How Hadoop MapReduce Job Works

They are **four** independent entities need for run a job

1. The client ,which submit mapreduce job
2. The **jobtracker**,which **coordinates of the job** run
3. The **tasktracker** ,which **run the task**
4. The HDFS,which is used for sharing job files between other entities

**Job Submission:**

The job submission process implemented by JobSubmmitter does following

1. The Job asks **jobtracker** for a new **jobid**(**step2**).
2. Checks the output specification ,which means whether output directory already exist or not. If o/p directory already exist,the job not submitted and an error thrown to the mapreduce program.
3. If input split can’t computed (input path doesn’t exist),then  the job not submitted and an error thrown to the mapreduce program.
4. copies the what are resources are needed to run the job (including the job jar file,configuration file,computed input splits)(**step3**).
5. Finally ,tells to the jobtracker that is ready for execution by calling **submitjob()** on jobtracker.(step4)

**Job Initialization:**

1. when the **Jobtracker** receives a call to **submitjob()** method,it puts into internal queue from where the job scheduler will pick up and initialize it(initialize involves creating an object to represent the job being run).(**step5**)
2. Create list of task to run,the job scheduler first retrieves input split,(**step6**)then it creates one map task for each split.The number of reducer task to create is determined by the mapred.reduce.task property of job which is set by setNumReducerTasks().
3. Two further task are also created.They are :
   1. **job setup task**( to setup job before map task and also create the final output directory for the job and temporary working space for task output)
   2. **job cleanup task** ( to clean up after all reduce task are completed, and also delete the temporary working space for the task output)

These two tasks are run by tasktracker.

**Task Assignment:**

1. **Tasktracker** run a simple loop that periodically send a **heartbeat** to the jobtracker.Heatbeat tell the jobtracker that task tracker is alive(**step7**)
2. Apart of heartbeat,tasktracker will indicate whether it is ready to run a new task and send a status of tasktracker.
3. Tasktracker have fixed number of slots for map tasks and for reduce task.we can configure the slots for map task and reduce ( which is depends onthe number of cores and amount of memory on the tasktracker)
4. So,if the tasktracker has atleast one empty map task slot,the jobtracker will select a maptask,otherwise,it wil select a reduce task.

**Task Execution:**

1. Tasktracker has been assigned a task by jobtracker,the next step is for it to run the task.
2. It localizes the job jar by copying it from the shared filesystem to tasktracker filesystem.(**step8**)
3. It creates local working directory for the task.
4. It creates an instance of TaskRunner to run the task
5. TaskRunner launces a new JVM(Java Virtual Machine)(**step9**) to run each task(**step10**)

