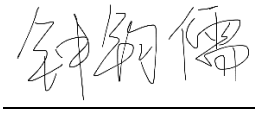
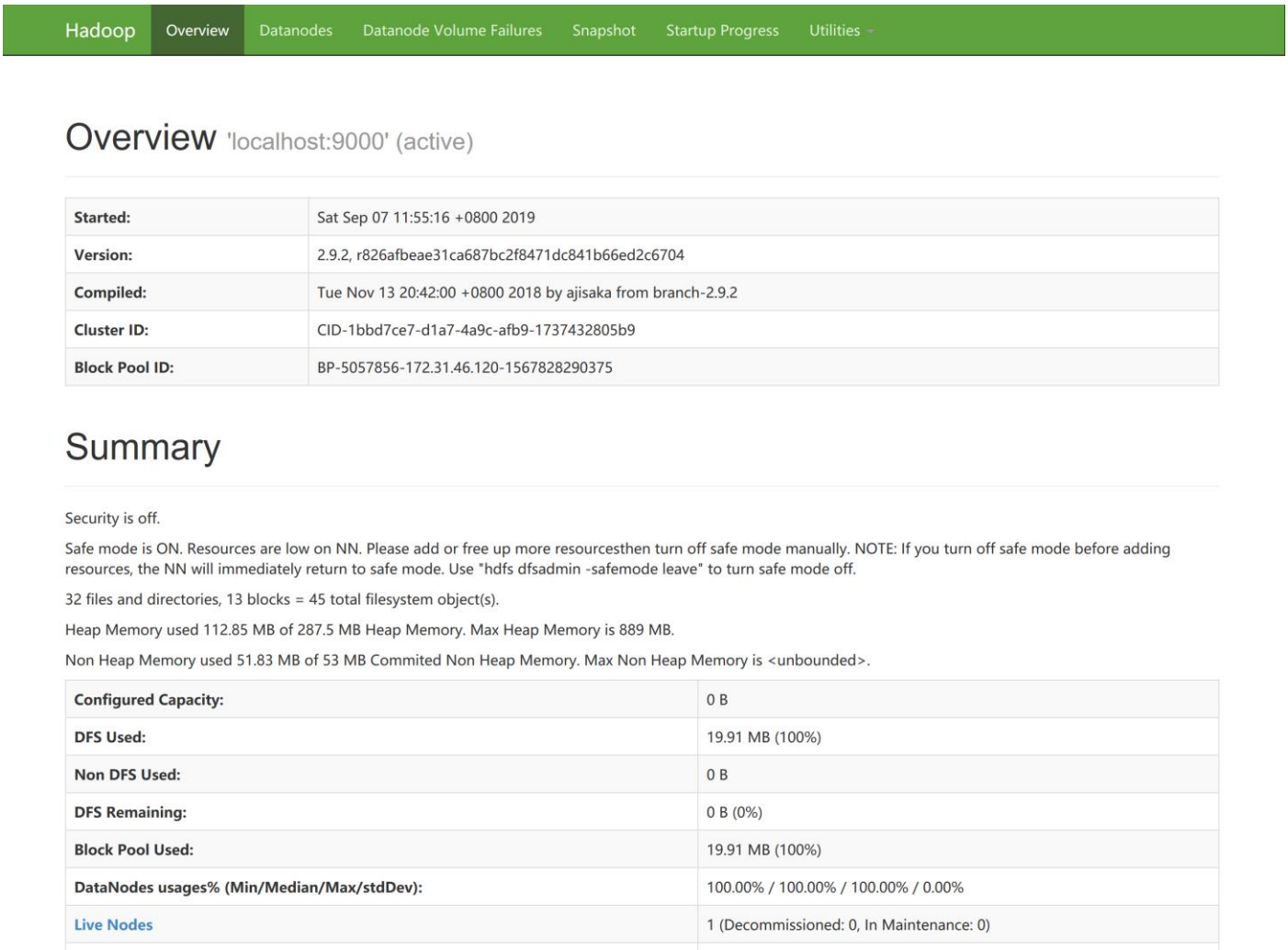
IERG 4300 – Homework #0

I declare that the assignment here submitted is original except for source material explicitly acknowledged, and that the same or related material has not been previously submitted for another course. I also acknowledge that I am aware of University policy and regulations on honesty in academic work, and of the disciplinary guidelines and procedures applicable to breaches of such policy and regulations, as contained in the website http://www.cuhk.edu.hk/policy/academichonesty/.

Student Name: Junru Zhong 鍾鈞儒 **Student ID:** 1155130306 **Signature**: 

a. Configurate a Pseudo-Distributed Hadoop Cluster

I configurated the cluster on AWS with a t3.large instance in HK. Here is the screenshot of the HDFS web interface, which can be find on http://address.to.ec2:50070.



Then, I ran the example "TeraSort" programme. The programme can be run by the following command:

```
ubuntu@ip-172-31-46-120:~/hadoop-2.9.2$ bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.9.2.jar teragen 100000 terasort/input
```

And the output of it looks like,

```
ubuntu@ip-172-31-46-120:~/hadoop-2.9.2$ bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.9.2.jar teragen 100000 terasort/input
19/09/09 11:49:56 INFO client.RMProxy: Connecting to ResourceManager at ec2-18-162-60-23.ap-east-1.compute.amazonaws.com/172.31.46.120:8032
19/09/09 11:49:57 INFO terasort.TeraGen: Generating 100000 using 2
19/09/09 11:49:57 INFO mapreduce.JobSubmitter: number of splits:2
19/09/09 11:49:57 INFO Configuration.deprecation: yarn.resourcemanager.system-metrics-publisher.enabled is deprecated. Instead, use yarn.system-metrics-publisher.enabled
19/09/09 11:49:57 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1567925462153_0025
19/09/09 11:49:57 INFO impl.YarnClientImpl: Submitted application application_1567925462153_0025
19/09/09 11:49:57 INFO mapreduce.Job: The url to track the job: http://ec2-18-162-60-23.ap-east-1.compute.amazonaws.com:8088/proxy/application_1567925462153_0025/
19/09/09 11:49:57 INFO mapreduce.Job: Running job: job_1567925462153_0025
19/09/09 11:50:03 INFO mapreduce.Job: Job job_1567925462153_0025 running in uber mode : false
19/09/09 11:50:03 INFO mapreduce.Job:  map 0% reduce 0%
19/09/09 11:50:09 INFO mapreduce.Job:  map 50% reduce 0%
19/09/09 11:50:10 INFO mapreduce.Job:  map 100% reduce 0%
19/09/09 11:50:10 INFO mapreduce.Job: Job job_1567925462153_0025 completed successfully
19/09/09 11:50:10 INFO mapreduce.Job: Counters: 31
        File System Counters
                FILE: Number of bytes read=0
                FILE: Number of bytes written=396592
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=164
                HDFS: Number of bytes written=10000000
                HDFS: Number of read operations=8
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=4
        Job Counters
                Launched map tasks=2
                Other local map tasks=2
                Total time spent by all maps in occupied slots (ms)=5972
                Total time spent by all reduces in occupied slots (ms)=0
                Total time spent by all map tasks (ms)=5972
                Total vcore-milliseconds taken by all map tasks=5972
                Total megabyte-milliseconds taken by all map tasks=6115328
        Map-Reduce Framework
                Map input records=100000
                Map output records=100000
                Input split bytes=164
                Spilled Records=0
                Failed Shuffles=0
                Merged Map outputs=0
                GC time elapsed (ms)=189
                CPU time spent (ms)=1550
                Physical memory (bytes) snapshot=376045568
                Virtual memory (bytes) snapshot=3961360384
                Total committed heap usage (bytes)=245366784
        org.apache.hadoop.examples.terasort.TeraGen$Counters
                CHECKSUM=214574985129000
        File Input Format Counters
                Bytes Read=0
        File Output Format Counters
                Bytes Written=10000000
```

With no error, continue to TeraSort by the following command.

```
ubuntu@ip-172-31-46-120:~/hadoop-2.9.2$ bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.9.2.jar terasort terasort/input terasort/output
```

After finishing this task, the output folder was added with the output file like the following. However, the output file is binary, and I cannot open it to see the inside.

```
ubuntu@ip-172-31-46-120:~/hadoop-2.9.2$ bin/hdfs dfs -ls /user/ubuntu/terasort/output
Found 3 items
-rw-r--r--   1 ubuntu supergroup          0 2019-09-09 11:53 /user/ubuntu/terasort/output/_SUCCESS
-rw-r--r--  10 ubuntu supergroup          0 2019-09-09 11:52 /user/ubuntu/terasort/output/_partition.lst
-rw-r--r--   1 ubuntu supergroup   10000000 2019-09-09 11:53 /user/ubuntu/terasort/output/part-r-00000
```

Finally, the TeraValidate can be run by the following command.

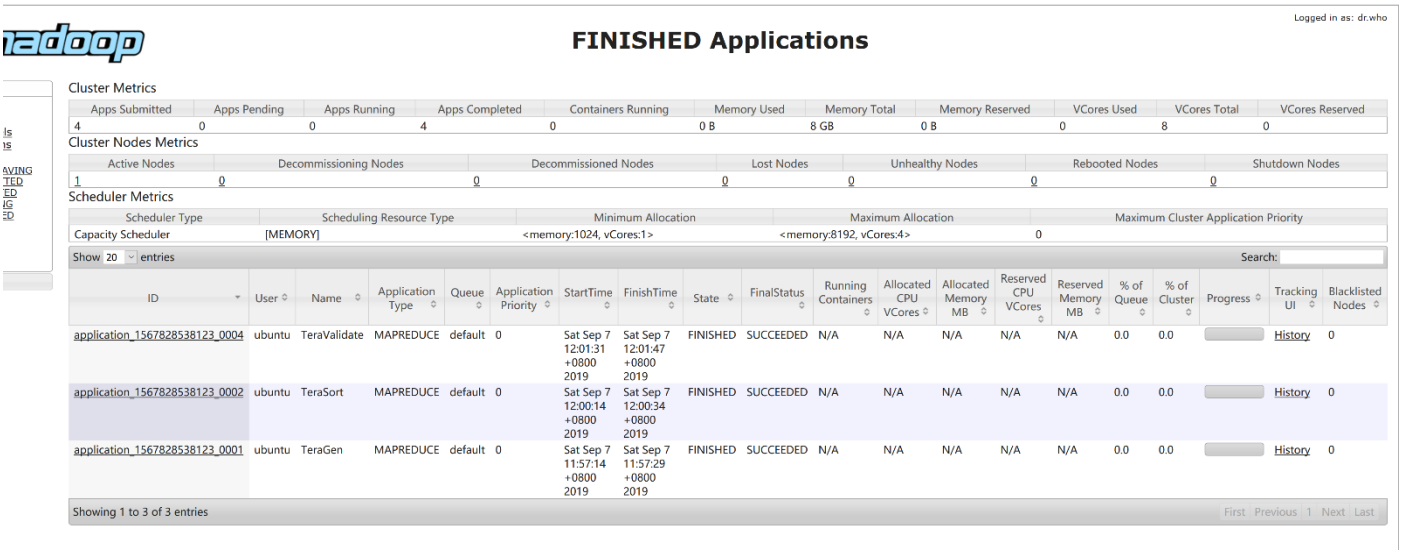
```
ubuntu@ip-172-31-46-120:~/hadoop-2.9.2$ bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.9.2.jar teravalidate terasort/output terasort/check
```

Then it outputted a checksum in the output file.

part-r-00000

```
1   checksum    c327a1c42c28
2
```

Here is the screenshot of the YARN web interface, showing three tasks on Terasort are finished.



Logged in as: dr.who

FINISHED Applications

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used	VCores Total	VCores Reserved
4	0	0	4	0	0 B	8 GB	0 B	0	8	0

Cluster Nodes Metrics

Active Nodes	Decommissioning Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Rebooted Nodes	Shutdown Nodes
1	0	0	0	0	0	0

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation	Maximum Cluster Application Priority
Capacity Scheduler	[MEMORY]	<memory:1024, vCores:1>	<memory:8192, vCores:4>	0

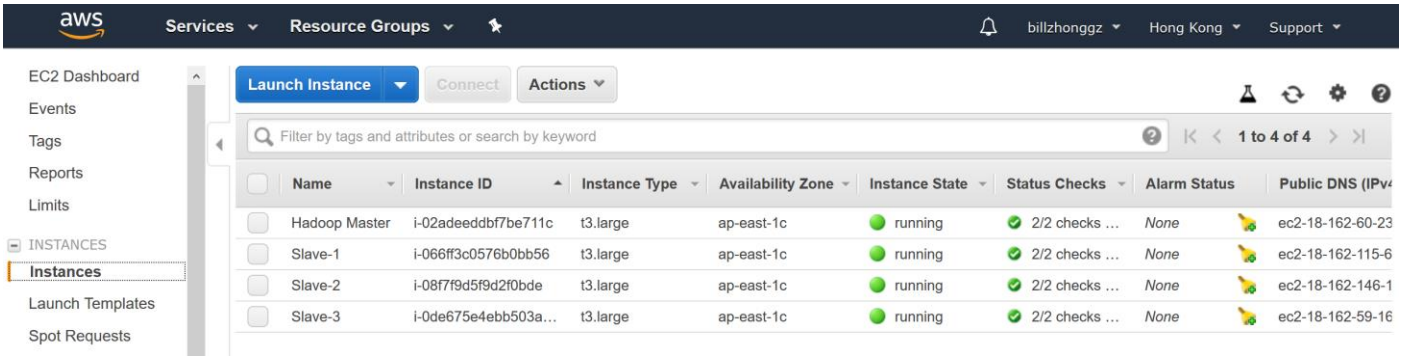
Show 20 entries Search:

ID	User	Name	Application Type	Queue	Application Priority	StartTime	FinishTime	State	FinalStatus	Running Containers	Allocated CPU VCores	Allocated Memory MB	Reserved CPU VCores	Reserved Memory MB	% of Queue	% of Cluster	Progress	Tracking UI	Blacklisted Nodes
application_1567828538123_0004	ubuntu	TeraValidate	MAPREDUCE	default	0	Sat Sep 7 12:01:31 +0800 2019	Sat Sep 7 12:01:47 +0800 2019	FINISHED	SUCCEEDED	N/A	N/A	N/A	N/A	N/A	0.0	0.0		History	0
application_1567828538123_0002	ubuntu	TeraSort	MAPREDUCE	default	0	Sat Sep 7 12:00:14 +0800 2019	Sat Sep 7 12:00:34 +0800 2019	FINISHED	SUCCEEDED	N/A	N/A	N/A	N/A	N/A	0.0	0.0		History	0
application_1567828538123_0001	ubuntu	TeraGen	MAPREDUCE	default	0	Sat Sep 7 11:57:14 +0800 2019	Sat Sep 7 11:57:29 +0800 2019	FINISHED	SUCCEEDED	N/A	N/A	N/A	N/A	N/A	0.0	0.0		History	0

Showing 1 to 3 of 3 entries First Previous 1 Next Last

b. Configure a Distributed Hadoop Cluster

I configurated the cluster with 4 instances on AWS. One as the master, three as the slaves.



aws Services Resource Groups billzhonggz Hong Kong Support

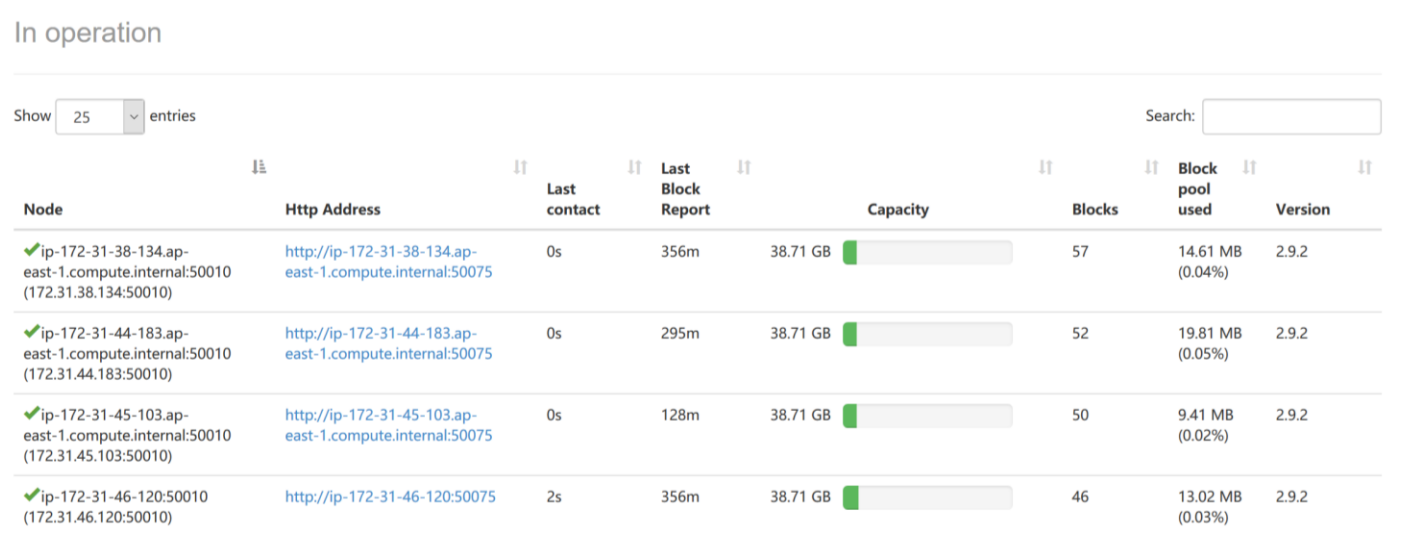
EC2 Dashboard
Events
Tags
Reports
Limits
INSTANCES
Instances
Launch Templates
Spot Requests

Launch Instance Connect Actions

Filter by tags and attributes or search by keyword 1 to 4 of 4

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4
Hadoop Master	i-02adeeddbf7be711c	t3.large	ap-east-1c	running	2/2 checks ...	None	ec2-18-162-60-23
Slave-1	i-066ff3c0576b0bb56	t3.large	ap-east-1c	running	2/2 checks ...	None	ec2-18-162-115-6
Slave-2	i-08f7f9d5f9d2f0bde	t3.large	ap-east-1c	running	2/2 checks ...	None	ec2-18-162-146-1
Slave-3	i-0de675e4ebb503a...	t3.large	ap-east-1c	running	2/2 checks ...	None	ec2-18-162-59-16

From the namenode web interface, information of 4 datanodes are shown. Due to the TeraSort 20G is extremely resource hungry, I also add the master machine into the computation and storage pool (datanode).



In operation

Show 25 entries Search:

Node	Http Address	Last contact	Last Block Report	Capacity	Blocks	Block pool used	Version
ip-172-31-38-134.ap-east-1.compute.internal:50010 (172.31.38.134:50010)	http://ip-172-31-38-134.ap-east-1.compute.internal:50075	0s	356m	38.71 GB	57	14.61 MB (0.04%)	2.9.2
ip-172-31-44-183.ap-east-1.compute.internal:50010 (172.31.44.183:50010)	http://ip-172-31-44-183.ap-east-1.compute.internal:50075	0s	295m	38.71 GB	52	19.81 MB (0.05%)	2.9.2
ip-172-31-45-103.ap-east-1.compute.internal:50010 (172.31.45.103:50010)	http://ip-172-31-45-103.ap-east-1.compute.internal:50075	0s	128m	38.71 GB	50	9.41 MB (0.02%)	2.9.2
ip-172-31-46-120:50010 (172.31.46.120:50010)	http://ip-172-31-46-120:50075	2s	356m	38.71 GB	46	13.02 MB (0.03%)	2.9.2

Then, I used the following command to generate 2GB of data,

```
ubuntu@ip-172-31-46-120:~/hadoop-2.9.2$ bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.9.2.jar teragen 20000000 terasort/input
```

Then the 2GB data is settled down in the file system.

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rw-r--r--	ubuntu	supergroup	0 B	Sep 09 12:05	3	128 MB	_SUCCESS
-rw-r--r--	ubuntu	supergroup	953.67 MB	Sep 09 12:05	3	128 MB	part-m-00000
-rw-r--r--	ubuntu	supergroup	953.67 MB	Sep 09 12:05	3	128 MB	part-m-00001

Showing 1 to 3 of 3 entries

Previous 1 Next

Then the actual TeraSort and TeraValidate programme can be run by this command, same as the previous ones. All of the three tasks were successful. The running time of these three tasks are shown below on Table 1.

application_1567925462153_0027	ubuntu	TeraValidate	MAPREDUCE	default	0	Mon Sep 9 11:57:22 +0800 2019	Mon Sep 9 11:57:35 +0800 2019	FINISHED	SUCCEEDED
application_1567925462153_0026	ubuntu	TeraSort	MAPREDUCE	default	0	Mon Sep 9 11:52:58 +0800 2019	Mon Sep 9 11:53:16 +0800 2019	FINISHED	SUCCEEDED
application_1567925462153_0025	ubuntu	TeraGen	MAPREDUCE	default	0	Mon Sep 9 11:49:57 +0800 2019	Mon Sep 9 11:50:08 +0800 2019	FINISHED	SUCCEEDED

Now move to TeraSort 20G. I generated the data by the following command,

```
ubuntu@ip-172-31-46-120:~/hadoop-2.9.2$ bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.9.2.jar teragen 200000000 terasort/input
```

20GB of dataset was generated.

Permission	Owner	Group	Size	Last Modified	Replication	Block Size	Name
-rw-r--r--	ubuntu	supergroup	0 B	Sep 09 12:20	3	128 MB	_SUCCESS
-rw-r--r--	ubuntu	supergroup	9.31 GB	Sep 09 12:20	3	128 MB	part-m-00000
-rw-r--r--	ubuntu	supergroup	9.31 GB	Sep 09 12:20	3	128 MB	part-m-00001

Showing 1 to 3 of 3 entries

Previous 1 Next

For the TeraSort 20G, I failed to run the sorting part in the default setting. By default, the TeraSort task runs only one reduce task, which be run on only one machine. However, the reduce process takes a lot of hard disk space that one machine cannot handle. So, by adding the parameter which increases the number of reduce tasks, the reduce tasks can be distributed to the entire cluster. Then it can be done by making use of all spaces on the whole cluster. The command with the parameter is like the follows, and I set 30 reduce tasks for this sorting job.

```
ubuntu@ip-172-31-46-120:~/hadoop-2.9.2$ bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.9.2.jar terasort -Dmapred.reduce.tasks=30 terasort/input terasort/output
```

Finally, I ran the TeraValidate task. All of the three tasks for TeraSort 20GB have finished. The running times can be seen on Table 1.

application_1567925462153_0034	ubuntu	TeraValidate	MAPREDUCE	default	0	Mon Sep 9 12:30:09 +0800 2019	Mon Sep 9 12:31:17 +0800 2019	FINISHED	SUCCEEDED
application_1567925462153_0033	ubuntu	TeraSort	MAPREDUCE	default	0	Mon Sep 9 12:20:38 +0800 2019	Mon Sep 9 12:28:06 +0800 2019	FINISHED	SUCCEEDED
application_1567925462153_0032	ubuntu	TeraGen	MAPREDUCE	default	0	Mon Sep 9 12:17:47 +0800 2019	Mon Sep 9 12:20:05 +0800 2019	FINISHED	SUCCEEDED

Task Name	2GB dataset	20GB dataset
TeraGen	20 sec	2 min, 18 sec
TeraSort (1 reduce task)	1 min, 16 sec	Failed
TeraSort (30 reduce tasks)	1 min, 11 sec	7 min, 28 sec
TeraValidate	24 sec	1 min, 7 sec

Table 1 TeraSort Running Times

c. Run a Python Programme Through Streaming

To run the Python word count programme, I need to download the scripts and dataset to the master machine. Then put the dataset to HDFS by this command,

```
# Create the folder on HDFS
ubuntu@ip-172-31-46-120:~/hadoop-2.9.2$ bin/hdfs dfs -mkdir -p wordcount/input
# Put the dataset to HDFS
ubuntu@ip-172-31-46-120:~/hadoop-2.9.2$ bin/hdfs dfs -put ../wordcount/large.txt
wordcount/input
```

In my case, I am using Ubuntu 18.04 across the cluster. Since Ubuntu 18.04 does not come with a Python 2.x, I need to install it by this command across the cluster (I have tried to modify the script to Python 3, but it failed),

```
sudo apt install python
```

Then, the programme can be run by this command,

```
ubuntu@ip-172-31-46-120:~/hadoop-2.9.2$ bin/hadoop jar share/hadoop/tools/lib/hadoop-
streaming-2.9.2.jar -file mapper.py -mapper mapper.py -file reducer.py -reducer reducer.py
-input /user/ubuntu/wordcount/input -output /user/ubuntu/wordcount/output
```

The result of it looks like following. And the running time can be seen on Table 2.

WordCount Tasks	Running Time
Python Through Streaming	3 mins, 27 sec
Compiled Java Source File	1 min, 36 sec

Table 2 WordCount Running Times

```
ubuntu@ip-172-31-46-120:~/hadoop-2.9.2$ tail ../part-00000
999997:993821   1
999998  1
999998:2553600  1
999999  3
999999:993821   1
99999:206184    1
9999:87 1
999:897791      1
99:95   1
9:0     1
```

d. Compile and Run a Java Programme

Following the instruction on https://hadoop.apache.org/docs/stable/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html, I compiled the Java source file, packed the jar package right on the virtual machine. It can be done by these commands.

```
# Get the environment variables set.
ubuntu@ip-172-31-46-120:~/wordcount$ export JAVA_HOME=/usr/java/default
ubuntu@ip-172-31-46-120:~/wordcount$ export PATH=${JAVA_HOME}bin:${PATH}
ubuntu@ip-172-31-46-120:~/wordcount$ export HADOOP_CLASSPATH=${JAVA_HOME}lib/tools.jar
# Compile the Java source file.
ubuntu@ip-172-31-46-120:~/wordcount$ ../hadoop-2.9.2/bin/hadoop com.sun.tools.javac.Main
WordCount.java
# Pack the jar package
ubuntu@ip-172-31-46-120:~/wordcount$ jar cf wc.jar WordCount*.class
# Run the jar package
ubuntu@ip-172-31-46-120:~/wordcount$ ../hadoop-2.9.2/bin/hadoop jar wc.jar WordCount
wordcount/input wordcount/output
```

The running time of this programme can be found on Table 2 on the pervious page. It can be seen that the Java programme is much faster than the Python one.

----- End of this submission. Last modified on Sept 9 -----