CS5531: Programming Project (Fall 2012)

Praveen Rao
Computer Science Electrical Engineering, UMKC
Email: `raopr@umkc.edu`

October 27, 2012

The goal of this project is to design, implement, and evaluate a new scheduling algorithm for Big Data processing using Apache Hadoop. This project will be done in groups of two. I encourage you to actively discuss with your peers the issues you may have w.r.t Linux and the project itself.

For this project, you will use IBM SmartCloud Enterprise. You can login via https://www-147.ibm.com/cloud/enterprise/dashboard. I will provide you instructions to use this cloud.

Here is a useful link to learn Linux commands: http://www.ibm.com/developerworks/training/kp/l-kp-command/index.html.

There are three phases in this project and they are worth 8%, 16%, and 8% of the final grade, respectively. This weightage is assigned based on the amount of effort required and the complexity of the phase. Project presentation constitutes 3% of your grade. **In all, this project is worth 35% of your final grade. All the best!**

All phases are due at 11.59 pm on the due date. You will turnin your code/reports via Blackboard.

Note: A regrade is possible for Phase 2, which involves coding. You will be allowed to fix your code within one week after I grade it. However, this will result in a 25% penalty for each test case that failed before.

1 Phase I (Download/Install/Test Apache Hadoop)

(Weightage 8%) **Due: Sep 14, 2012**

Complete the following tasks on the IBM cloud.

- You should be able to login into the IBM cloud portal. Start 1 master node (or name node) and 5 data nodes on the IBM cloud. Go through the videos listed under "Support" in your IBM account for instructions on how to create instances and login into them using SSH. Here are useful instructions from the cloud portal.

  ```
  Creating Your Cluster
  ```

```
---------------------

You should first launch a Master Node (Image: IBM BigInsights Basic
 1.1 - Hadoop Master Node PAYG). Once startup is fully complete, you
 can copy the public IP address for your Master Node and launch
 additional Data Nodes (Image: IBM BigInsights Basic 1.1 - Hadoop Data
 Node PAYG). The Data Nodes will add themselves to the cluster as they
 come up.


Connect to Your Instance
-----------------------

You can SSH to our instance using the standard cloud idcuser account
 with your SSH key. Putty would be an example of an SSH client you
 could use on Windows. The login name is idcuser. Once you are
 connected, you should be able to issue start-all.sh and stop-all.sh
 commands as needed. BigInsights is installed under /mnt/biginsights.
```

- If you click the master node listed under "Control Panel," then you will see the following near the bottom of the page:

  ```
  Web Consoles on Your Instance

  BigInsights Web Console
  NameNode Status
  JobTracker Status
  TaskTracker Status
  HBase Master UI
  HBase Region Server UI
  Hive Web Interface
  ```

  Click on BigInsights Web Console. Select Hadoop as the component and take a screenshot. You should have 6 healthy nodes. `[TURNIN]`

- Read through the documentation in http://hadoop.apache.org/common/docs/r0.20.2/hdfs_shell.html. Create a directory "cs5531" under /user/idcuser. Then copy the file `testinput.dat` given on Blackboard into /user/idcuser/cs5531.

- Run the Word Count example provided in /mnt/biginsights/opt/ibm/biginsights/IHC/hadoop-0.20.2-examples.jar. Use /user/idcuser/cs5531/testinput.dat as the input. Store the output in /user/idcuser/cs5531/output. Collect the output and store it in a single local file. `[TURNIN]`

Complete the following tasks on your personal machine.

- Download hadoop-0.20.2.tar.gz from ftp://apache.mirrors.pair.com/hadoop/common/hadoop-0.20.2/.

- Install Eclipse (or any other IDE) on your computer. Create a new Java project and import hadoop-0.20.2.tar.gz as an archive file. Make sure there are no errors during the build phase.
- Locate the source code for Fair Scheduler inside src/contrib/fairscheduler. Read through the sources files and understand the big picture.

  Read through http://hadoop.apache.org/common/docs/r0.20.2/fair_scheduler.html.
- Answer the following questions `[TURNIN]`:
  1. How can you enable Hadoop to use the Fair Scheduler for running jobs?
  2. How can you revert to FIFO scheduling within the Fair Scheduler?

1.1 What to turnin?

Submit all the items that are marked `[TURNIN]`.

2 Phase II

(Weightage 16%) **Due: ~~Oct 26~~ ~~Nov 2~~ Nov 9, 2012**

1. You will be implementing three scheduling algorithms in this phase.
   - Scheduling algorithm 1

     Requirement: the job with least number of (map + reduce tasks) should go first; break ties by first examining the priority and then the start time
   - Scheduling algorithm 2

     Requirement: the job with the smallest percentage of map tasks completed should go first; break ties by using Scheduling Algorithm 1
   - Scheduling algorithm 3

     Requirement: the job with the highest number of pending tasks should go first; break ties by using Scheduling Algorithm 2
2. You can create a bash script to automate the invocation of jobs with different priorities, maps, and reduces.

```
#!/bin/bash

if [ $# -ne 1 ];
then
  echo "Usage: run_jobs <# of jobs>"
  exit
fi

hadoop fs -rmr /user/idcuser/output*

# change the number of entries depending on the number of jobs
priorityarray=( NORMAL NORMAL LOW VERY_LOW HIGH HIGH NORMAL )

for ((i=0;i<${1};i++))
do
```

```
    # Assuming 20 maps and 5 reduces
    # you can change the priority
    echo "Jobid: " ${i} " Priority: " ${priorityarray[i]}
    hadoop jar /home/idcuser/WordCount.jar org.apache.hadoop.examples.WordCount \
               /user/idcuser/input /user/idcuser/output${i} 20 5 ${priorityarray[i]} &
done
```

3. You can modify WordCount.java as follows:

```
 public static void main(String[] args) throws Exception {
    Configuration conf = new Configuration();
    String[] otherArgs = new GenericOptionsParser(conf, args).getRemainingArgs();
    if (otherArgs.length != 5) {
      System.err.println("Usage: wordcount <in> <out> <# of map tasks> <# of reduce tasks>" \
                        " <VERY_HIGH|HIGH|NORMAL|LOW|VERY_LOW>");
      System.exit(2);
    }

    conf.set("mapred.map.tasks", otherArgs[2]);
    conf.set("mapred.reduce.tasks", otherArgs[3]);
    conf.set("mapred.job.priority", otherArgs[4]);

    Job job = new Job(conf, "word count");
    job.setJarByClass(WordCount.class);
    job.setMapperClass(TokenizerMapper.class);
    job.setCombinerClass(IntSumReducer.class);
    job.setReducerClass(IntSumReducer.class);
    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(IntWritable.class);
    FileInputFormat.addInputPath(job, new Path(otherArgs[0]));
    FileOutputFormat.setOutputPath(job, new Path(otherArgs[1]));


    System.exit(job.waitForCompletion(true) ? 0 : 1);
  }
}
```

4. You can create a Makefile to compile the WordCount program.

```
HADOOP_HOME=/home/rpraveen/Downloads/hadoop-0.20.2
all:
mkdir -p WordCount_classes
        javac -classpath ${HADOOP_HOME}/hadoop-0.20.2-core.jar:${HADOOP_HOME}/lib/commons-cli-1.2.jar\
                -d WordCount_classes WordCount.java
        jar -cvf WordCount.jar -C WordCount_classes/ .
```

What to turnin?

1. Your entire source code in tar.gz so that it can be imported as a new project in Eclipse.
2. A 2 page report that describes: (a) your design, (b) your test cases, (c) the challenges your faced, and (d) contributions of each team member.

3 Phase III

(Weightage 8%)

Due Date: November 16, 2012

What to turnin?

4 Project Presentation

(Weightage: 3%)

Each team will present their work and results in class on Dec 12, 2012.