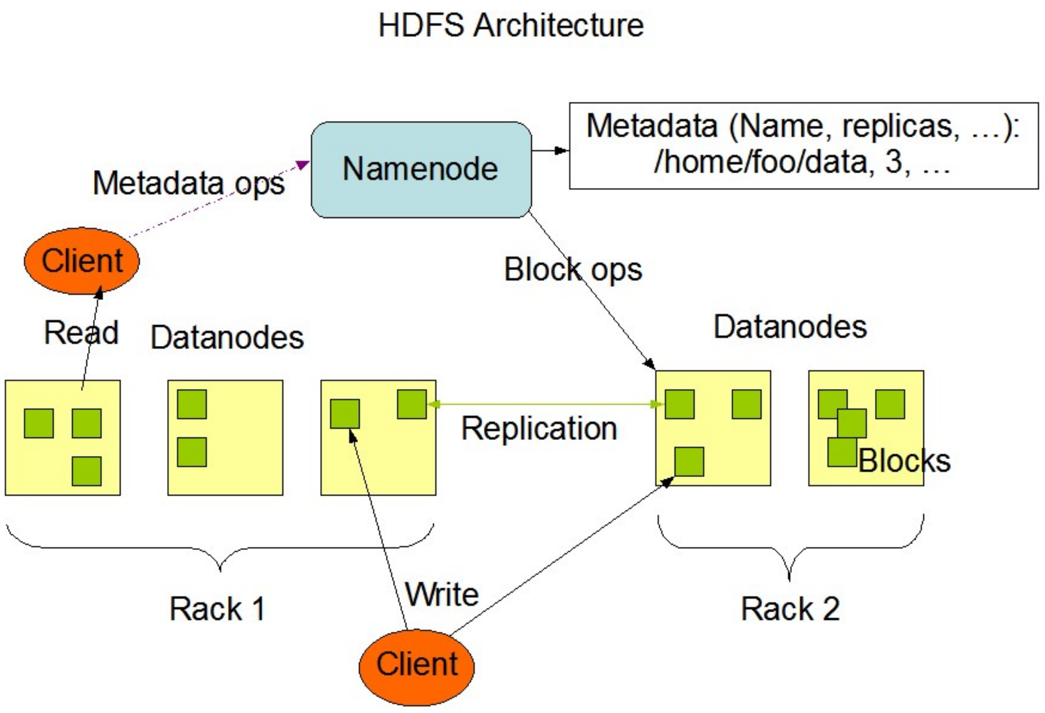
Hadoop File System was developed using distributed file system design. It is run on commodity hardware. Unlike other distributed systems, HDFS is highly fault olerant and designed using low-cost hardware.

HDFS holds very large amount of data and provides easier access. To store such huge data, the files are stored across multiple machines. These files are stored in redundant fashion to rescue the system from possible data losses in case of failure. HDFS also makes applications available to parallel processing.

Features of HDFS

- It is suitable for the distributed storage and processing.
- Hadoop provides a command interface to interact with HDFS.
- The built-in servers of name node and data node help users to easily check the status of cluster.
- Streaming access to file system data.
- HDFS provides file permissions and authentication.

HDFS Architecture:-



[Fig 1.5-HDFS Architecture]

HDFS follows the master-slave architecture and it has the following elements.

Name Node

The name node is the commodity hardware that contains the GNU/Linux operating system and the name node software. It is software that can be run on commodity hardware. 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.

Data Node

The data node is a commodity hardware having the GNU/Linux operating system and data node software. 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.

Block

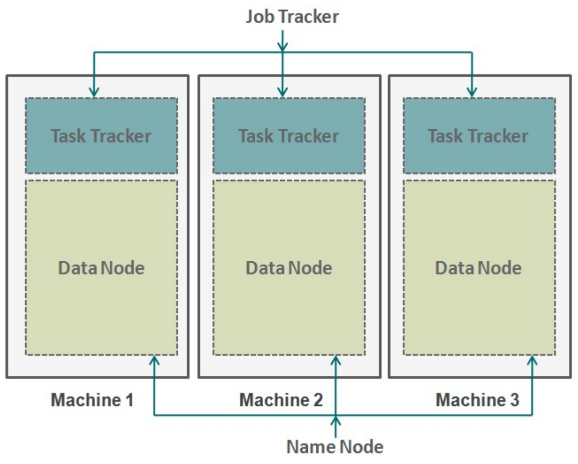
Generally the user data is stored in the files of HDFS. The file in a file system will be divided into one or more segments and/or stored in individual data nodes. These file segments are called as blocks. In other words, the minimum amount of data that HDFS can read or write is called a Block. The default block size is 64MB, but it can be increased as per the need to change in HDFS configuration.

Goals of HDFS

- **Fault detection and recovery :** Since HDFS includes a large number of commodity hardware, failure of components is frequent. Therefore HDFS should have mechanisms for quick and automatic fault detection and recovery.
- **Huge data sets :** HDFS should have hundreds of nodes per cluster to manage the applications having huge datasets.
- **Hardware at data:** A requested task can be done efficiently, when the computation takes place near the data. Especially where huge datasets are involved, it reduces the network traffic and increases the throughput.

How Application works:-

There are machine nodes on the bottom. Data node is also known as hadoop distribution file system. Data node contains the entire set of data and Task tracker does all the operations. . Job Tracker makes sure that each operation is completed and if there is a process failure at any node, it needs to assign a duplicate task to some task tracker. Job tracker also distributes the entire task to all the machines.



[Fig-1.6 Hadoop Architecture Model]

How Does Hadoop Work?

It is quite expensive to build bigger servers with heavy configurations that handle large scale processing, but as an alternative, you can tie together many commodity computers with single-CPU, as a single functional distributed system and practically, the clustered machines can read the dataset in parallel and provide a much higher throughput.

Moreover, it is cheaper than one high-end server. So this is the first motivational factor behind using Hadoop that it runs across clustered and low-cost machines.
Hadoop runs code across a cluster of computers. This process includes the following core tasks that Hadoop performs:

- Data is initially divided into directories and files. Files are divided into uniform sized blocks of 128M and 64M (preferably 128M).
- These files are then distributed across various cluster nodes for further processing.
- HDFS, being on top of the local file system, supervises the processing.
- Blocks are replicated for handling hardware failure.
- Checking that the code was executed successfully.
- Performing the sort that takes place between the map and reduce stages.

Data Management for Hadoop:-Big data skills are in high demand. Now business users can profile, transform and cleanse data – on Hadoop or anywhere else it may reside – using an intuitive user interface. Data analysts can run SAS code on Hadoop for even better performance. With **SAS**("Statistical Analysis System"), We can:

- **Access and load Hadoop data fast:-** Turn big data into valuable data with quick, easy access to Hadoop and the ability to load to and from relational data sources as well as SAS datasets.

- **Stop the "garbage in, garbage out" cycle:-** Integrated data quality delivers pristine data that fuels accurate analytics amplified by the power of Hadoop.

- **Put big data to work for you:-** Transform, filter and summarize data yourself, and get more value from your big data.

- **Get more out of your computing resources:-** Optimize your workloads, and gain high availability across the Hadoop cluster.

WHY SAS("Statistical Analysis System")?

- **Better productivity through faster management of big data:-** In-Hadoop data quality and code execution take advantage of MapReduce and YARN to speed the process of accessing trusted data.

- **Big data management:-** Big data is becoming the backbone of business information. We help business and IT work together to deliver big data that's enterprise ready – no need to write code (unless you want to).

- **Data you can trust:-** Make big data better. SAS provides multiple data integration and data quality transformations to profile, parse and join your data without moving it out of Hadoop.

Conclusion

We have entered on Big Data era. The paper describes the concept of Big Data analytics with the help of partitioning mechanism-MAP Reduce and describes the management of large amount of data through HDFS. The paper also focuses on Big Data processing problems. These technical challenges must be addressed for efficient and fast processing of Big data.Hadoop provides solution to all big data problem.

References:-

1) S.Vikram Phaneendra & E.Madhusudhan Reddy "Big Data- solutions for RDBMS problems- A survey" In 12th IEEE/IFIP Network Operations & Management Symposium (NOMS 2010) (Osaka, Japan, Apr 19{23 2013).

2) Kiran kumara Reddi & Dnvsl Indira "Different Technique to Transfer Big Data : survey" IEEE Transactions on 52(8) (Aug.2013) 2348 { 2355}

3) Z. Zheng, J. Zhu, M. R. Lyu. "Service-generated Big Data and Big Data-as-a-Service: An Overview," in Proc. IEEE BigData, pp. 403-410, October 2013. A . Bellogín, I. Cantador, F. Díez, et al., "An empirical comparison of social, collaborative filtering, and hybrid recommenders," ACM Trans. on Intelligent Systems and Technology, vol. 4, no. 1, pp. 1-37, January 2013.

4) W. Zeng, M. S. Shang, Q. M. Zhang, et al., "Can Dissimilar Users Contribute to Accuracy and Diversity of Personalized Recommendation?," International Journal of Modern Physics C, vol. 21, no. 10, pp. 1217- 1227, June 2010.

5) T. C. Havens, J. C. Bezdek, C. Leckie, L. O. Hall, and M. Palaniswami, "Fuzzy c-Means Algorithms for Very Large Data," IEEE Trans. on Fuzzy Systems, vol. 20, no. 6, pp. 1130-1146, December 2012.

6) Z. Liu, P. Li, Y. Zheng, et al., "Clustering to find exemplar terms for keyphrase extraction," in Proc. 2009 Conf. on Empirical Methods in Natural Language Processing, pp. 257-266, May 2009

7) Sameer Agarwal†, Barzan MozafariX, Aurojit Panda†, Henry Milner†, Samuel MaddenX, Ion Stoica "BlinkDB: Queries with Bounded Errors and Bounded Response Times on Very Large Data" Copyright © 2013ì ACM 978-1-4503-1994 2/13/04

8) Yingyi Bu _ Bill Howe _ Magdalena Balazinska _ Michael D. Ernst "The HaLoop Approach to Large-Scale Iterative Data Analysis" VLDB 2010 paper "HaLoop: Efficient Iterative Data Processing on Large Clusters.