**#Assignment\_8.4**

**Que.1) Explain the difference between FIFO, Capacity scheduler and fair Scheduler.**

**FIFO Scheduler:-**

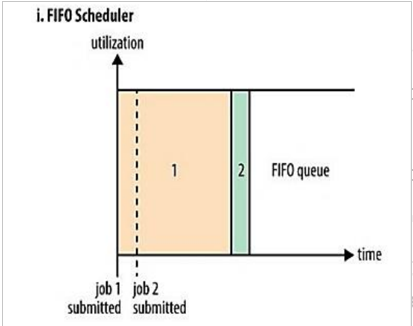
FIFO scheduler places applications in a queue and run them in order of their submission. The First submission will be preferred firstly and likewise for henceforth submissions.

First preference in queue is allocated a space first, once the task of that application is terminated then the second application in queue is served and likewise.

The method of scheduling is very easy to understand and no configuration is needed.

The scheduling is not suitable for shared clusters, as multiple tasks are going on in a clusters hence it could be a problem.

The remedy to it is to use Fair scheduler or capacity scheduler.

****

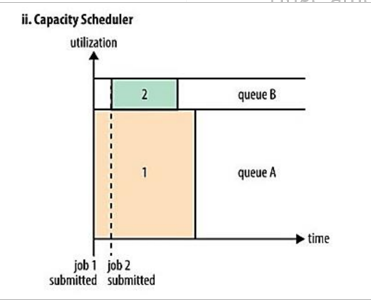
**Capacity Scheduler:-**

Capacity Scheduler, we define number of named queues. Every queue has a configurable number for map and reduce slots. When it contains any job, capacity is provided to it. Within queue FIFO priority is used but you can place limit.

If queues are not properly used some queue might get overloaded with more traffic while some will left unused.

Cluster utilization is there but the queue capacity is reserved for the jobs that are in queue.

Large file take long time to finish, while smaller jobs are done quickly as compared to FIFO.

****

**Fair Scheduler:-**

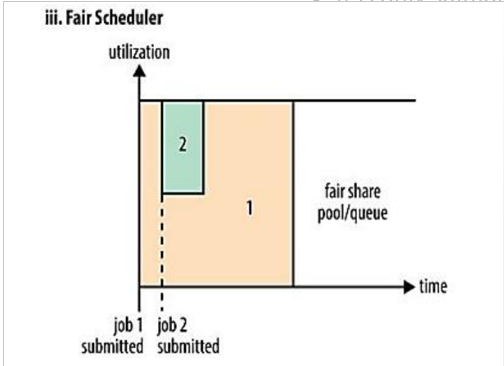
The fair scheduler was arose out when Facebook thought of its need to share its data warehouse with multiple users.

It basically works on the principle of “pools”.

The jobs to be done are placed into pool based on the configuration, each pool has guarantee capacity which we specify through the configuration file. It provides minimum map and reduce slots. If slots are empty any other job can take that slot.

So, small jobs finish quickly and large jobs do not have to wait long. If small job is complete and need no resource then large job take complete cluster capacity to perform its job.

Higher Cluster utilization is there and small jobs are completely on time.

****

**Que.2) what are the limitations of Hadoop 1.x and how they were overcome in Hadoop 2.x.**

Hadoop 1.x has various limitations that are later improvised in Hadoop2.x. Few **Limitations are mentioned below**-

1. It is only suitable for the huge data present on HDFS using batch processing.
2. It is not suitable for the data streaming and real time data processing.
3. It don not support multi-tenancy for multiple cluster access. It supports single name node and namespace per cluster.
4. The limit on number of nodes is limited up to only **4000.**
5. The block size where data is saved in chunks is only **64MB.**
6. It is suitable only to run Map and reduce jobs.
7. Do not support horizontal scaling.
8. Concept of slots is followed.
9. In case of job failure Admin has to explicitly come and resume or restart the job.

**Features improved in Hadoop2.x-**

1. Concept of containers is introduced rather than using slots.
2. **Resource management** is introduced in yarn. MapReduce responsibilities are decoupled into various components.
3. **Decoupling** helped in achieving multiple namespace, high availability, high scalability and multi tenancy.
4. The block size is levelled up to **128Mb.**
5. The limit on namenode is levelled up to 40000 namenodes.
6. Admin do not need to come in case of failure as passive and secondary namenode will be up in case of failure.

This is how limitations of hadoop1.x were overcome in hadoop2.x.