Robert Werthman
CSCI 5673

Steps to Complete HW 3 Problem 7 : Running Hadoop

I. Signing Up and Creating the Instances

A. I'm not going to go into detail here but I created an AWS account and created 10 instances with a Linux AMI.
B. I created a public/private key to connect to the instances.
C. I then connected to each instance with the command:
   1. ssh -i bob.pem ec2-user@<public dns> where bob.pem is the asymmetric key pair and public dns is the public dns hostname Amazon assigned to each instance.
D. I turned on and off the instances by accessing the AWS EC2 management console.
E. I then had to make it so the master instance could connect to each slave instance without have to enter a password.
   1. I copied the original public/private keypair to the master so the master can connect to the other machines.
      a) scp -i bob.pem bob.pem ec2-user@<public dns>:~/
   2. I created another key pair so the master could send the public key to the other machines.
      a) ssh-keygen -t dsa -P '' -f ~/.ssh/id_dsa
      b) cat ~/.ssh/id_dsa.pub >> ~/.ssh/authorized_keys
   3. I then copied the public key of that new key pair to the slave machines.
      a) cat .ssh/id_dsa.pub | ssh -i <original keypair> <public dns of machine to copy public key to> 'cat >> .ssh/authorized_keys'
   4. This allowed me to ssh from the master to slaves without a password.

II. Configuring Hadoop

A. First I had to download hadoop on each instance with these commands.
   1. wget http://apache.mirrorcatalogs.com/hadoop/core/stable/hadoop-2.7.1.tar.gz
   2. tar xvzf hadoop-2.7.1.tar.gz
   3. rm hadoop-2.7.1.tar.gz
   4. mv hadoop-2.7.1 hadoop
B. I then had to set up the hadoop configuration files. I followed these two tutorials starting with the master.
   1. https://chawlasumit.wordpress.com/2015/03/09/install-a-multi-node-hadoop-cluster-on-ubuntu-14-04/
   2. https://hadoop.apache.org/docs/r2.7.0/hadoop-project-dist/hadoop-common/ClusterSetup.html
C. I copied the almost all of the configuration files from the master to the slaves.
   1. scp core-site.xml yarn-site.xml hdfs-site.xml <node private ip>:/home/ec2-user/hadoop/etc/hadoop/
D. I added the Hadoop executables to my Path environment variable by appending them to .bashrc.

    1. export HADOOP_PREFIX=/home/ec2-user/hadoop
    2. export PATH=$PATH:$HADOOP_PREFIX/bin
  E. I downloaded jps which allows you to see what java processes are running.
    1. sudo yum install java-1.7.0-openjdk*

III. Creating a Hadoop File System and Adding Files

A. There are a number of commands required to create the hadoop file system on the master and add files to it.
   1. Format the hadoop filesystem to the directory specified in core-site.xml
      a) bin/hdfs namenode -format
   2. Delete a hadoop fs directory
      a) bin/hdfs dfs -rm -r <directory>
   3. Make a Hadoop fs directory
      a) bin/hdfs dfs -mkdir <directory>
   4. Put a file into a Hadoop fs directory
      a) bin/hdfs dfs -put <file> <directory>
   5. See what is in a Hadoop directory
      a) bin/hdfs dfs -ls <directory>
   6. See what is in a Hadoop fs file
      a) bin/hdfs dfs -cat <file>
   7. Copy files to from the Hadoop fs to the local filesystem
      a) bin/hadoop fs -get <hadoop directory> <local fs directory>

IV. Creating and Running a Hadoop Job

A. The first thing you need to do to run Hadoop is to start the file system.
   1. sbin/start-dfs.sh
B. And the job manager/tracker.
   1. sbin/start-yarn.sh
C. If you are going to create a Hadoop job with Java there are a few commands to compile the code into a jar file which is run by Hadoop.
   1. export HADOOP_CLASSPATH=$(hadoop/bin/hadoop classpath)
   2. javac -classpath ${HADOOP_CLASSPATH} -d <directory>/ <class>.java
   3. jar -cvf <jarname>.jar -C <directory>/ .
   4. Source: http://janzhou.org/2014/10/08/how-to-compile-hadoop/
D. In my case I used examples provided with Hadoop to run the MapReduce jobs I needed.
   1. In order to output the words from the 100 MB text file with their individual word counts.
      a) bin/hadoop jar hadoop-mapreduce-examples-2.7.1.jar wordcount input output
   2. In order to sort the words with their word counts from highest frequency to lowest frequency. This takes the output of the first MapReduce job and uses it as input.
      a) st='sort -nrk2'; bin/hadoop jar hadoop-streaming-2.7.1.jar -input output/part-r-00000 -output outputSort -mapper /bin/cat -reducer "$st"
   3. The sources I used to figure out how to do this.
      a) https://hadoop.apache.org/docs/current/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html

b) https://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-common/SingleCluster.html

V. Analysis of The Performance of the Instances

How is performance affected by the number of instances?
I don't have any quantitative data to answer this question. Based on my observations of how the jobs were distributed in the cluster, the mapping part of the job was run on a single node while the reduce was run on another. This may have been affected by where the data was located on the nodes. I don't believe any parallelism was going on so increasing the number of nodes/instances would not do anything for performance. If jobs could be run in parallel then having more nodes would be beneficial because there would be more workers available to complete jobs.

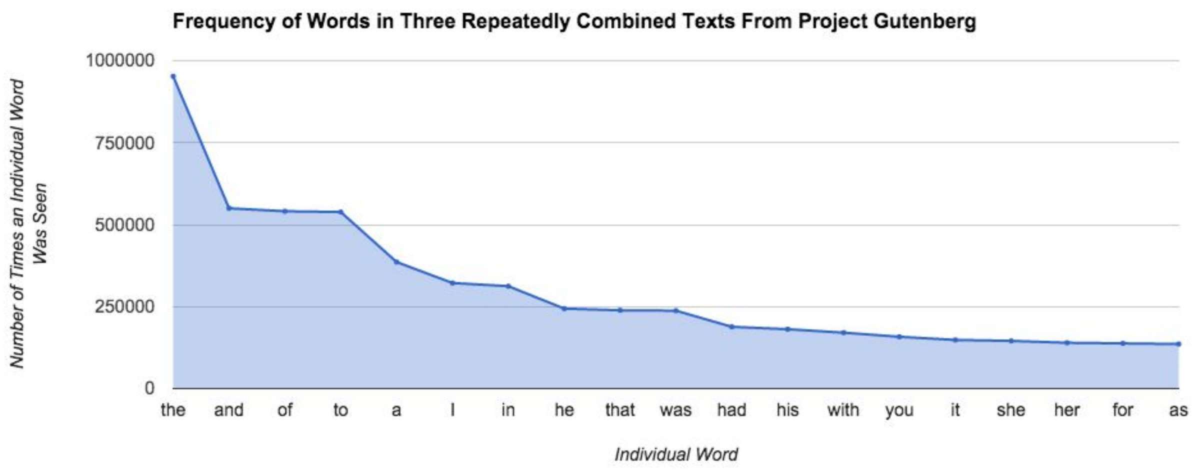
VI. Source Of The 100 MB File

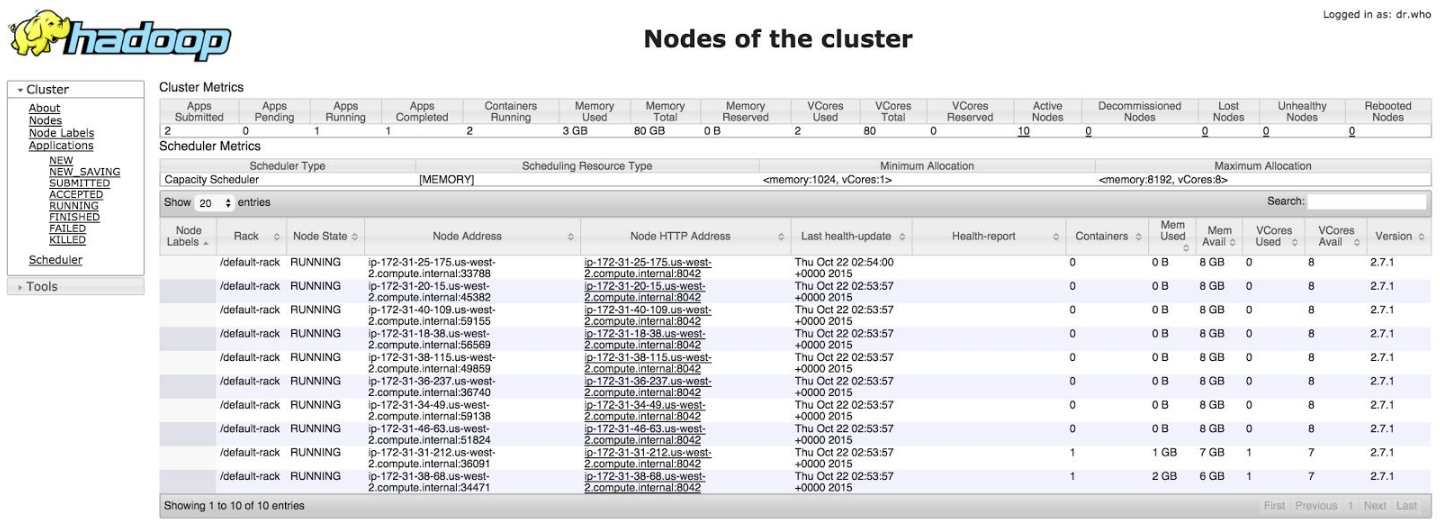
The source of the 100 MB file comes from three complete works found on project Gutenberg.

1. Project Gutenberg Complete Works of Winston Churchill by Winston Churchill
2. Complete Project Gutenberg William Dean Howells Works by William Dean Howells
3. The Complete Works of William Shakespeare by William Shakespeare

These were downloaded as text files repeatedly appended to a single text file until the size of the file reached 100 MB. I used Hadoop to separate the individual words out of the text file, count the total times the words were seen, and sorted these words from highest frequency to lowest.

The following graph shows the top 20 most frequent words in the 100 MB source file. I used a line graph instead of a histogram because I thought the data would be easier to understand.





Nodes of the cluster

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used	VCores Total	VCores Reserved	Active Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Rebooted Nodes
2	0	1	1	2	3 GB	80 GB	0 B	2	80	0	10	0	0	0	0

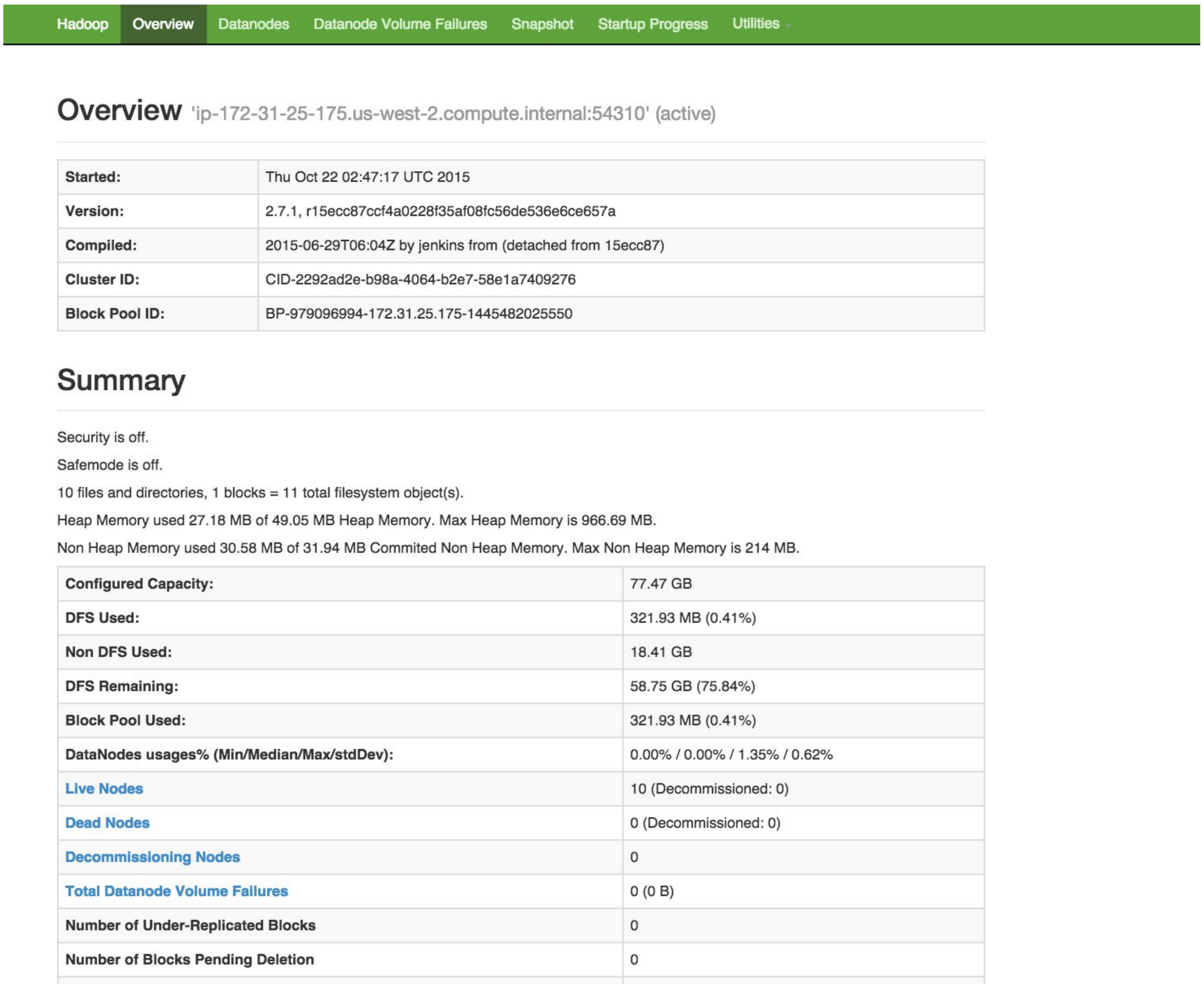
Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation
Capacity Scheduler	[MEMORY]	<memory:1024, vCores:1>	<memory:8192, vCores:8>

Show 20 entries Search:

Node Labels	Rack	Node State	Node Address	Node HTTP Address	Last health-update	Health-report	Containers	Mem Used	Mem Avail	VCores Used	VCores Avail	Version
	/default-rack	RUNNING	ip-172-31-25-175.us-west-2.compute.internal:33788	ip-172-31-25-175.us-west-2.compute.internal:8042	Thu Oct 22 02:54:00 +0000 2015		0	0 B	8 GB	0	8	2.7.1
	/default-rack	RUNNING	ip-172-31-20-15.us-west-2.compute.internal:45382	ip-172-31-20-15.us-west-2.compute.internal:8042	Thu Oct 22 02:53:57 +0000 2015		0	0 B	8 GB	0	8	2.7.1
	/default-rack	RUNNING	ip-172-31-40-109.us-west-2.compute.internal:59155	ip-172-31-40-109.us-west-2.compute.internal:8042	Thu Oct 22 02:53:57 +0000 2015		0	0 B	8 GB	0	8	2.7.1
	/default-rack	RUNNING	ip-172-31-18-38.us-west-2.compute.internal:56569	ip-172-31-18-38.us-west-2.compute.internal:8042	Thu Oct 22 02:53:57 +0000 2015		0	0 B	8 GB	0	8	2.7.1
	/default-rack	RUNNING	ip-172-31-38-115.us-west-2.compute.internal:49859	ip-172-31-38-115.us-west-2.compute.internal:8042	Thu Oct 22 02:53:57 +0000 2015		0	0 B	8 GB	0	8	2.7.1
	/default-rack	RUNNING	ip-172-31-36-237.us-west-2.compute.internal:36740	ip-172-31-36-237.us-west-2.compute.internal:8042	Thu Oct 22 02:53:57 +0000 2015		0	0 B	8 GB	0	8	2.7.1
	/default-rack	RUNNING	ip-172-31-34-49.us-west-2.compute.internal:59138	ip-172-31-34-49.us-west-2.compute.internal:8042	Thu Oct 22 02:53:57 +0000 2015		0	0 B	8 GB	0	8	2.7.1
	/default-rack	RUNNING	ip-172-31-46-63.us-west-2.compute.internal:51824	ip-172-31-46-63.us-west-2.compute.internal:8042	Thu Oct 22 02:53:57 +0000 2015		0	0 B	8 GB	0	8	2.7.1
	/default-rack	RUNNING	ip-172-31-31-212.us-west-2.compute.internal:36091	ip-172-31-31-212.us-west-2.compute.internal:8042	Thu Oct 22 02:53:57 +0000 2015		1	1 GB	7 GB	1	7	2.7.1
	/default-rack	RUNNING	ip-172-31-38-68.us-west-2.compute.internal:34471	ip-172-31-38-68.us-west-2.compute.internal:8042	Thu Oct 22 02:53:57 +0000 2015		1	2 GB	6 GB	1	7	2.7.1

Showing 1 to 10 of 10 entries



Overview 'ip-172-31-25-175.us-west-2.compute.internal:54310' (active)

Started:	Thu Oct 22 02:47:17 UTC 2015
Version:	2.7.1, r15ecc87ccf4a0228f35af08fc56de536e6ce657a
Compiled:	2015-06-29T06:04Z by jenkins from (detached from 15ecc87)
Cluster ID:	CID-2292ad2e-b98a-4064-b2e7-58e1a7409276
Block Pool ID:	BP-979096994-172.31.25.175-1445482025550

Summary

Security is off.

Safemode is off.

10 files and directories, 1 blocks = 11 total filesystem object(s).

Heap Memory used 27.18 MB of 49.05 MB Heap Memory. Max Heap Memory is 966.69 MB.

Non Heap Memory used 30.58 MB of 31.94 MB Commited Non Heap Memory. Max Non Heap Memory is 214 MB.

Configured Capacity:	77.47 GB
DFS Used:	321.93 MB (0.41%)
Non DFS Used:	18.41 GB
DFS Remaining:	58.75 GB (75.84%)
Block Pool Used:	321.93 MB (0.41%)
DataNodes usages% (Min/Median/Max/stdDev):	0.00% / 0.00% / 1.35% / 0.62%
Live Nodes	10 (Decommissioned: 0)
Dead Nodes	0 (Decommissioned: 0)
Decommissioning Nodes	0
Total Datanode Volume Failures	0 (0 B)
Number of Under-Replicated Blocks	0
Number of Blocks Pending Deletion	0