**Assignment 8.4**

**Explain the difference between FIFO and Capacity scheduler:**

**FIFO:**

FIFO stands for first in first out. In FIFO scheduling, Job Tracker pulls oldest job first from job queue. It doesn't consider about priority or size of the job. Hadoop’s built-in scheduler runs jobs in FIFO order.

Scheduler scans through jobs in order of priority when a task slot becomes free. Then it picks the map task in the job with data closest to the slave.

**Capacity Scheduler:**

The CapacityScheduler is designed to allow sharing a large cluster while giving each organization a minimum capacity guarantee. The central idea is that the available resources in the Hadoop Map-Reduce cluster are partitioned among multiple organizations who collectively fund the cluster based on computing needs.

There is an added benefit that an organization can access any excess capacity no being used by others. This provides elasticity for the organizations in a cost-effective manner.

**Explain the difference between FIFO and Fair scheduler:**

**FIFO:**

FIFO stands for first in first out. In FIFO scheduling, Job Tracker pulls oldest job first from job queue. It doesn't consider about priority or size of the job. Hadoop’s built-in scheduler runs jobs in FIFO order.

Scheduler scans through jobs in order of priority when a task slot becomes free. Then it picks the map task in the job with data closest to the slave.

**Fair Scheduler:**

Fair scheduling is a method of assigning resources to jobs such that all jobs get, on average, an equal share of resources over time. When there is a single job running, that job uses the entire cluster. When other jobs are submitted, tasks slots that free up are assigned to the new jobs, so that each job gets roughly the same amount of CPU time.

**Explain the difference between Capacity and Fair scheduler:**

**Fair Scheduler:**

Fair scheduling is a method of assigning resources to jobs such that all jobs get, on average, an equal share of resources over time. When there is a single job running, that job uses the entire cluster. When other jobs are submitted, tasks slots that free up are assigned to the new jobs, so that each job gets roughly the same amount of CPU time.

**Capacity Scheduler:**

The CapacityScheduler is designed to allow sharing a large cluster while giving each organization a minimum capacity guarantee. The central idea is that the available resources in the Hadoop Map-Reduce cluster are partitioned among multiple organizations who collectively fund the cluster based on computing needs. There is an added benefit that an organization can access any excess capacity no being used by others. This provides elasticity for the organizations in a cost-effective manner.

**What are the limitations of Hadoop 1.x and how they were overcome in Hadoop 2.x:**

Hadoop 1.x has many limitations or drawbacks. Main drawback of Hadoop 1.x is that MapReduce Component in its Architecture. That means it supports only MapReduce-based Batch/Data Processing Applications.

**Hadoop 1.x has the following Limitations/Drawbacks:**

* It is only suitable for Batch Processing of Huge amount of Data, which is already in Hadoop System.
* It is not suitable for Real-time Data Processing.
* It is not suitable for Data Streaming.
* It supports upto 4000 Nodes per Cluster.
* It has a single component: JobTracker to perform many activities like Resource Management, Job Scheduling, Job Monitoring, Re-scheduling Jobs etc.
* JobTracker is the single point of failure.
* It does not support Multi-tenancy Support.
* It supports only one Name Node and One Namespace per Cluster.
* It does not support Horizontal Scalability.
* It runs only Map/Reduce jobs.
* It follows Slots concept in HDFS to allocate Resources (Memory, RAM, CPU). It has static Map and Reduce Slots. That means once it assigns resources to Map/Reduce jobs, it cannot re-use them even though some slots are idle.

**Hadoop 2.x has resolved most of the Hadoop 1.x limitations by using new architecture.**

* By decoupling MapReduce component responsibilities into different components.
* By introducing new YARN component for Resource management.
* By decoupling component’s responsibilities, it supports multiple namespace, Multi-tenancy, Higher Availability and Higher Scalability