— Hadoop —

1.1 Overview

Apache Hadoop is a collection of open-source software utilities that facilitate using a network of many computers to solve problems involving massive amounts of data and computation. It provides a SOFTWARE FRAMEWORK for DISTRIBUTED STORAGE and processing of BIG DATA using the MAP-REDUCE PROGRAMMING MODEL.

Apache Hadoop framework is composed of the following modules:
1. **Hadoop common**
libraries and utilities needed by other Hadoop modules
1. **HDFS**
a distributed file system that stores data on commodity machines providing very high aggregate bandwidth across the cluster. 1. **Hadoop Yarn (2012)**
a platform responsible for managing computing resources in clusters and using them for scheduling users' applications 1. **Hadoop MapReduce**
an implementation of the MapReduce programming model for large scale data processing.

Hadoop refers to the modules, sub-modules and also the ecosystem. There are collection of additional software packages that can be installed on top of or alongside Hadoop. Ex - Pig *(scripting language like SQL)*, Hive *(data warehouse)*, HBase *(database)*, Spark *(cluster computing framework)*, ZooKeeper, etc.

Difference between Hadoop 1 and 2

Hadoop 1	Hadoop 2
Map-reduce Engine	YARN (Yet Another Resource Negotiator) Runs 2 daemons **- Resource manager:** does job tracking and resource allocation to applications**- Application master:** which monitors progress of the execution.

Difference between Hadoop 2 and 3

Hadoop 2	Hadoop 3
Single name node only	Enables having multiple namenodes - solves the problem of single point of failure

Hadoop 2	Hadoop 3
	Containers work in principle of docker - reduces application development time Decreases storage overhead using *erasure coding* Permits usage of GPU hardware within the cluster - beneficial for executing deep learning algorithms on a Hadoop cluster

1.2 Software Framework

Software framework: A software framework is an abstraction in which software providing generic functionality can be selectively changed by additional user-written code, thus providing application-specific software. It provides a standard way to build and deploy applications and is a unviersal, reusable software environment that provides particular functionality as part of a larger software platform to facilitate development of software applications, products and solutions.

> "A Software framework is an applicatio-specific software construction material. It gives the developer more than just the bare bones; it also provides you with the (software) platform and the runtime environment."

Key distinguishing features from libraries

1. **Inversion of control:** Overall program control flow is determined by framework, and not caller (user program).
2. **Non-modifiable framework code:** In general, a software framework is not supposed to modified. Should only accpet user-implemented extensions.
3. **Extensibility:** Users can extend the framework - by selective overriding, or by adding specialised code.

Further definitions

- **Software environment** *(AKA runtime environment)*
- **Software platform** *(AKA computing platform)* **:** It is the environment in which a piece of software is executed. It may be the hardware, the OS, even a web browser and associated APIs, or the other underlying software as long as the program code is executed by it. They have differnet abstraction levels:
  - Computer Architecture

    - OS
    - Runtime libraries
- **Runtime librarires:** Runtime libraries are a set of low-level routines used by a compiler to invoke some of the behaviours of a runtime environment, by inserting calls to the runtime library into compiled executable binary.

1.3 Clustered File System:

A clustered filesystem is a filesystem which is being shared by being simultaneously mounted on multiple servers. It's main features include: 1. location independent addressing 1. redundancy

Clustered file systems are of the following 2 types: 1. Shared File Systems 1. Distibuted File Systems

Shared File System

A shared file system uses a 'Storage Area Network' to allow multiple computers to gain direct access at the block level (a set of bits/bytes *OR* a whole number of records). Access control and translation, from file level operations that applications use to block-level operations used by the SAN, must take place on the client node.

Distributed File System (DFS)

DFSs does not share block level access to the same storage but uses a network protocol. These are also known as network file systems.

Note: A DFS and a Distributed Data Store aren't the same.

Distributed File Systems	Distributed Data Store
Allows files to be accessed using the same interfaces and semantics as local files. Ex: (un)mounting, listing directories, read/write at byte boundaries, systems native permission model	Distributed data store requires using a different API or library and have different semantics (most often those of a database)

Design goals - Transparency

Distributed File Systems aim to be 'invisible' to client programs and essentially make them see a system similar to a local file system. Behind the scenes, the distributed file system handles locating files, transporting data, and potentially providing the other features listed below.

1. **Access transparency** Clients are unaware that files are distributed. They

can access the files in the same way local files are accessed.

2. **Location transparency** A consistent namespace exists encompassing local as well as remote files. The name of a file does not give its location.
3. **Concurrency transparency** All clients have the same view of the state of the file system. Modifications are seen by all clients in a coherent manner.
4. **Failure transparency** Client and client programs should operate correctly after a server failure.
5. **Replication transaparency** Clients should be unaware of the file replication performed across multiple servers to support scalability.
6. **Migration transparency** Files should be able to move between different servers without the clients knowledge.
7. **Heterogenity** File service should be provided across different platforms (hardware and OS)
8. **Scalability** Should work well in small and large environments

1.4 Big data

Big data is a field that deals with processing and analysing large datasets. Associated with 4 V's: - **Volume** Large amounts of data - **Variety** Images, video, audio, text (both structured and unstructured) and so on… - **Velocity** Data enters into the system rapidly - **Veracity** How clean the data is and how to clean the data

Challenges include capturing data, data storage, data analysis, searching, sharing, transfer, visualisation, querying, updating, information privacy and data source.

Important Note: The major problem faced in the domain of big data is structuring the data pipeline - how do we identify the data, - how do we bring the data into the system - where do we store the data - how do we handle the problem of missing data - how are we going to merge data from different input streams

These aspects of Big Data are as crucial as the machine learning algorithm in a deep learning software because this collected data is what is fed into the machine learning algorithm.

(courtesy: KV sir)

1.5 MapReduce Programming Model

Definitions

Let us start with breaking down what each of these words mean.

Programming model

A programming model refers to the style of programming where execution is invoked by making what appears to be library calls. It's ***execution model*** is different from that of the base language in which the code is written.

Technically, a programming model is an abstraction of the underlying computer system that allows for the exprression of both algorithms and data structures. In comparison, programming languages and APIs provide implementations of these abstractions and allow the algorithms and data structures to be put into practice.

That also means that the model exists independently of the choice of both the language and supporting APIs.

Execution model

A programming language consists of its syntax, plus the execution model. It specifies the behaviour of the elements of the language. By applying it, one can derive the behaviour of the program.

It covers things such as what is an indivisible unit of work, what are the constraints on the order in which those units of work take place.

Example: The C Language - is made up 'statements' - a chunk of syntax terminated by ';' - language specs say program execution proceeds statement by statement. In other words, statement = indivisible unit of work and execution order is one after the other, further determined by IF and WHILE statements. - execution order in the statement itself is determined by 'precedence'.

MapReduce

MapReduce is a programming model and an associated implementation for processing and generating 'big data' sets with a parallel, distributed algorithm on a cluster. 1. **Map:** performs filtering and sorting 1. **Reduce:** sperforms the summary operation

Computer Cluster

Set of loosely or tightly connected computers that work together and can be viewed as a single system. Each node (a computer OR a server) runs its own instance of an OS which is connected through fast local area networks. The hardware and the OS does not necesarily need to be identical across the cluster.