This document will talk about how to use hadoop in CCR interactively and use single node for debugging MapReduce program.

Follow the steps:

1. Apply for nodes:
   fisbatch --partition=debug --nodes=1 --ntasks-per-node=2 --time=01:00:00 --exclusive
2. Assign environment variables
   export | grep SLURM
3. Load modules
   module load java/1.6.0_22
   module load hadoop/2.5.1
   module load myhadoop/0.30b
4. Set more environmental variables
   export MH_SCRATCH_DIR=/scratch/$SLURM_JOBID
   export HADOOP_CONF_DIR=$SLURM_SUBMIT_DIR/config-$SLURM_JOBID
   NPROCS=`srun --nodes=${SLURM_NNODES} bash -c 'hostname' |wc -l`
5. Start hadoop
   $MH_HOME/bin/myhadoop-configure.sh
   $HADOOP_HOME/sbin/start-all.sh
   $HADOOP_HOME/bin/hadoop dfsadmin -report.
6. Run and debug program
   - Make hdfs directory
     $HADOOP_HOME/bin/hdfs --config $HADOOP_CONF_DIR dfs -mkdir /data
   - Copy files/directory to hdfs
     $HADOOP_HOME/bin/hdfs --config $HADOOP_CONF_DIR dfs -put ./WordCount.txt /data/
   - Submit job
     $HADOOP_HOME/bin/hadoop --config $HADOOP_CONF_DIR jar WordCount.jar WordCount /data wordcount-output
   - If there's error, edit the files, compile, export jar file and resubmit
     #compile
     # make sure you have made directories /bin /lib /src before compilation and all the files needed are present in these directories.

     javac -cp "lib/hadoop-common-2.5.1.jar:lib/hadoop-mapreduce-client-core-2.5.1.jar" -d bin src/*.java

     # exporting jar file
     # make sure you have Manifest.txt file made, in which Main-class and Class-path are present.
     # Manifest.txt has format such as

```
Main-Class: Main
Class-Path: lib/*.jar
```

jar -cvmf Manifest.txt hw.jar -C bin . -C src .

7. List and get output
   $HADOOP_HOME/bin/hdfs --config $HADOOP_CONF_DIR dfs -ls wordcount-output
   $HADOOP_HOME/bin/hdfs --config $HADOOP_CONF_DIR dfs -get wordcount-output ./myoutput-$SLURM_JOBID
8. Stop hadoop
   $HADOOP_HOME/sbin/stop-all.sh
   $MH_HOME/bin/myhadoop-cleanup.sh