Welcome to Hadoop

This document should help one make an entry into the world of Apache Hadoop and Big Data.

Understanding Big Data

Big Data is an umbrella term used to refer to large and often complex tracts of data that cannot be processed using traditional methods and tools. Big Data is often characterized by the 3 'V's, though some authors also add **Veracity** and **Value** to the list.

Velocity : the speed at which data is produced
Volume : the quantity of data
Variety : the amount of variability in incoming data that has to be accounted for

Why do traditional solutions fail?

- Given the magnitude of the data, one may find traditional methods to be inefficient, especially due to difficulties in **scaling** and **cost.** Processing the data can also take longer and require heavier computing resources.
- As the incoming data generally tends to be unstructured, traditional systems often lack the **flexibility** to accommodate the variety.
- Centralized data banks generally used in traditional systems are susceptible to **reliability** and **security** issues. As the volume of data increases, traditional operations for accessing records can become inefficient.

What in the world is Hadoop, actually?

In the words of the developers, "**The Apache Hadoop software library is a framework that allows for the distributed processing of large data sets across clusters of computers using simple programming models. It is designed to scale up from single servers to thousands of machines, each offering local computation and storage.**" In essence, the Hadoop framework allows large tasks to be distributed among clusters of computing units.

And all that complication is supposed to help?

- **Scalability** : One can easily add more compute and storage clusters or nodes to an existing Hadoop system to handle growths in the data stream.
- **Efficient Processing** : Hadoop works by splitting the data into chunks that can then each be processed on multiple parallel computing nodes.
- **Data Safety** : Hadoop's proprietary HDFS (Hadoop Distributed File System) splits data among multiple storage nodes and ensures data safety in the event of a node failure through **replication** and **redundancy**.
- Additionally, the Hadoop architecture allows for efficient access to the data and costs lower than that of traditional large computing systems, given that it can be run even on consumer-grade hardware.

Admiring the architecture.

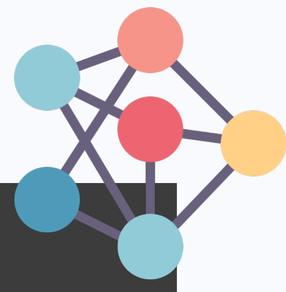
The Hadoop framework can be thought of as having three layers:

- **Application Layer** : Hadoop MapReduce and Other Applications
- **Resource Management Layer** : Hadoop YARN
- **Storage Layer** : Hadoop HDFS

Delving into the Hadoop Architecture

Hadoop implements a master-slave topology, wherein we have one master node and multiple slave nodes. The master node's function is to assign a task to various slave nodes and manage resources. The slave nodes do the actual computing. Slave nodes store the real data whereas on master we have metadata. The architecture comprises of three major layers:-

- MapReduce
- Yarn
- HDFS (Hadoop Distributed File System)



Understanding the layers

MapReduce is a programming model used to process large data sets in parallel and distribute processing across multiple nodes in a Hadoop cluster.
It consists of two main phases: Map and Reduce.

1. Map Phase: In the Map phase, the input data is divided into smaller chunks, called split, and each split is processed by a separate node in the Hadoop cluster. The map function processes each split and produces intermediate key-value pairs. These intermediate key-value pairs are grouped by key and shuffled to the reduce nodes.
2. Reduce Phase: In the Reduce phase, the reduce function processes the intermediate key-value pairs produced by the map phase. The reduce function takes a set of values for each key and aggregates them into a smaller set of values. The final output of the reduce phase is stored in the HDFS (Hadoop Distributed File System).

This process helps to handle large data sets efficiently and provides scalability by distributing the processing across multiple nodes.

Delving into the Hadoop Architecture

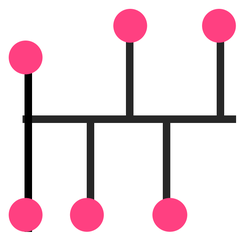
Understanding the layers

Hadoop YARN (Yet Another Resource Negotiator) is a resource management framework in Apache Hadoop that manages the computing resources and scheduling of jobs in a Hadoop cluster. It was introduced in Hadoop 2.0 and allows multiple data processing engines such as MapReduce, Spark, and Hive to run and process data stored in HDFS (Hadoop Distributed File System). Inside the YARN framework, we have two daemons ResourceManager and NodeManager. The ResourceManager arbitrates resources (CPU, memory, network etc.) among all the competing applications in the system, while the NodeManager monitors the resource usage by the cluster and reports the same to ResourceManager.

Understanding the layers

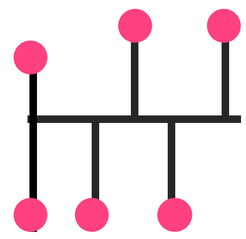
Hadoop HDFS (Hadoop Distributed File System) is a distributed file system designed to store and manage large amounts of data on commodity hardware. In HDFS, data is stored across multiple nodes in a cluster, allowing for parallel processing of data by MapReduce and other Hadoop tools.When a user wants to read a file from HDFS, the HDFS client first contacts the NameNode (the master node in HDFS) to determine the locations of the blocks making up the file. The client then retrieves the blocks from the DataNodes (the worker nodes in HDFS) where the data is stored and combines them to reconstruct the original file. This allows for parallel processing of data. Additionally, HDFS supports the storage of structured, semi-structured, and unstructured data.

Basic Terminologies



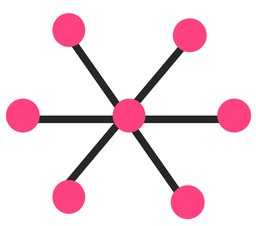
Cluster

A group of interconnected computers working together as a single system in Hadoop.



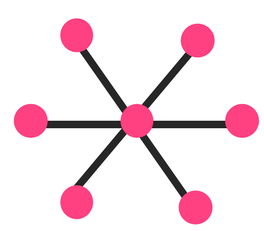
Node

A single machine in a Hadoop cluster.



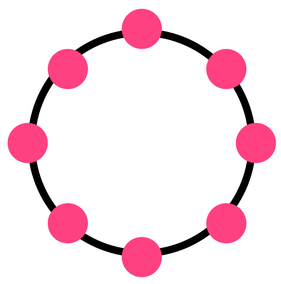
Data Block

The smallest unit of data in HDFS, typically 128 MB in size.



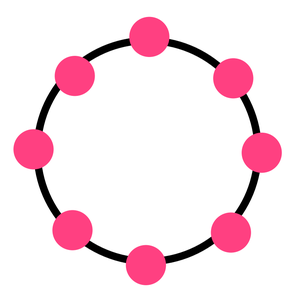
Rack

A physical grouping of nodes in a Hadoop cluster that share the same network switch.



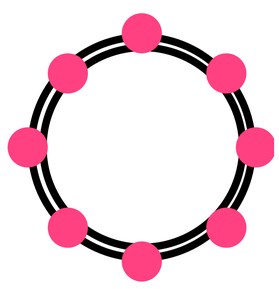
Name Node

The master node in HDFS that manages the file system namespace and metadata for all files in the file system.



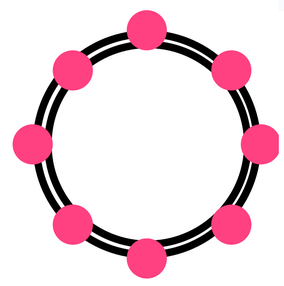
Secondary Name Node

A helper node for the NameNode in HDFS that maintains a copy of the file system metadata.



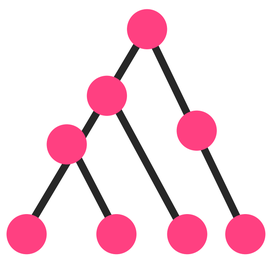
Data Node

A helper node for the NameNode in HDFS that maintains a copy of the file system metadata.



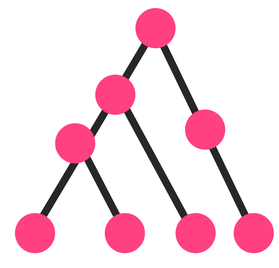
Job tracker

The service in MapReduce that manages the processing of jobs in a Hadoop cluster.



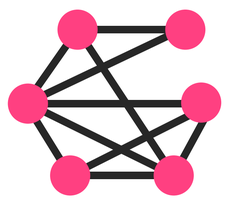
Task Tracker

The service in MapReduce that manages the processing of individual tasks within a job.



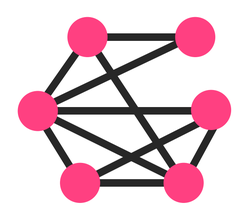
Input Split

The division of an input dataset into smaller, separate units for processing in MapReduce.



Partition

The division of data in a MapReduce job into separate, manageable chunks for processing.



Reduce

The operation in MapReduce that aggregates the output from the map phase into a single, reduced output.

Modules Used in the Hadoop Ecosystem

HDFS (Hadoop Distributed File System)

A distributed file system that stores data across multiple nodes in a Hadoop cluster.

MapReduce

A programming model for processing large data sets that is the foundation of Hadoop processing.

YARN (Yet Another Resource Negotiator)

A resource management system in Hadoop that manages resources for the processing of data in a Hadoop cluster.

Hive

A data warehousing and SQL-like query language for Hadoop.

Pig

A high-level platform for creating MapReduce programs used with Hadoop.

HBase

A NoSQL database for Hadoop that provides real-time access to data.

Spark

An open-source, big data processing engine for use in Hadoop clusters.

Mahout

A machine learning library for Hadoop.

Zookeeper

A coordination service for distributed systems that is used in Hadoop to coordinate actions between nodes.

Flume

A service for collecting, aggregating, and moving large amounts of log data into Hadoop.

Sqoop

A tool for transferring data between Hadoop and structured data stores such as relational databases.

Oozie

A workflow management system for Hadoop that coordinates and manages Hadoop jobs.

Ambari

A web-based tool for managing and monitoring Hadoop clusters.

Avro

A data serialization system for Hadoop that provides a compact binary format for data and schema.

Parquet

A columnar storage format for Hadoop that provides efficient storage and processing for big data.