ASSINGMENT -2

What is input split and how to configure mappers based on need?

InputSpli**t** is the logical representation of data. Hadoop framework divides InputSpli**t** into records. Then mapper will process each record. Size of split is approximately equal to HDFS Block size (128 MB). In MapReduce program, Inputsplit is user defined.

Mappers are configured on the basis of the number of inputsplits that the framework is been dived into.

Hadoop configuration.

Configurations are specified by resources. A resource contains a set of name/value pairs as XML data. Each resource is named by either a String or by a path. If named by a String, then the classpath is examined for a file with that name. If named by a Path, then the local filesystem is examined directly, without referring to the classpath.

The configurations in Hadoop are based of two types.

1. Read-only default configuration.

2. Site-specific configuration.

One of the important steps in site-configuration is configuring the environment of the Hadoop demons. In this the Administrators should use the conf/hadoop-env.sh script to do site-specific customization of the Hadoop daemons' process environment. At the very least you should specify the JAVA\_HOME so that it is correctly defined on each remote node. In most cases you should also specify HADOOP\_PID\_DIR to point a directory that can only be written to by the users that are going to run the hadoop daemons. Otherwise there is the potential for a symlink attack.

Block in HDFS

When you store a file in HDFS, the system breaks it down into a set of individual blocks and stores these blocks in various slave nodes in the Hadoop cluster. Blocks are large. They default to 64 megabytes each and most systems run with block sizes of 128 megabytes or larger.

HDFS.

The Hadoop Distributed File System (HDFS) is designed to store very large data sets reliably, and to stream those data sets at high bandwidth to user applications. In a large cluster, thousands of servers both host directly attached storage and execute user application tasks

MapReduce.

Hadoop MapReduce is a software framework for easily writing applications which process vast amounts of data in-parallel on large clusters of commodity hardware in a reliable, fault-tolerant manner.

Different stages of map reduce.

The two main stages of MapReduce are,

1. Map phase.
2. Reduce phase.

Map phase: It consist of different tasks such as, 1. Map Task launch, 2. Map task: INIT, 3. Map task execution, 4. Map task spilling, 5. Map task shuffle.

Reduce phase comprises of YARN and Map reduce interaction

Shuffle in MapReduce.

The process of transferring data from the mappers to reduce is known as shuffling, the process by which the system performs the sort and transfers the map output to the reducer as input. So, MapReduce shuffle phase is necessary for the reducers, otherwise, they would not have any input. As shuffling can start even before the map phase has finished so this saves some time and completes the tasks in lesser time.

where is mapper output written?

The output of the mapper is written in the local disk.

What is a combiner?

A Combiner, also known as a semi-reducer, it is an optional class that operates by accepting the inputs from the Map class and thereafter passing the output key-value pairs to the Reducer class. The main function of a Combiner is to summarize the map output records with the same key.

What is a Partitioner?

A partitioner partitions the key-value pairs of intermediate Map-outputs. It partitions the data using a user-defined condition, which works like a hash function. The total number of partitions is same as the number of Reducer tasks for the job.

What is Github and their commands?

In the word github, git is version control system. Where developers create Something and they make constant changes to the code, releasing new versions up to and after first official release. Version control keeps these revisions straight , storing the modifications in a central respiratory system which allows developers to easily collaborate . Hub is a place all developers store their projects and network with like-minded people.

Git hub commands.

Commit all edited files and add a message

git commit -a -m "My commit"

* Add all new files

git add .

* Perform a pull operation

git pull REMOTENAME BRANCHNAME

* Perform a push operation

git push REMOTENAME BRANCHNAME

* Prune all stale remote tracking branches

git remote prune REMOTENAME

* Create a branch

git branch BRANCHNAME

* View branches

git branch

* Checkout a different branch

git checkout BRANCHNAME

* Checkout a remote branch

git checkout -b LOCALBRANCHNAME origin/REMOTEBRANCHNAME

* Merge the changes made in another branch in to the current branch

git merge BRANCHNAME

* Delete a local branch

git branch -d BRANCHNAME

* Delete a remote branch

git push origin :BRANCHNAME

* Delete a remote branch (sexier syntax)

git push origin --delete BRANCHNAME

* Scrap uncommitted state and return the working tree to the last committed state

git reset --hard HEAD

* Delete the latest commit, and return to the one previous (one before HEAD)

git reset --hard HEAD~1

* Return a single file to it's last committed state

git checkout -- FILENAME

git checkout HEAD FILENAME

* Git log

git log

git log --pretty=oneline

git log --pretty=short

* Cherry pick commits and apply them to another branch (first grab the commit ID from the branch with said commit, then checkout the branch you wish to apply the commit to)

git cherry-pick COMMIT-ID

* Stash uncommitted changes

git stash save "message"

* Apply stashed changes somewhere

git stash apply

* Stop a file being tracked (but do not delete it from the working directory, add to .gitignore etc after this)

git rm --cached <file/folder>

* Restore a file to a previous commit

git checkout <commitID> <file/to/restore>

* Restore a file to one before a commit (say you know the commitID where something went wrong, and want one before that point)

git checkout <commitID>~1 <file/to/restore>