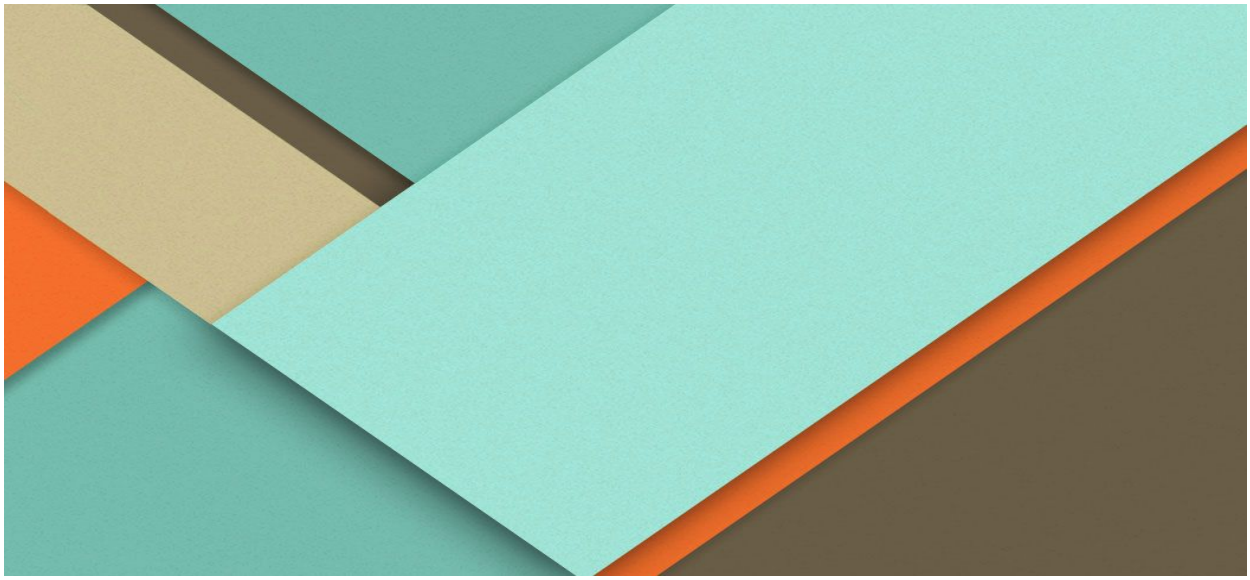

Cloud Computing

Map Reduce

24.10.2017

Submitted by: [GROUP 15]

Aditi (2014CS10205)
Aditi Singla (2014CS50277)
Aditi Gupta (2017BSY7508)

Installation of MapReduce Framework

The name node was using the default temp directory, which gets cleared on every reboot. Hence, we had to change the hadoop tmp directory so as to function map reduce properly with the name node.

1. To get the namenode up and running, add the following to the core-site.xml at all nodes.

   ```
   <property>
   <name>hadoop.tmp.dir</name>
   <value>/home/hadoopuser/hadoop_tmp</value>
   </property>
   ```

2. Run the following commands again on all the nodes. (Setting permissions for the tmp folder)
   a. $cd /home/hadoopuser
   b. $mkdir hadoop_tmp
   c. $sudo chown hadoopuser:hadoopgroup /home/hadoopuser/hadoop_tmp
   d. $sudo chown 777 -R /home/hadoopuser/hadoop_tmp
3. On the namenode, on running start-all.sh, if we get binding error, run the following commands on namenode.
   a. $sudo netstat -tulpn | grep :50070 (check any processes running in port 50070, if there are any the / will appear at the RHS of the output).
   b. $sudo kill -9 <process_id> (kill the process)
4. After this, start-all.sh should run fine. But, if we observe datanodes not running (only jps and task tracker running), run the following commands on each of the datanodes.
   a. $cd /opt/hadoop/hadoop/bin
   b. $hadoop datanode
   c. If we find an error saying difference in namespace IDs, format the master, get the latest namenode namespace ID and update the namespaceID in /opt/hadoop/hadoop/dfs/name/data/current/VERSION , to the one of master, for all data nodes.
   d. $~/bin/hadoop datanode (on all datanodes)
   e. Once the above command finishes on each of the datanode, run the following on the namenode (master).
   f. $/opt/hadoop/hadoop/bin stop-all.sh
   g. $/opt/hadoop/hadoop/bin start-all.sh
5. We don't need to run separate commands to install MapReduce in the older versions of Hadoop, since during the installation of Hadoop, we had made the

needed modifications in mapred-site.xml. So, just to check the proper installation of MapReduce, run the following commands on namenode.

This map reduce task was a small task that took as input the average consumption of a house every month for 5 years, and returned the max average.

a. $mkdir units
b. $javac -classpath hadoop-core-1.2.1.jar -d units ProcessUnits.java
c. $jar -cvf units.jar -C units/ .
d. $HADOOP_HOME/bin/hadoop fs -mkdir input_dir
e. $HADOOP_HOME/bin/hadoop fs -put /home/hadoop/sample.txt input_dir
f. $HADOOP_HOME/bin/hadoop fs -ls input_dir/
g. $HADOOP_HOME/bin/hadoop jar units.jar hadoop.ProcessUnits input_dir output_dir
h. If this command gives an error saying PriviledgedActionException, run the following commands.
i. export HADOOP_USER_NAME=hadoopuser.
j. $HADOOP_HOME/bin/hadoop fs -chmod -R 777 /tmp/hadoop-root/mapred/

If everything is fine by this step, MapReduce framework has been installed properly.

WordCount for Large Text Files

1. The code for wordcount has been written in **WordCount.java** and converted to a jar file.
2. Few datasets were obtained online, randomly shuffled, concatenated, and then replicated multiple times, using a python script, to get a large data file called **wc_mapred_data.txt**. (26.2 MB)
3. Run the following commands on namenode. The input file is sent to hadoop and then the jar file is run.
   a. $mkdir wordcount
   b. $javac -classpath hadoop-core-1.2.0.jar -d wordcount WordCount.java
   c. $jar -cvf wordcount.jar -C wordcount/ .
   d. $ $HADOOP_HOME/bin/hadoop fs -mkdir input_word_dir
   e. $ $HADOOP_HOME/bin/hadoop fs -put /home/hadoopuser/wc_mapred_data.txt input_word_dir
   f. $ $HADOOP_HOME/bin/hadoop fs -ls input_word_dir/
   g. $ $HADOOP_HOME/bin/hadoop jar wordcount.jar WordCount input_word_dir output_word_dir
   h. $ $HADOOP_HOME/bin/hadoop fs -ls output_word_dir
4. The job reported CPU time 10340 milli sec.

Experimentation

One of the data nodes (VM basically) was shut down while running the mapreduce for wordcount, for a 122 MB file, which is spread over two blocks. (since block size is 64 MB)

VM was shut down (rebooted actually) using the command **$sudo reboot.**

1. Reboot at **Map 19% Reduce 0%**

```
17/10/25 22:08:41 INFO mapred.JobClient: Running job: job_201709062117_0021
17/10/25 22:08:42 INFO mapred.JobClient:  map 0% reduce 0%
17/10/25 22:08:54 INFO mapred.JobClient:  map 19% reduce 0%
17/10/25 22:08:57 INFO mapred.JobClient:  map 27% reduce 0%
17/10/25 22:08:58 INFO mapred.JobClient:  map 35% reduce 0%
17/10/25 22:09:00 INFO mapred.JobClient:  map 43% reduce 0%
17/10/25 22:09:01 INFO mapred.JobClient:  map 51% reduce 0%
17/10/25 22:09:03 INFO mapred.JobClient:  map 58% reduce 0%
17/10/25 22:09:06 INFO mapred.JobClient:  map 66% reduce 0%
17/10/25 22:09:09 INFO mapred.JobClient:  map 74% reduce 0%
17/10/25 22:09:11 INFO mapred.JobClient:  map 76% reduce 0%
17/10/25 22:09:19 INFO mapred.JobClient:  map 76% reduce 16%
17/10/25 22:09:45 INFO mapred.JobClient: Task Id :
attempt_201709062117_0021_m_000001_1, Status : FAILED
Error: WordCount$TokenizerMapper : Unsupported major.minor version 51.0
17/10/25 22:09:46 WARN mapred.JobClient: Error reading task outputbaadaldesktopvm
17/10/25 22:09:46 WARN mapred.JobClient: Error reading task outputbaadaldesktopvm
17/10/25 22:10:01 INFO mapred.JobClient:  map 83% reduce 16%
17/10/25 22:10:05 INFO mapred.JobClient:  map 93% reduce 16%
17/10/25 22:10:08 INFO mapred.JobClient:  map 100% reduce 16%
17/10/25 22:10:13 INFO mapred.JobClient:  map 100% reduce 100%
17/10/25 22:10:15 INFO mapred.JobClient: Job complete: job_201709062117_0021
```

Observation: Since the VM was shut down quite early, the complete Map Reduce tasks were done on the other two VMs. Hence, there was no effect on the map reduce percentages. It takes around two minutes for the complete Map Reduce along with the rebooting. CPU Time of the command : 42650 milli sec.

2. Reboot at **Map 93% Reduce 0%**

```
17/10/25 22:30:34 INFO mapred.JobClient: Running job: job_201709062117_0023
17/10/25 22:30:35 INFO mapred.JobClient:  map 0% reduce 0%
17/10/25 22:30:40 INFO mapred.JobClient: Task Id :
attempt_201709062117_0023_m_000001_0, Status : FAILED
Error: WordCount$TokenizerMapper : Unsupported major.minor version 51.0
17/10/25 22:30:40 WARN mapred.JobClient: Error reading task outputbaadaldesktopvm
```

```
17/10/25 22:30:40 WARN mapred.JobClient: Error reading task outputbaadaldesktopvm
17/10/25 22:30:47 INFO mapred.JobClient:  map 9% reduce 0%
17/10/25 22:30:50 INFO mapred.JobClient:  map 32% reduce 0%
17/10/25 22:30:53 INFO mapred.JobClient:  map 51% reduce 0%
17/10/25 22:30:56 INFO mapred.JobClient:  map 70% reduce 0%
17/10/25 22:30:59 INFO mapred.JobClient:  map 89% reduce 0%
17/10/25 22:31:01 INFO mapred.JobClient:  map 93% reduce 0%
17/10/25 22:31:02 INFO mapred.JobClient:  map 100% reduce 0%
17/10/25 22:31:10 INFO mapred.JobClient:  map 100% reduce 16%
17/10/25 22:31:49 INFO mapred.JobClient: Task Id :
attempt_201709062117_0023_m_000001_1, Status : FAILED
Too many fetch-failures
17/10/25 22:31:49 WARN mapred.JobClient: Error reading task outputConnection refused
(Connection refused)
17/10/25 22:31:49 WARN mapred.JobClient: Error reading task outputConnection refused
(Connection refused)
17/10/25 22:31:50 INFO mapred.JobClient:  map 50% reduce 16%
17/10/25 22:31:58 INFO mapred.JobClient:  map 64% reduce 16%
17/10/25 22:32:01 INFO mapred.JobClient:  map 75% reduce 16%
17/10/25 22:32:04 INFO mapred.JobClient:  map 85% reduce 16%
17/10/25 22:32:07 INFO mapred.JobClient:  map 96% reduce 16%
17/10/25 22:32:10 INFO mapred.JobClient:  map 100% reduce 16%
17/10/25 22:32:18 INFO mapred.JobClient:  map 100% reduce 100%
17/10/25 22:32:20 INFO mapred.JobClient: Job complete: job_201709062117_0023
```

Observation: Since the VM was shut down when almost the complete Map was done, some part of Map output became unreadable (Connection refused error) and had to be carried out again, and hence the dip in the Map percentage. It takes around two minutes for the complete Map Reduce along with the rebooting. CPU Time of the command : 39200 milli sec.

Calculation of Average Grades for Class Records

1. The code for average grade calculation has been written in **GradeAverage.java** and converted to a jar file.
2. Python script has been written to randomly generate class records for 1000 students, in 10 courses and grades ranging from 4-10. For better execution of MapReduce, the dataset has been shuffled to randomly distribute the records of every student. The dataset is stored in a file named **grades.txt**.
3. Run the following commands on namenode. The input file is sent to hadoop and the jar file is run.
    a. $mkdir grades
    b. $javac -classpath hadoop-core-1.2.0.jar -d grades GradeAverage.java
    c. $jar -cvf grades.jar -C grades/ .

d. $HADOOP_HOME/bin/hadoop fs -mkdir input_grade_dir
e. $HADOOP_HOME/bin/hadoop fs -put /home/hadoopuser/grades.txt input_grade_dir
f. $HADOOP_HOME/bin/hadoop fs -ls input_grade_dir/
g. $HADOOP_HOME/bin/hadoop jar grades.jar GradeAverage input_grade_dir output_grade_dir
h. $HADOOP_HOME/bin/hadoop fs -ls output_grade_dir/
i. $HADOOP_HOME/bin/hadoop fs -cat output_grade_dir/part-r-00000

The job was completed in very less time, CPU Time = 1980 milli seconds.

References

1. https://www.tutorialspoint.com/map_reduce/implementation_in_hadoop.htm
2. https://hadoop.apache.org/docs/current/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html
3. http://www.devinline.com/2015/12/find-total-and-average-salary.html