

CSS553 Programming Assignment 2

By

George Mathew

(A20352131)

The programming assignment involved sorting of data using the following methodologies

- 1)Simple Merge Sort
- 2)Shared Memory Sort
- 3)Hadoop Pseudo Mode Sorting
- 4)Hadoop 16 node cluster Sorting
- 5)Spark Pseudo Mode Sorting
- 6)Spark 16 node cluster Sorting

The assignment had 8 parts as explained below

1)Virtual Cluster One Node:

The following kinds of virtual instances were used.

- 1)c3.large
- 2)d2.xlarge

Ubuntu instances were used. SSH, All TCP, All UDP, All ICMP security groups were added to the instances

All instances were updated and java was installed. Raid was used to combine the available hard disk spaces.

Java Version :1.7.0_95

Spot instances were used. c3.large was bid at \$0.09 and d2.xlarge was bid at \$0.5

The initial Instance was made using d2.xlarge and an AMI image of this was made for future use

2) Shared Memory Sort

The problem with shared memory is that the data to be sorted is bigger than the available memory. So external sort algorithm was used to do this

10 gb of data was sorted using 3.5 gb of RAM

The screenshot shows the AWS EC2 Management Console interface. On the left is a navigation sidebar with sections like EC2 Dashboard, Events, Tags, Reports, Limits, Instances, Images, AMIs, Bundle Tasks, Elastic Block Store, Volumes, Snapshots, and Network & Security. The main area displays a table of instances. The first instance, `i-5096cf97`, is listed as terminated. The second instance, `i-d7e0b910`, is listed as running with a Public IP of `54.187.20.43`. A detailed view of the running instance shows its state as running, type as c3.xlarge, private DNS as `ip-172-31-26-141.us-west-2.compute.internal`, and private IP as `172.31.26.141`. It also lists secondary private IPs, public IP, elastic IP, availability zone (us-west-2a), security groups (`launch-wizard-45`), and scheduled events.

```
ubuntu@ip-172-31-26-141: /mnt
ubuntu@ip-172-31-26-141: /mnt$ sudo chmod 777 auto.sh
ubuntu@ip-172-31-26-141: /mnt$ ./auto.sh
Generating 10 GB file
10GB file generated
10G    input
Sorting and dividing into Chunks
Time Taken 310s
Doing External Sort
Flags initiated and Reading from Chunks
Time Taken 3130s
10G    Output
Validating the sorted value
Records: 107000000
Checksum: 33054c596163a17
Duplicate keys: 0
SUCCESS - all records are in order
ubuntu@ip-172-31-26-141: /mnt$
ubuntu@ip-172-31-26-141: /mnt$ du -h Output
16G    Output
ubuntu@ip-172-31-26-141: /mnt$
```

The sorting was done in 57.3 minutes

Valsort was done and the output was verified

3)Virtual Cluster

A Virtual Cluster of 17 nodes was setup using 1 d2.xlarge and 16 c3.large instances using the images made from step 1.

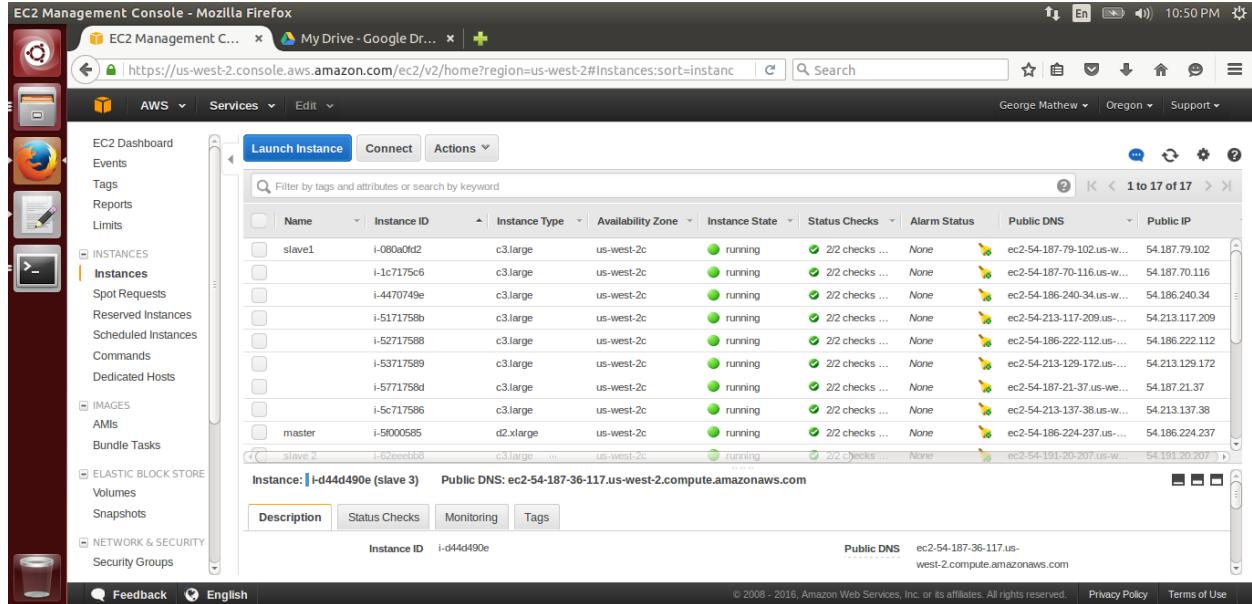
An EBS volumes were also added to the c3.large systems. RAID0 was used to configure the disks of the instances

Password less SSH was enabled and the keys of all the instances were added to the master so that all the instances can be accessed from the Master

A user 'me' was made in all the instances and was granted super user privileges

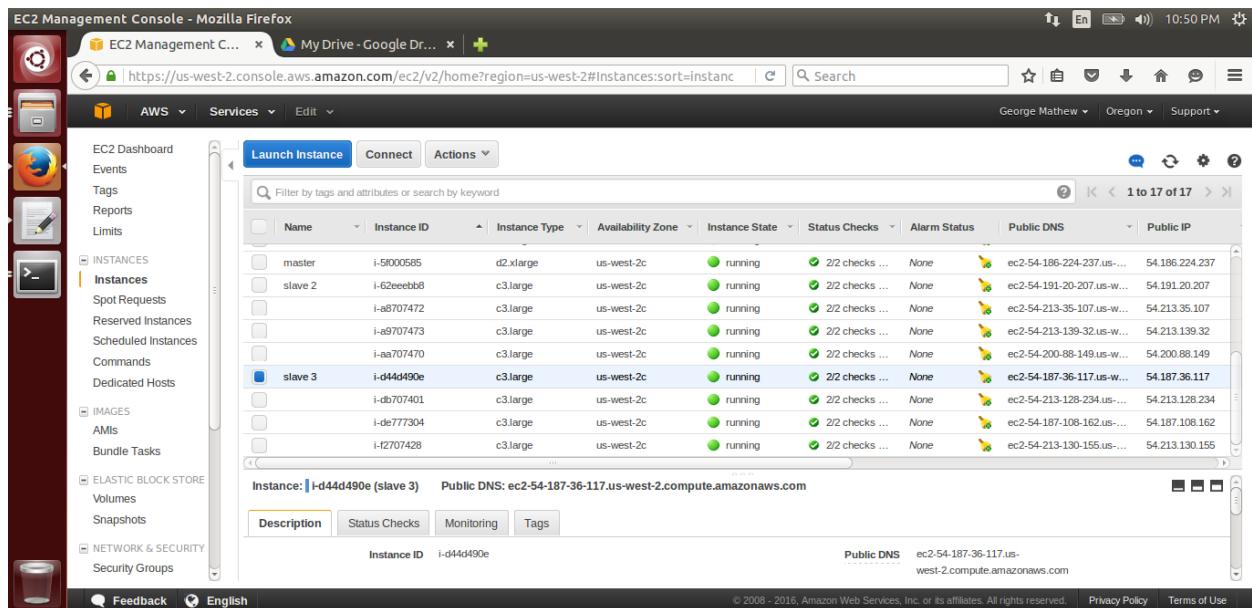
Pssh was used to push commands from the master to all the nodes

Scp was used to push data from the master to all the 16 nodes



The screenshot shows the AWS EC2 Management Console in Mozilla Firefox. The left sidebar is collapsed. The main area displays a table of 17 instances. The columns include Name, Instance ID, Instance Type, Availability Zone, Instance State, Status Checks, Alarm Status, Public DNS, and Public IP. The 'Instances' section in the sidebar is selected. The master instance is highlighted in blue.

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS	Public IP
slave1	i-080a0fd2	c3.large	us-west-2c	running	2/2 checks ...	None	ec2-54-187-79-102.us-w...	54.187.79.102
	i-1c7175c6	c3.large	us-west-2c	running	2/2 checks ...	None	ec2-54-187-70-116.us-w...	54.187.70.116
	i-4470749e	c3.large	us-west-2c	running	2/2 checks ...	None	ec2-54-186-240-34.us-w...	54.186.240.34
	i-5171758b	c3.large	us-west-2c	running	2/2 checks ...	None	ec2-54-213-117-209.us-...	54.213.117.209
	i-5271758b	c3.large	us-west-2c	running	2/2 checks ...	None	ec2-54-186-222-112.us-...	54.186.222.112
	i-53717589	c3.large	us-west-2c	running	2/2 checks ...	None	ec2-54-213-129-172.us-...	54.213.129.172
	i-5671758d	c3.large	us-west-2c	running	2/2 checks ...	None	ec2-54-187-21-37.us-w...	54.187.21.37
	i-5c717586	c3.large	us-west-2c	running	2/2 checks ...	None	ec2-54-213-137-38.us-w...	54.213.137.38
master	i-50000585	d2.xlarge	us-west-2c	running	2/2 checks ...	None	ec2-54-186-224-237.us-...	54.186.224.237
slave 2	i-62eeeb8b	c3.large	us-west-2c	running	2/2 checks ...	None	ec2-54-191-20-207.us-w...	54.191.20.207
Instance: i-d44d490e (slave 3) Public DNS: ec2-54-187-36-117.us-west-2.compute.amazonaws.com								
Description Status Checks Monitoring Tags			Instance ID: i-d44d490e Public DNS: ec2-54-187-36-117.us-west-2.compute.amazonaws.com					
Instance ID: i-d44d490e			Public DNS: ec2-54-187-36-117.us-west-2.compute.amazonaws.com					



The screenshot shows the AWS EC2 Management Console in Mozilla Firefox. The left sidebar is collapsed. The main area displays a table of 17 instances. The columns include Name, Instance ID, Instance Type, Availability Zone, Instance State, Status Checks, Alarm Status, Public DNS, and Public IP. The 'Instances' section in the sidebar is selected. The slave 3 instance is highlighted in blue.

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS	Public IP
master	i-50000585	d2.xlarge	us-west-2c	running	2/2 checks ...	None	ec2-54-186-224-237.us-...	54.186.224.237
slave 2	i-62eeeb8b	c3.large	us-west-2c	running	2/2 checks ...	None	ec2-54-191-20-207.us-w...	54.191.20.207
	i-a8707472	c3.large	us-west-2c	running	2/2 checks ...	None	ec2-54-213-35-107.us-w...	54.213.35.107
	i-a9707473	c3.large	us-west-2c	running	2/2 checks ...	None	ec2-54-213-139-32.us-w...	54.213.139.32
	i-aa707470	c3.large	us-west-2c	running	2/2 checks ...	None	ec2-54-200-88-149.us-w...	54.200.88.149
slave 3	i-d44d490e	c3.large	us-west-2c	running	2/2 checks ...	None	ec2-54-187-36-117.us-w...	54.187.36.117
	i-db707401	c3.large	us-west-2c	running	2/2 checks ...	None	ec2-54-213-128-234.us-...	54.213.128.234
	i-de777304	c3.large	us-west-2c	running	2/2 checks ...	None	ec2-54-187-108-162.us-...	54.187.108.162
	i-f2707428	c3.large	us-west-2c	running	2/2 checks ...	None	ec2-54-213-130-155.us-...	54.213.130.155
Instance: i-d44d490e (slave 3) Public DNS: ec2-54-187-36-117.us-west-2.compute.amazonaws.com								
Description Status Checks Monitoring Tags			Instance ID: i-d44d490e Public DNS: ec2-54-187-36-117.us-west-2.compute.amazonaws.com					
Instance ID: i-d44d490e			Public DNS: ec2-54-187-36-117.us-west-2.compute.amazonaws.com					

4)Hadoop

Hadoop was installed and used in pseudo mode and 16 cluster

Hadoop 2.7.2 was downloaded and moved to my mount point /mnt

The bashrc file for the user 'me' was edited and the following lines were added

```
export JAVA_HOME=/usr/lib/jvm/java-7-openjdk-amd64
export HADOOP_HOME=/mnt/hadoop
export PATH=${JAVA_HOME}/bin:${PATH}
export HADOOP_CLASSPATH=${JAVA_HOME}/lib/tools.jar
export PATH=$PATH:$HADOOP_HOME/bin
export PATH=$PATH:$HADOOP_HOME/sbin
export HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export YARN_HOME=$HADOOP_HOME
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
```

The following xml files inside Hadoop/etc/Hadoop was configured for using Hadoop

Core-site.xml

```
<configuration>
<property>
<name>fs.default.name</name>
<value>hdfs://localhost:9000</value>
</property>
<property>
<name>hadoop.tmp.dir</name>
<value>/mnt/raid/tmp/</value>
</property>
</configuration>
```

A tmp file was made in the location /mnt/raid/tmp so that that the Hadoop can use this location for its temporary files. This was pointed to the /mnt so that EBS volumes and s3 volumes can be mounted here if required

Hdfs-site.xml

```
<configuration>
<property>
<name>dfs.replication</name>
<value>1</value>
</property>
<property>
<name>dfs.namenode.name.dir</name>
<value>file:/mnt/hadoop/hadoop_data/hdfs/namenode</value>
</property>
```

```

<property>
<name>dfs.datanode.data.dir</name>
<value>file:/mnt/hadoop/hadoop_data/hdfs/datanode</value>
</property>
</configuration>

```

Replication was turned off so that data won't be replicated in other nodes and space can be conserved
The locations for the data node and namenode was pointed to the /mnt so that additional devices can be mounted if necessary

Mapred-site.xml

```

<configuration>
<property>
<name>mapreduce.framework.name</name>
<value>yarn</value>
</property>
<property>
<name>mapreduce.jobtracker.address</name>
<value>hdfs://localhost:9001</value>
</property>
<property>
<name>mapred.tasktracker.map.tasks</name>
<value>2</value>
</property>
<property>
<name>mapred.tasktracker.reduce.tasks</name>
<value>2</value>
</property>
</configuration>

```

The reducer and map tasks were set as two as there were only 2 virtual cores on the c3.large systems which were the slaves

Yarn-site.xml

```

<configuration>
<property>
<name>yarn.resourcemanager.hostname</name>
<value>localhost</value>
</property>
<property>
<name>yarn.nodemanager.aux-services</name>
<value>mapreduce_shuffle</value>
</property>
<property>
<name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>
<value>org.apache.hadoop.mapred.ShuffleHandler</value>
</property>
<property>
<name>yarn.resourcemanager.scheduler.address</name>

```

```
<value>localhost:8030</value>
</property>
<property>
<name>yarn.resourcemanager.address</name>
<value>localhost:8032</value>
</property>
<property>
<name>yarn.resourcemanager.webapp.address</name>
<value>localhost:8088</value>
</property>
<property>
<name>yarn.resourcemanager.resource-tracker.address</name>
<value>localhost:8031</value>
</property>
<property>
<name>yarn.resourcemanager.admin.address</name>
<value>localhost:8033</value>
</property>
</configuration>
```

All the processes of yarns are pointed to different ports

Slaves

Localhost is added to the slave file

In the pseudo mode the master and the slave both are in one system and all the processes happen in the same node

Then namenode is formatted using the command

Hdfs namenode -format

Then dfs is started using the command start-dfs.sh

And then the yarn is started using the command start-yarn.sh

Then using the command jps all the running Hadoop processes can be displayed

```

me@ip-172-31-11-124: /mnt/hadoop/hadoop_data/hdfs/datanode
16/03/29 05:59:48 INFO util.GSet: 0.029999999329447746% max memory 889 MB = 273.1 KB
16/03/29 05:59:48 INFO util.GSet: capacity = 2^15 = 32768 entries
Re-format filesystem in Storage Directory /mnt/hadoop/hadoop_data/hdfs/namenode ? (Y or N) y
16/03/29 05:59:50 INFO namenode.FSImage: Allocated new BlockPoolId: BP-1588025761-172.31.11.124-1459231190390
16/03/29 05:59:50 INFO namenode.NNStorageRetentionManager: Going to retain 1 images with txid >= 0
16/03/29 05:59:50 INFO util.ExitUtil: Exiting with status 0
16/03/29 05:59:50 INFO namenode.NameNode: SHUTDOWN_MSG:
*****STARTUP_MSG: Shutting down NameNode at ip-172-31-11-124.us-west-2.compute.internal/172.31.11.124
*****
me@ip-172-31-11-124:/mnt/hadoop/hadoop_data/hdfs/datanode$ start-all.sh
This script is Deprecated. Instead use start-dfs.sh and start-yarn.sh
16/03/29 06:00:00 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Starting namenodes on [localhost]
localhost: starting namenode, logging to /mnt/hadoop/logs/hadoop-me-namenode-ip-172-31-11-124.out
localhost: starting datanode, logging to /mnt/hadoop/logs/hadoop-me-datanode-ip-172-31-11-124.out
Starting secondary namenodes [0.0.0.0]
0.0.0.0: starting secondarynamenode, logging to /mnt/hadoop/logs/hadoop-me-secondarynamenode-ip-172-31-11-124.out
16/03/29 06:00:17 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
starting yarn daemons
starting resourcemanager, logging to /mnt/hadoop/logs/yarn-me-resourcemanager-ip-172-31-11-124.out
localhost: starting nodemanager, logging to /mnt/hadoop/logs/yarn-me-nodemanager-ip-172-31-11-124.out
me@ip-172-31-11-124:/mnt/hadoop/hadoop_data/hdfs/datanode$ ls
current_in_use.lock
me@ip-172-31-11-124:/mnt/hadoop/hadoop_data/hdfs/datanode$ jps
12021 Jps
11193 DataNode
11403 SecondaryNameNode
11559 ResourceManager
11708 NodeManager
11019 NameNode
me@ip-172-31-11-124:/mnt/hadoop/hadoop_data/hdfs/datanode$ 

```

The single node was setup on the system with ip 172.31.11.124

About the Cluster - Mozilla Firefox

ec2-54-186-224-237.us-west-2.compute.amazonaws.com:8088/cluster/cluster

About 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
0	0	0	0	0 B	128 GB	0 B	0	128	0	16	0	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>

Cluster Overview

Cluster ID: 1459226183235
ResourceManager state: STARTED
ResourceManager HA state: active
ResourceManager RMStateStore: org.apache.hadoop.yarn.server.resourcemanager.recovery.NullRMStateStore
ResourceManager started on: Tue Mar 29 04:36:23 +0000 2016
ResourceManager version: 2.7.1 from 15ecc87ccf4a0228f35af08fc56de536e6ce657a by jenkins source checksum 1042198b3cfb903a5d8de2fdcd09218 on 2015-06-29T06:12Z
Hadoop version: 2.7.1 from 15ecc87ccf4a0228f35af08fc56de536e6ce657a by jenkins source checksum fc0a1a23fc1868e4d5ee7fa2b28a58a on 2015-06-29T06:04Z

On this system the Hadoop program was written srt.java

```

import java.io.IOException;
import java.util.StringTokenizer;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.io.NullWritable;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;

```

```

import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class srt {

    public static class Mppr
        extends Mapper<Object, Text, Text, NullWritable>{

        NullWritable n=NullWritable.get();
        Text word = new Text();

        public void map(Object key, Text value, Context context
                        ) throws IOException, InterruptedException {

            word.set(value.toString()+" ");
            context.write(word, n);
        }
    }

    public static class Rdcr
        extends Reducer<Text,NullWritable,Text,NullWritable> {

        NullWritable s=NullWritable.get();
        public void reduce(Text key, Iterable<NullWritable> values,
                           Context context
                           ) throws IOException, InterruptedException {

            for (NullWritable val : values) {
                s= val;
            }

            context.write(key,s);
        }
    }

    public static void main(String[] args) throws Exception {
        Configuration conf = new Configuration();
        Job job = Job.getInstance(conf, "word count");
        job.setJarByClass(srt.class);
        job.setMapperClass(Mppr.class);
        job.setCombinerClass(Rdcr.class);
        job.setReducerClass(Rdcr.class);
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(NullWritable.class);
        FileInputFormat.addInputPath(job, new Path(args[0]));
        FileOutputFormat.setOutputPath(job, new Path(args[1]));
        System.exit(job.waitForCompletion(true) ? 0 : 1);
    }
}

```

This was compiled using the command

```
bin/hadoop com.sun.tools.javac.Main srt.java
```

This was made into a jar using the command

```
jar cf srt.jar srt*.class
```

5)Hadoop Sort

Pseudo mode

10 gb data was sorted in pseudo mode using one c3.large system with hadoop

The screenshot shows the AWS EC2 Management Console in Mozilla Firefox. The main window displays a list of EC2 instances, with one instance highlighted: "1gb hadoop sort". The details pane shows the instance type as "c3.large", private DNS as "ip-172-31-28-132.us-west-2.compute.internal", and elastic IP as "172.31.28.132". The bottom part of the screenshot shows a terminal window on the instance itself, displaying the command-line steps to run a Hadoop sort job.

```
me@ip-172-31-28-132:~$ du -h /mnt/input
10G  /mnt/input
me@ip-172-31-28-132:~$ hadoop fs -copyFromLocal /mnt/input /
16/03/31 14:32:09 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
me@ip-172-31-28-132:~$ hadoop jar /home/me/srt.jar srt /input /output
16/03/31 14:48:56 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
16/03/31 14:48:57 INFO client.RMProxy: Connecting to ResourceManager at localhost/127.0.0.1:8032
16/03/31 14:48:57 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
16/03/31 14:48:57 INFO input.FileInputFormat: Total input paths to process : 1
16/03/31 14:48:58 INFO mapreduce.JobSubmitter: number of splits:80
16/03/31 14:48:58 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1459433941764_0001
16/03/31 14:48:59 INFO impl.YarnClientImpl: Submitted application application_1459433941764_0001
16/03/31 14:48:59 INFO mapreduce.Job: The url to track the job: http://ip-172-31-28-132.us-west-2.compute.internal:8088/proxy/application_1459433941764_0001/
16/03/31 14:48:59 INFO mapreduce.Job: Running job: job_1459433941764_0001
16/03/31 14:49:06 INFO mapreduce.Job: Job job_1459433941764_0001 running in uber mode : false
16/03/31 14:49:06 INFO mapreduce.Job: map 0% reduce 0%
16/03/31 14:49:35 INFO mapreduce.Job: map 1% reduce 0%
16/03/31 14:49:38 INFO mapreduce.Job: map 2% reduce 0%
16/03/31 14:49:41 INFO mapreduce.Job: map 3% reduce 0%
16/03/31 14:49:57 INFO mapreduce.Job: map 4% reduce 0%
16/03/31 14:50:01 INFO mapreduce.Job: map 5% reduce 0%
16/03/31 14:50:31 INFO mapreduce.Job: map 6% reduce 0%
16/03/31 14:50:51 INFO mapreduce.Job: map 7% reduce 0%
16/03/31 14:51:11 INFO mapreduce.Job: map 8% reduce 0%
16/03/31 14:51:43 INFO mapreduce.Job: map 10% reduce 0%
16/03/31 14:51:47 INFO mapreduce.Job: map 11% reduce 0%
16/03/31 14:52:11 INFO mapreduce.Job: map 12% reduce 0%
16/03/31 14:52:15 INFO mapreduce.Job: map 13% reduce 0%
16/03/31 14:52:45 INFO mapreduce.Job: map 14% reduce 0%
16/03/31 14:53:03 INFO mapreduce.Job: map 15% reduce 0%
16/03/31 14:53:41 INFO mapreduce.Job: map 16% reduce 0%
```

```
me@ip-172-31-28-132: ~
16/03/31 15:40:15 INFO mapreduce.Job: map 100% reduce 93%
16/03/31 15:40:27 INFO mapreduce.Job: map 100% reduce 94%
16/03/31 15:40:43 INFO mapreduce.Job: map 100% reduce 95%
16/03/31 15:40:59 INFO mapreduce.Job: map 100% reduce 96%
16/03/31 15:41:14 INFO mapreduce.Job: map 100% reduce 97%
16/03/31 15:41:33 INFO mapreduce.Job: map 100% reduce 98%
16/03/31 15:41:45 INFO mapreduce.Job: map 100% reduce 99%
16/03/31 15:42:07 INFO mapreduce.Job: map 100% reduce 100%
16/03/31 15:42:15 INFO mapreduce.Job: Job job_1459433941764_0001 completed successfully
16/03/31 15:42:16 INFO mapreduce.Job: Counters: 50
    File System Counters
        FILE: Number of bytes read=32267461776
        FILE: Number of bytes written=43139831298
        FILE: Number of read operations=0
        FILE: Number of large read operations=0
        FILE: Number of write operations=0
        HDFS: Number of bytes read=10650330944
        HDFS: Number of bytes written=106500000000
        HDFS: Number of read operations=243
        HDFS: Number of large read operations=0
        HDFS: Number of write operations=2
    Job Counters
        Killed map tasks=2
        Launched map tasks=82
        Launched reduce tasks=1
        Data-local map tasks=82
        Total time spent by all maps in occupied slots (ms)=12820047
        Total time spent by all reduces in occupied slots (ms)=2813186
        Total time spent by all map tasks (ms)=12820047
        Total time spent by all reduce tasks (ms)=2813186
        Total vcore-seconds taken by all map tasks=12820047
        Total vcore-seconds taken by all reduce tasks=2813186
        Total megabyte-seconds taken by all map tasks=13127728128
        Total megabyte-seconds taken by all reduce tasks=2880702464
```

```
me@ip-172-31-28-132: ~
Total megabyte-seconds taken by all map tasks=13127728128
Total megabyte-seconds taken by all reduce tasks=2880702464
Map-Reduce Framework
    Map input records=106500000
    Map output records=106500000
    Map output bytes=106500000000
    Map output materialized bytes=10863000480
    Input split bytes=7360
    Combine input records=106500000
    Combine output records=106500000
    Reduce input groups=106500000
    Reduce shuffle bytes=10863000480
    Reduce input records=106500000
    Reduce output records=106500000
    Spilled Records=422847650
    Shuffled Maps =80
    Failed Shuffles=0
    Merged Map outputs=80
    GC time elapsed (ms)=74780
    CPU time spent (ms)=1484750
    Physical memory (bytes) snapshot=20625170432
    Virtual memory (bytes) snapshot=67025125376
    Total committed heap usage (bytes)=16484139008
Shuffle Errors
    BAD_ID=0
    CONNECTION=0
    IO_ERROR=0
    WRONG_LENGTH=0
    WRONG_MAP=0
    WRONG_REDUCE=0
File Input Format Counters
    Bytes Read=10650323584
File Output Format Counters
    Bytes Written=106500000000
me@ip-172-31-28-132: ~
```

1 GB was sorted in 24.7 minutes

The output was verified using `valsrt`

Hadoop Cluster

Then Hadoop was setup on a 16 node cluster .

The slaves file in master was updated with ips of all the 16 nodes

172.31.11.124
172.31.4.234
172.31.13.181
172.31.12.191
172.31.4.190
172.31.14.98
172.31.9.196
172.31.14.174
172.31.0.15
172.31.11.64
172.31.3.97
172.31.6.168
172.31.15.117
172.31.14.57
172.31.7.94
172.31.11.46

The core-site of the master was updated with the ip of the master.

```
<configuration>
<property>
<name>fs.default.name</name>
<value>hdfs://172.31.14.182:9000</value>
</property>
<property>
```

```
<name>hadoop.tmp.dir</name>
<value>/mnt/raid/tmp/</value>
</property>
</configuration>
```

Hdfs site didn't need any change

Mapred site was updated with the ip of the master

```
<configuration>
<property>
<name>mapreduce.framework.name</name>
<value>yarn</value>
</property>
<property>
<name>mapreduce.jobtracker.address</name>
<value>hdfs://172.31.14.182:9001</value>
</property>
<property>
<name>mapred.tasktracker.map.tasks</name>
<value>2</value>
</property>
<property>
<name>mapred.tasktracker.reduce.tasks</name>
<value>2</value>
</property>
</configuration>
```

The yarn site was updated with the ip of the master

```
<configuration>
<property>
<name>yarn.resourcemanager.hostname</name>
<value>172.31.14.182</value>
</property>
<property>
<name>yarn.nodemanager.aux-services</name>
<value>mapreduce_shuffle</value>
</property>
<property>
<name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>
<value>org.apache.hadoop.mapred.ShuffleHandler</value>
</property>
<property>
<name>yarn.resourcemanager.scheduler.address</name>
<value>172.31.14.182:8030</value>
</property>
<property>
<name>yarn.resourcemanager.address</name>
<value>172.31.14.182:8032</value>
</property>
```

```

<property>
<name>yarn.resourcemanager.webapp.address</name>
<value>172.31.14.182:8088</value>
</property>
<property>
<name>yarn.resourcemanager.resource-tracker.address</name>
<value>172.31.14.182:8031</value>
</property>
<property>
<name>yarn.resourcemanager.admin.address</name>
<value>172.31.14.182:8033</value>
</property>
</configuration>

```

The slaves are specified in the slaves folder and the master node starts the data nodes on these slaves

This configured Hadoop is pushed from the master to the slaves .As these files have the ip of the master the node manager,resource manager,namenode for the slaves are working on the master node

Node-Id	Node-State	Node-Http-Address	Number-of-Running-Containers
ip-172-31-15-117.us-west-2.compute.internal:45632	RUNNING	ip-172-31-15-117.us-west-2.compute.internal:8042	
0			
ip-172-31-12-191.us-west-2.compute.internal:42707	RUNNING	ip-172-31-12-191.us-west-2.compute.internal:8042	
0			
ip-172-31-14-174.us-west-2.compute.internal:37146	RUNNING	ip-172-31-14-174.us-west-2.compute.internal:8042	
0			
ip-172-31-11-124.us-west-2.compute.internal:42270	RUNNING	ip-172-31-11-124.us-west-2.compute.internal:8042	
0			
ip-172-31-4-234.us-west-2.compute.internal:46748	RUNNING	ip-172-31-4-234.us-west-2.compute.internal:8042	
0			
ip-172-31-7-94.us-west-2.compute.internal:33249	RUNNING	ip-172-31-7-94.us-west-2.compute.internal:8042	0
ip-172-31-14-98.us-west-2.compute.internal:34195	RUNNING	ip-172-31-14-98.us-west-2.compute.internal:8042	
0			
ip-172-31-14-57.us-west-2.compute.internal:32901	RUNNING	ip-172-31-14-57.us-west-2.compute.internal:8042	
0			
ip-172-31-3-97.us-west-2.compute.internal:40914	RUNNING	ip-172-31-3-97.us-west-2.compute.internal:8042	0
ip-172-31-13-181.us-west-2.compute.internal:56416	RUNNING	ip-172-31-13-181.us-west-2.compute.internal:8042	
0			
ip-172-31-4-190.us-west-2.compute.internal:43269	RUNNING	ip-172-31-4-190.us-west-2.compute.internal:8042	
0			
ip-172-31-11-64.us-west-2.compute.internal:46698	RUNNING	ip-172-31-11-64.us-west-2.compute.internal:8042	
0			
ip-172-31-11-46.us-west-2.compute.internal:55928	RUNNING	ip-172-31-11-46.us-west-2.compute.internal:8042	
0			
ip-172-31-9-196.us-west-2.compute.internal:43081	RUNNING	ip-172-31-9-196.us-west-2.compute.internal:8042	
0			
ip-172-31-0-15.us-west-2.compute.internal:45834	RUNNING	ip-172-31-0-15.us-west-2.compute.internal:8042	0
ip-172-31-6-168.us-west-2.compute.internal:49151	RUNNING	ip-172-31-6-168.us-west-2.compute.internal:8042	
0			

EC2 Management Console - Mozilla Firefox

EC2 Management C... My Drive - Google Dr... + https://us-west-2.console.aws.amazon.com/ec2/v2/home?region=us-west-2#Instances:sort=instance

Search George Mathew Oregon Support

Launch Instance Connect Actions

Filter by tags and attributes or search by keyword

1 to 17 of 17

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS	Public IP
slave1	i-080a0fd2	c3.large	us-west-2c	running	2/2 checks ...	None	ec2-54-187-79-102.us-w...	54.187.79.102
	i-1c7175c6	c3.large	us-west-2c	running	2/2 checks ...	None	ec2-54-187-70-116.us-w...	54.187.70.116
	i-4470749e	c3.large	us-west-2c	running	2/2 checks ...	None	ec2-54-186-240-34.us-w...	54.186.240.34
	i-5171758b	c3.large	us-west-2c	running	2/2 checks ...	None	ec2-54-213-117-209.us...	54.213.117.209
	i-5271758b	c3.large	us-west-2c	running	2/2 checks ...	None	ec2-54-186-222-112.us...	54.186.222.112
	i-5371758b	c3.large	us-west-2c	running	2/2 checks ...	None	ec2-54-213-129-172.us...	54.213.129.172
	i-5771758d	c3.large	us-west-2c	running	2/2 checks ...	None	ec2-54-187-21-37.us-w...	54.187.21.37
	i-5c71758b	c3.large	us-west-2c	running	2/2 checks ...	None	ec2-54-213-137-38.us-w...	54.213.137.38
master	i-5f000585	d2.xlarge	us-west-2c	running	2/2 checks ...	None	ec2-54-186-224-237.us...	54.186.224.237
slave 2	i-62eeebb8	c3.large	us-west-2c	running	2/2 checks ...	None	ec2-54-191-20-207.us-w...	54.191.20.207

Instance: i-d44d490e (slave 3) Public DNS: ec2-54-187-36-117.us-west-2.compute.amazonaws.com

Description Status Checks Monitoring Tags

Instance ID: i-d44d490e Public DNS: ec2-54-187-36-117.us-west-2.compute.amazonaws.com

Feedback English © 2008 - 2016, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

EC2 Management Console - Mozilla Firefox

EC2 Management C... My Drive - Google Dr... + https://us-west-2.console.aws.amazon.com/ec2/v2/home?region=us-west-2#Instances:sort=instance

Search George Mathew Oregon Support

Launch Instance Connect Actions

Filter by tags and attributes or search by keyword

1 to 17 of 17

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS	Public IP
master	i-5f000585	d2.xlarge	us-west-2c	running	2/2 checks ...	None	ec2-54-186-224-237.us...	54.186.224.237
slave 2	i-62eeebb8	c3.large	us-west-2c	running	2/2 checks ...	None	ec2-54-191-20-207.us-w...	54.191.20.207
	i-a8707472	c3.large	us-west-2c	running	2/2 checks ...	None	ec2-54-213-35-107.us-w...	54.213.35.107
	i-a9707473	c3.large	us-west-2c	running	2/2 checks ...	None	ec2-54-213-139-32.us-w...	54.213.139.32
	i-aa707470	c3.large	us-west-2c	running	2/2 checks ...	None	ec2-54-200-88-149.us-w...	54.200.88.149
slave 3	i-d44d490e	c3.large	us-west-2c	running	2/2 checks ...	None	ec2-54-187-36-117.us-w...	54.187.36.117
	i-db707401	c3.large	us-west-2c	running	2/2 checks ...	None	ec2-54-213-128-234.us...	54.213.128.234
	i-de777304	c3.large	us-west-2c	running	2/2 checks ...	None	ec2-54-187-108-162.us...	54.187.108.162
	i-f2707428	c3.large	us-west-2c	running	2/2 checks ...	None	ec2-54-213-130-155.us...	54.213.130.155

Instance: i-d44d490e (slave 3) Public DNS: ec2-54-187-36-117.us-west-2.compute.amazonaws.com

Description Status Checks Monitoring Tags

Instance ID: i-d44d490e Public DNS: ec2-54-187-36-117.us-west-2.compute.amazonaws.com

Feedback English © 2008 - 2016, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

About the Cluster - Mozilla Firefox

The screenshot shows the Hadoop cluster management interface. It features a sidebar with cluster navigation options like 'About Nodes', 'Node Labels', 'Applications', and 'Scheduler'. The main area displays 'Cluster Metrics' and 'Scheduler Metrics'. Under 'Cluster Metrics', there are tables for 'Apps Submitted', 'Apps Pending', 'Apps Running', 'Apps Completed', 'Containers Running', 'Memory Used', 'Memory Total', 'Memory Reserved', 'VCores Total', 'VCores Used', 'VCores Reserved', 'Active Nodes', 'Decommissioned Nodes', 'Lost Nodes', 'Unhealthy Nodes', and 'Rebooted Nodes'. The 'Schedulers Metrics' table includes columns for 'Capacity Scheduler', 'Scheduling Resource Type', 'Minimum Allocation', and 'Maximum Allocation'. Specific cluster details are listed below, such as Cluster ID (1459226183235), ResourceManager state (STARTED), and ResourceManager HA state (active).

What is a master Node and Slave Node?

The resource manager and the Node manager runs on the manager and the keep tracks of which data goes to which node. The nodes that the master use to process the data are called the slave nodes. The master uses the resources of the slave nodes to do the processing

Why do we need to set unique available ports to those configuration files on a shared environment?

What errors or side-effects will show if we use same port number for each user?

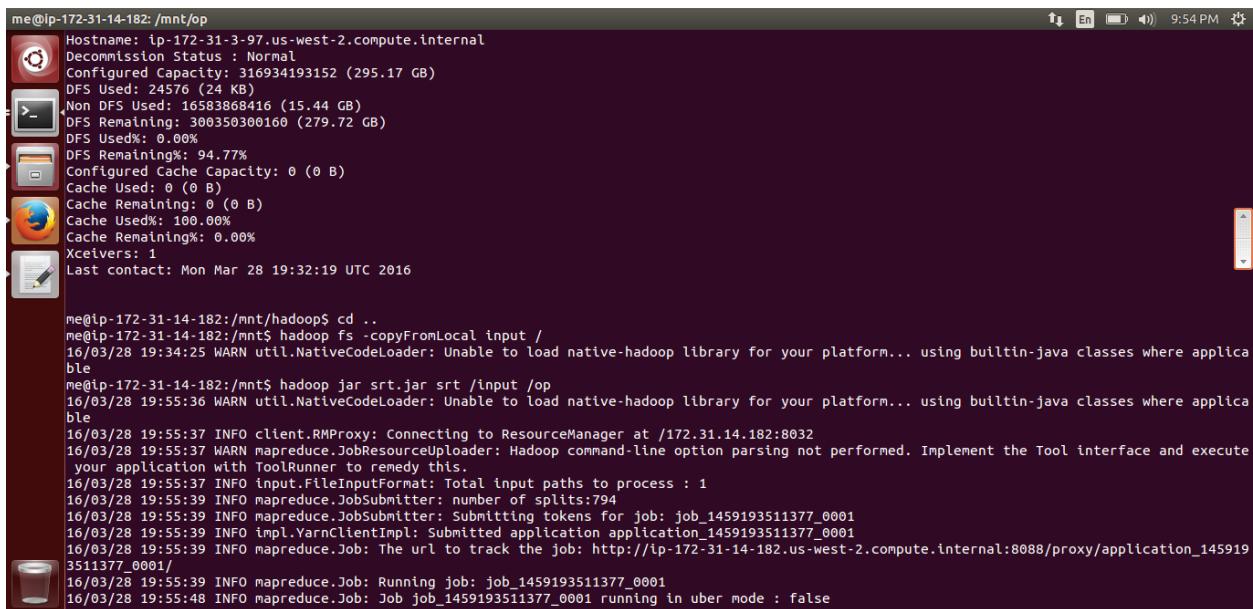
Only one process can run on a single port. If we give the same port number to two different processes, then there will be conflict and the process will fail

How can we change the number of mappers and reducers from the configuration file?

The number of mappers and reducers can be changed in the map-reduce .xml config file. As mentioned above my cluster was running with two mappers and two reducers as there are two virtual cores in c3.large

```
<property>
<name>mapred.tasktracker.map.tasks</name>
<value>2</value>
</property>
<property>
<name>mapred.tasktracker.reduce.tasks</name>
<value>2</value>
</property>
```

The Hadoop program was run to sort 100gb data



me@ip-172-31-14-182:/mnt/op

```
Hostname: ip-172-31-3-97.us-west-2.compute.internal
Decommission Status : Normal
Configured Capacity: 316934193152 (295.17 GB)
DFS Used: 24576 (24 KB)
Non DFS Used: 16583868416 (15.44 GB)
DFS Remaining: 300350300160 (279.72 GB)
DFS Used%: 0.00%
DFS Remaining%: 94.77%
Configured Cache Capacity: 0 (0 B)
Cache Used: 0 (0 B)
Cache Remaining: 0 (0 B)
Cache Used%: 100.00%
Cache Remaining%: 0.00%
Xceivers: 1
Last contact: Mon Mar 28 19:32:19 UTC 2016
```

```
me@ip-172-31-14-182:/mnt/hadoop$ cd ..
me@ip-172-31-14-182:/mnt$ hadoop fs -copyFromLocal input /
16/03/28 19:34:25 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
me@ip-172-31-14-182:/mnt$ hadoop jar srt.jar srt /input /op
16/03/28 19:55:36 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
16/03/28 19:55:37 INFO client.RMProxy: Connecting to ResourceManager at /172.31.14.182:8032
16/03/28 19:55:37 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this
16/03/28 19:55:37 INFO input.FileInputFormat: Total input paths to process : 1
16/03/28 19:55:39 INFO mapreduce.JobSubmitter: number of splits:794
16/03/28 19:55:39 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1459193511377_0001
16/03/28 19:55:39 INFO impl.YarnClientImpl: Submitted application application_1459193511377_0001
16/03/28 19:55:39 INFO mapreduce.Job: The url to track the job: http://ip-172-31-14-182.us-west-2.compute.internal:8088/proxy/application_1459193511377_0001/
16/03/28 19:55:39 INFO mapreduce.Job: Running job: job_1459193511377_0001
16/03/28 19:55:48 INFO mapreduce.Job: Job job_1459193511377_0001 running in uber mode : false
```

```

me@ip-172-31-14-182: /mnt/op
      FILE: Number of bytes written=546323886940
      FILE: Number of read operations=0
      FILE: Number of large read operations=0
      FILE: Number of write operations=0
      HDFS: Number of bytes read=106503324352
      HDFS: Number of bytes written=1065000000000
      HDFS: Number of read operations=2385
      HDFS: Number of large read operations=0
      HDFS: Number of write operations=2
Job Counters
      Killed map tasks=9
      Launched map tasks=803
      Launched reduce tasks=1
      Data-local map tasks=722
      Rack-local map tasks=81
      Total time spent by all maps in occupied slots (ms)=118110674
      Total time spent by all reduces in occupied slots (ms)=20951979
      Total time spent by all map tasks (ms)=118110674
      Total time spent by all reduce tasks (ms)=20951979
      Total vcore-seconds taken by all map tasks=118110674
      Total vcore-seconds taken by all reduce tasks=20951979
      Total negabyte-seconds taken by all map tasks=120945330176
      Total negabyte-seconds taken by all reduce tasks=21454826496
Map-Reduce Framework
      Map input records=1065000000
      Map output records=1065000000
      Map output bytes=106500000000
      Map output materialized bytes=108630004764
      Input split bytes=76224
      Combine input records=1065000000
      Combine output records=1065000000
      Reduce input groups=1065000000
      Reduce shuffle bytes=108630004764
      Reduce input records=1065000000
      Reduce output records=1065000000

```

This was completed in 4.39 hours

```

me@ip-172-31-14-182: /mnt/op
      Reduce output records=1065000000
      Spilled Records=5355216650
      Shuffled Maps =794
      Failed Shuffles=0
      Merged Map outputs=794
      GC time elapsed (ms)=1058890
      CPU time spent (ms)=15836660
      Physical memory (bytes) snapshot=206761775104
      Virtual memory (bytes) snapshot=658464784384
      Total committed heap usage (bytes)=162279718912
Shuffle Errors
      BAD_ID=0
      CONNECTION=0
      IO_ERROR=0
      WRONG_LENGTH=0
      WRONG_MAP=0
      WRONG_REDUCE=0
File Input Format Counters
      Bytes Read=106503248128
File Output Format Counters
      Bytes Written=106500000000
me@ip-172-31-14-182:/mnt$ hadoop fs -copyToLocal /op /mnt
16/03/29 01:51:25 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
16/03/29 01:51:25 WARN hdfs.DFSClient: DFSInputStream has been closed already
16/03/29 02:20:27 WARN hdfs.DFSClient: DFSInputStream has been closed already
me@ip-172-31-14-182:/mnt$ cp valsrt op/
me@ip-172-31-14-182:/mnt$ cd op
me@ip-172-31-14-182:/mnt/op$ ls
part-r-00000  _SUCCESS  valsrt
me@ip-172-31-14-182:/mnt/op$ ./valsrt part-r-00000
Records: 1065000000
Checksum: 1fbdb3cc9e3fd72c
Duplicate keys: 0
_SUCCESS - all records are in order

```

Valsort was done and the output was verified

6) Spark

Spark was installed on 17 systems.

1 c3.4large system was used as a master and all other systems were c3.large

The program to sort the entries was written using python

This program was used to sort 10gb in pseudo mode and 100gb in 16 clusters

7)Spark Sort

Pseudo Mode

10 gb of data was sorted using the program on a single node spark system

The screenshot shows the AWS EC2 Management Console interface. On the left, there's a sidebar with various AWS services like EC2 Dashboard, Events, Tags, Reports, Limits, Instances, Images, Elastic Block Store, and Network & Security. The 'Instances' section is currently selected.

The main pane displays a table of instances. One instance, 'pseudoSort-master-i-6e1621a9', is highlighted and shown in a detailed view below the table. This instance is a c3.large type, running in the us-west-2c availability zone, and has a status of 'running'. It has a Public DNS of 'ec2-54-186-159-147.us-west-2.amazonaws.com' and a Private IP of '172.31.24.63'. The detailed view also shows its VPC ID ('vpc-349f8051') and Subnet ID ('subnet-76e90a12').

At the bottom of the page, there are links for Feedback, English, Privacy Policy, and Terms of Use.

Detailed View of the Selected Instance:

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS
pseudoSort-master-i-6e1621a9	i-6e1621a9	c3.large	us-west-2a	running	2/2 checks ...	None	ec2-54-186-159-147.us-west-2.amazonaws.com

Instance Details:

Private DNS	Private IPs	Secondary private IPs	VPC ID	Subnet ID	Availability zone	Security groups	Scheduled events	AMI ID	Platform
ip-172-31-24-63.us-west-2.compute.internal	172.31.24.63		vpc-349f8051	subnet-76e90a12	us-west-2a	pseudoSort-master, view rules	No scheduled events	spark.ami.pvm.v9 (ami-9a6e0daa)	

Spark Master at spark://ec2-54-186-159-147.us-west-2.compute.amazonaws.com:7077 - Mozilla Firefox

EC2 Management C... Sort Benchmark Dat... Spark Master at spark:/... +

ec2-54-186-159-147.us-west-2.compute.amazonaws.com:8080 Search

Spark 1.6.1

Spark Master at spark://ec2-54-186-159-147.us-west-2.compute.amazonaws.com:7077

URL: spark://ec2-54-186-159-147.us-west-2.compute.amazonaws.com:7077
REST URL: spark://ec2-54-186-159-147.us-west-2.compute.amazonaws.com:6066 (cluster mode)

Alive Workers: 1
Cores in use: 2 Total, 0 Used
Memory in use: 2.7 GB Total, 0.0 B Used
Applications: 0 Running, 0 Completed
Drivers: 0 Running, 0 Completed
Status: ALIVE

Workers

Worker Id	Address	State	Cores	Memory
worker-20160331153822-172.31.18.130-43219	172.31.18.130:43219	ALIVE	2 (0 Used)	2.7 GB (0.0 B Used)

Running Applications

Application ID	Name	Cores	Memory per Node	Submitted Time	User	State	Duration

Completed Applications

Application ID	Name	Cores	Memory per Node	Submitted Time	User	State	Duration

Spark Master at spark://ec2-54-186-159-147.us-west-2.compute.amazonaws.com:7077 - Mozilla Firefox

EC2 Management C... Sort Benchmark Dat... Spark Master at spark:/... +

ec2-54-186-159-147.us-west-2.compute.amazonaws.com:8080 Search

Spark 1.6.1

Spark Master at spark://ec2-54-186-159-147.us-west-2.compute.amazonaws.com:7077

URL: spark://ec2-54-186-159-147.us-west-2.compute.amazonaws.com:7077
REST URL: spark://ec2-54-186-159-147.us-west-2.compute.amazonaws.com:6066 (cluster mode)

Alive Workers: 1
Cores in use: 2 Total, 2 Used
Memory in use: 2.7 GB Total, 2.4 GB Used
Applications: 1 Running, 0 Completed
Drivers: 0 Running, 0 Completed
Status: ALIVE

Workers

Worker Id	Address	State	Cores	Memory
worker-20160331153822-172.31.18.130-43219	172.31.18.130:43219	ALIVE	2 (2 Used)	2.7 GB (2.4 GB Used)

Running Applications

Application ID	Name	Cores	Memory per Node	Submitted Time	User	State	Duration
app-20160331160106-0000	(kill) Sort	2	2.4 GB	2016/03/31 16:01:06	root	RUNNING	8 s

Completed Applications

Application ID	Name	Cores	Memory per Node	Submitted Time	User	State	Duration

Sort - Spark Jobs - Mozilla Firefox

EC2 Management C... Sort Benchmark Dat... Sort - Spark Jobs

ec2-54-186-159-147.us-west-2.compute.amazonaws.com:4040/jobs/

Spark 1.6.1 Jobs Stages Storage Environment Executors Sort application UI

Spark Jobs (?)

Total Uptime: 11 min
Scheduling Mode: FIFO
Active Jobs: 1
Completed Jobs: 1

Event Timeline

Active Jobs (1)

Job Id	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
1	saveAsTextFile at SimpleApp.scala:11	2016/03/31 16:02:14	10 min	1/2	128/160

Completed Jobs (1)

Job Id	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
0	sortByKey at SimpleApp.scala:11	2016/03/31 16:01:08	1.1 min	1/1	80/80

Spark Master at spark://ec2-54-186-159-147.us-west-2.compute.amazonaws.com:7077 - Mozilla Firefox

EC2 Management C... Sort Benchmark Dat... Spark Master at spark://... +

ec2-54-186-159-147.us-west-2.compute.amazonaws.com:8080

Search

Spark Master at spark://ec2-54-186-159-147.us-west-2.compute.amazonaws.com:7077

URL: spark://ec2-54-186-159-147.us-west-2.compute.amazonaws.com:7077
REST URL: spark://ec2-54-186-159-147.us-west-2.compute.amazonaws.com:6066 (cluster mode)
Alive Workers: 1
Cores in use: 2 Total, 0 Used
Memory in use: 2.7 GB Total, 0.0 B Used
Applications: 0 Running, 1 Completed
Drivers: 0 Running, 0 Completed
Status: ALIVE

Workers

Worker Id	Address	State	Cores	Memory
worker-20160331153822-172.31.18.130-43219	172.31.18.130:43219	ALIVE	2 (0 Used)	2.7 GB (0.0 B Used)

Running Applications

Application ID	Name	Cores	Memory per Node	Submitted Time	User	State	Duration
----------------	------	-------	-----------------	----------------	------	-------	----------

Completed Applications

Application ID	Name	Cores	Memory per Node	Submitted Time	User	State	Duration
app-20160331160106-0000	Sort	2	2.4 GB	2016/03/31 16:01:06	root	FINISHED	15 min

The sorting was completed in 15 mins

The output was verified using `valsrt`

Cluster Sort

The cluster was setup with 17 instances 1 c4.large as master and 16 c3.large instances as slaves were initiated

The screenshot shows the AWS EC2 Management Console in Mozilla Firefox. The left sidebar is collapsed, showing the following navigation menu:

- AWS
- Services
- Edit

Below the sidebar, the main content area displays a list of EC2 instances. The top navigation bar includes tabs for "Launch Instance", "Connect", and "Actions". A search bar at the top right allows filtering by tags and attributes or searching by keyword.

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS
SparkSort-slave-i-387a3fe0	i-387a3fe0	c3.large	us-west-2b	running	✓ 2/2 checks ...	None	ec2-54-187-227-51.us-w...
SparkSort-slave-i-d47a3f0c	i-d47a3f0c	c3.large	us-west-2b	running	✓ 2/2 checks ...	None	ec2-54-187-208-70.us-w...
SparkSort-slave-i-5085c088	i-5085c088	c3.large	us-west-2b	running	✓ 2/2 checks ...	None	ec2-54-218-66-155.us-w...
SparkSort-slave-i-8cb7b3e54	i-8cb7b3e54	c3.large	us-west-2b	running	✓ 2/2 checks ...	None	ec2-54-191-129-217.us...
SparkSort-slave-i-807b3e58	i-807b3e58	c3.large	us-west-2b	running	✓ 2/2 checks ...	None	ec2-54-201-239-100.us...
SparkSort-slave-i-397a3fe1	i-397a3fe1	c3.large	us-west-2b	running	✓ 2/2 checks ...	None	ec2-54-218-66-42.us-w...
SparkSort-slave-i-8d7b3e55	i-8d7b3e55	c3.large	us-west-2b	running	✓ 2/2 checks ...	None	ec2-54-191-164-228.us...
SparkSort-slave-i-bf7a3f67	i-bf7a3f67	c3.large	us-west-2b	running	✓ 2/2 checks ...	None	ec2-54-213-218-118.us...
SparkSort-master-i-1b7a3fc3	i-1b7a3fc3	c3.4xlarge	us-west-2b	running	✓ 2/2 checks ...	None	ec2-54-200-213-60.us-w...

At the bottom of the instance list, there are detailed information cards for the selected master instance:

- Instance state: running
- Instance type: c3.4xlarge
- Private DNS: ip-172-31-46-206.us-west-2.compute.internal
- Private IPs: 172.31.46.206
- Secondary private IPs: (empty)
- Public IP: west-2.compute.amazonaws.com
- Public IP: 54.200.213.60
- Elastic IP: -
- Availability zone: us-west-2b
- Security groups: SparkSort-master, view rules
- Scheduled events: No scheduled events

The program was run on this cluster to sort 100gb data

Spark Master at spark://ec2-54-200-213-60.us-west-2.compute.amazonaws.com:7077 - Mozilla Firefox

EC2 Management C... x Spark Master at spark://... x +

ec2-54-200-213-60.us-west-2.compute.amazonaws.com:8080

			Search	☆	自	下載	上傳	關閉
worker-20160331222331-172.31.34.67-44294	172.31.34.67:44294	ALIVE	2 (2 Used)	2.7 GB (2.4 GB Used)				
worker-20160331222331-172.31.36.209-47611	172.31.36.209:47611	ALIVE	2 (2 Used)	2.7 GB (2.4 GB Used)				
worker-20160331222331-172.31.36.229-35369	172.31.36.229:35369	ALIVE	2 (2 Used)	2.7 GB (2.4 GB Used)				
worker-20160331222331-172.31.36.243-46162	172.31.36.243:46162	ALIVE	2 (2 Used)	2.7 GB (2.4 GB Used)				
worker-20160331222331-172.31.41.139-60916	172.31.41.139:60916	ALIVE	2 (2 Used)	2.7 GB (2.4 GB Used)				
worker-20160331222331-172.31.41.174-51156	172.31.41.174:51156	ALIVE	2 (2 Used)	2.7 GB (2.4 GB Used)				
worker-20160331222331-172.31.43.0-38645	172.31.43.0:38645	ALIVE	2 (2 Used)	2.7 GB (2.4 GB Used)				
worker-20160331222331-172.31.43.230-40649	172.31.43.230:40649	ALIVE	2 (2 Used)	2.7 GB (2.4 GB Used)				
worker-20160331222331-172.31.45.253-58895	172.31.45.253:58895	ALIVE	2 (2 Used)	2.7 GB (2.4 GB Used)				
worker-20160331222331-172.31.46.120-47883	172.31.46.120:47883	ALIVE	2 (2 Used)	2.7 GB (2.4 GB Used)				
worker-20160331222331-172.31.47.17-55045	172.31.47.17:55045	ALIVE	2 (2 Used)	2.7 GB (2.4 GB Used)				
worker-20160331222332-172.31.37.18-34894	172.31.37.18:34894	ALIVE	2 (2 Used)	2.7 GB (2.4 GB Used)				

Running Applications

Application ID	Name	Cores	Memory per Node	Submitted Time	User	State	Duration
app-20160331232128-0000	(kill) Sort	32	2.4 GB	2016/03/31 23:21:28	root	RUNNING	7 s

Completed Applications

Application ID	Name	Cores	Memory per Node	Submitted Time	User	State	Duration
----------------	------	-------	-----------------	----------------	------	-------	----------

Sort - Spark Jobs - Mozilla Firefox

EC2 Management C... x Sort - Spark Jobs x +

ec2-54-200-213-60.us-west-2.compute.amazonaws.com:4040/jobs/

Spark 1.6.1 Jobs Stages Storage Environment Executors Sort application UI

Spark Jobs (?)

Total Uptime: 6.3 min
 Scheduling Mode: FIFO
Active Jobs: 1
Completed Jobs: 1
[Event Timeline](#)

Active Jobs (1)

Job Id	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
1	saveAsTextFile at SimpleApp.scala:11	2016/03/31 23:22:26	5.3 min	1/2	826/1588

Completed Jobs (1)

Job Id	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
0	sortByKey at SimpleApp.scala:11	2016/03/31 23:21:29	57 s	1/1	794/794

The screenshot shows the Spark Master UI running on port 7077. The main table lists 14 worker nodes, all of which are ALIVE. The completed applications table shows one entry: app-20160331232128-0000, which was submitted by root and finished 11 minutes ago.

The sorting was completed in 11 minutes

```

george@george-VirtualBox: ~/Downloads/spark-1.6.1-bin-hadoop2.6/ec2
part-0048 part-00115 part-00182 part-00249 part-00316 part-00383 part-00450 part-00517 part-00584 part-00651 part-00718 part-00785
part-0049 part-00116 part-00183 part-00250 part-00317 part-00384 part-00451 part-00518 part-00585 part-00652 part-00719 part-00786
part-0050 part-00117 part-00184 part-00251 part-00318 part-00385 part-00452 part-00519 part-00586 part-00653 part-00720 part-00787
part-0051 part-00118 part-00185 part-00252 part-00319 part-00386 part-00453 part-00520 part-00587 part-00654 part-00721 part-00788
part-0052 part-00119 part-00186 part-00253 part-00320 part-00387 part-00454 part-00521 part-00588 part-00655 part-00722 part-00789
part-0053 part-00120 part-00187 part-00254 part-00321 part-00388 part-00455 part-00522 part-00589 part-00656 part-00723 part-00790
part-0054 part-00121 part-00188 part-00255 part-00322 part-00389 part-00456 part-00523 part-00590 part-00657 part-00724 part-00791
part-0055 part-00122 part-00189 part-00256 part-00323 part-00390 part-00457 part-00524 part-00591 part-00658 part-00725 part-00792
part-0056 part-00123 part-00190 part-00257 part-00324 part-00391 part-00458 part-00525 part-00592 part-00659 part-00726 part-00793
part-0057 part-00124 part-00191 part-00258 part-00325 part-00392 part-00459 part-00526 part-00593 part-00660 part-00727 _SUCCESS
part-0058 part-00125 part-00192 part-00259 part-00326 part-00393 part-00460 part-00527 part-00594 part-00661 part-00728
part-0059 part-00126 part-00193 part-00260 part-00327 part-00394 part-00461 part-00528 part-00595 part-00662 part-00729
part-00600 part-00127 part-00194 part-00261 part-00328 part-00395 part-00462 part-00529 part-00596 part-00663 part-00730
part-00601 part-00128 part-00195 part-00262 part-00329 part-00396 part-00463 part-00530 part-00597 part-00664 part-00731
part-00602 part-00129 part-00196 part-00263 part-00330 part-00397 part-00464 part-00531 part-00598 part-00665 part-00732
part-00603 part-00130 part-00197 part-00264 part-00331 part-00398 part-00465 part-00532 part-00599 part-00666 part-00733
part-00604 part-00131 part-00198 part-00265 part-00332 part-00399 part-00466 part-00533 part-00600 part-00667 part-00734
part-00605 part-00132 part-00199 part-00266 part-00333 part-00400 part-00467 part-00534 part-00601 part-00668 part-00735
part-00606 part-00133 part-00200 part-00267 part-00334 part-00401 part-00468 part-00535 part-00602 part-00669 part-00736
root@ip-172-31-46-206 output]$ /root/myfiles/64/./valsrt part-00000
Records: 1114192
Checksum: 8819bcacd59aa
Duplicate keys: 0
SUCCESS - all records are in order
root@ip-172-31-46-206 output]$ /root/myfiles/64/./valsrt part-00793
Records: 1472734
Checksum: b3acfcc6d607d1
Duplicate keys: 0
SUCCESS - all records are in order
root@ip-172-31-46-206 output]$ cd ..
root@ip-172-31-46-206 mnt]$ du -h output
du: cannot access `output': No such file or directory
root@ip-172-31-46-206 mnt]$ du -h output
160G    output
root@ip-172-31-46-206 mnt]$ 

```

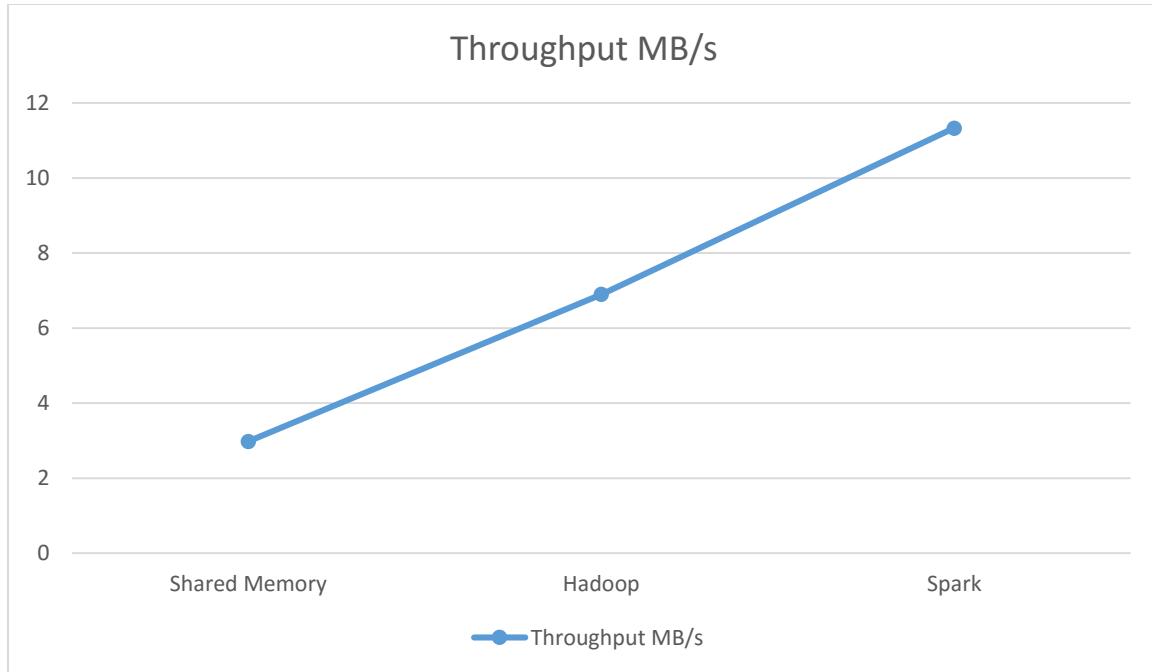
The output was verified using valsrt

8) Performance

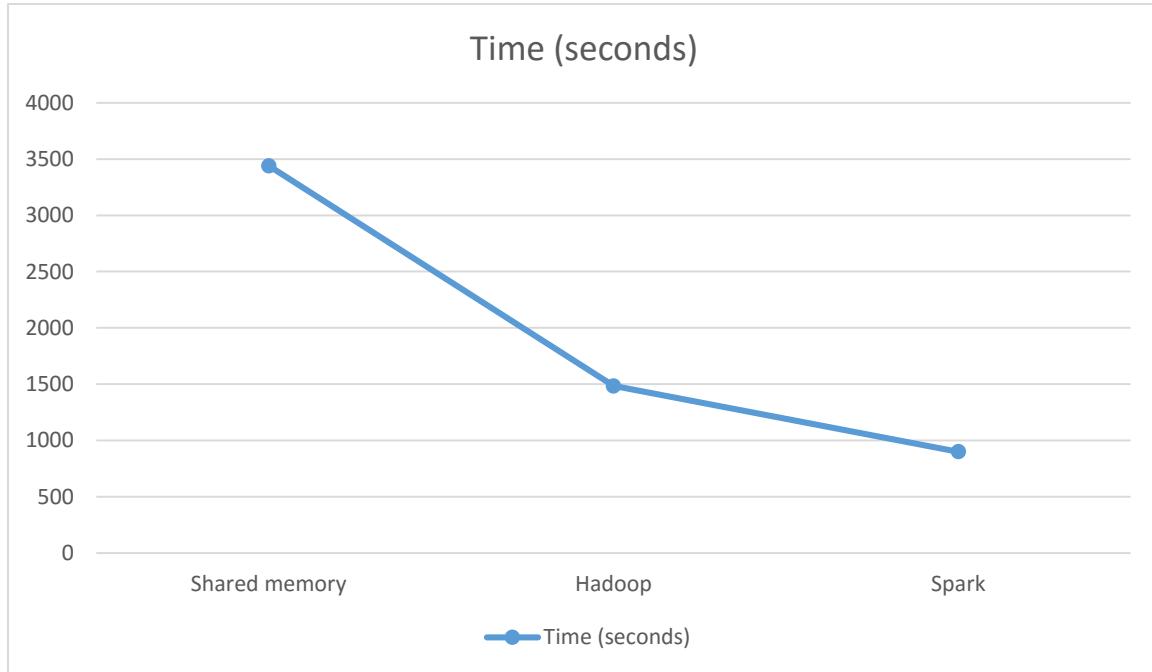
1 node performance

10GB sorts

Name	Time	Throughput
Shared Memory	3440 s	2.98 MB/s
Hadoop	1484 s	6.90 MB/s
Spark	900 s	11.33 MB/s



Graph showing increased throughput for Hadoop and spark as compared to shared memory sort

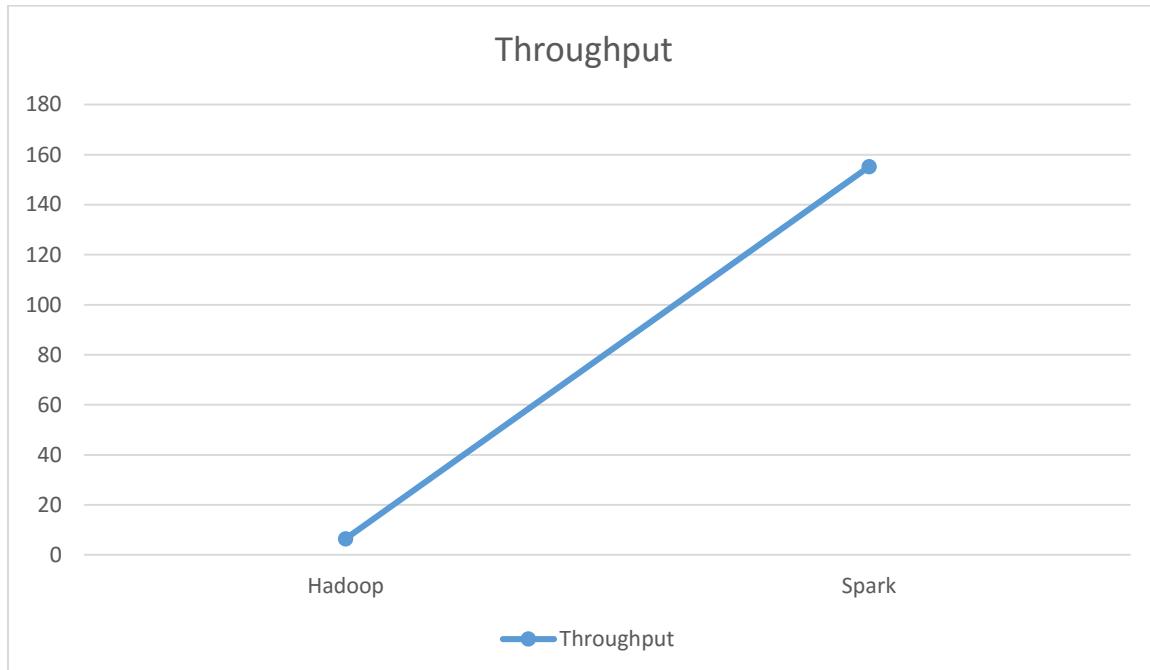


Graph showing Decreased Time of execution as compared to shared memory

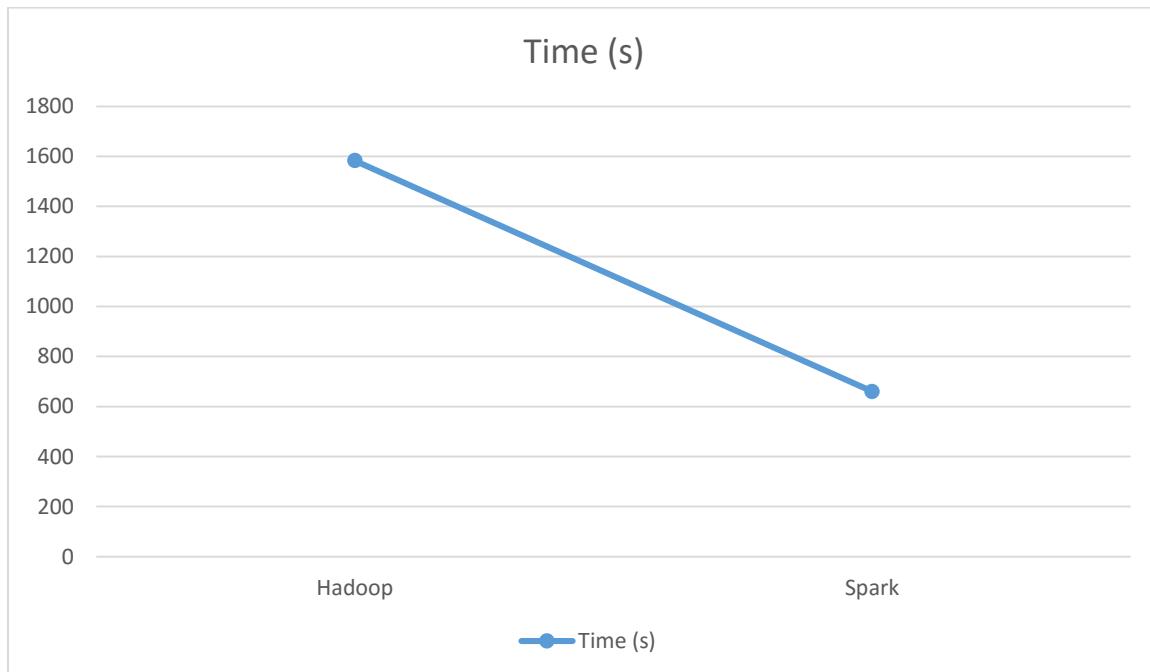
16 node performance

100GB sorts

Name	Time	Throughput
Hadoop	15836s	6.46 MB/s
Spark	660 s	155.15 MB/s



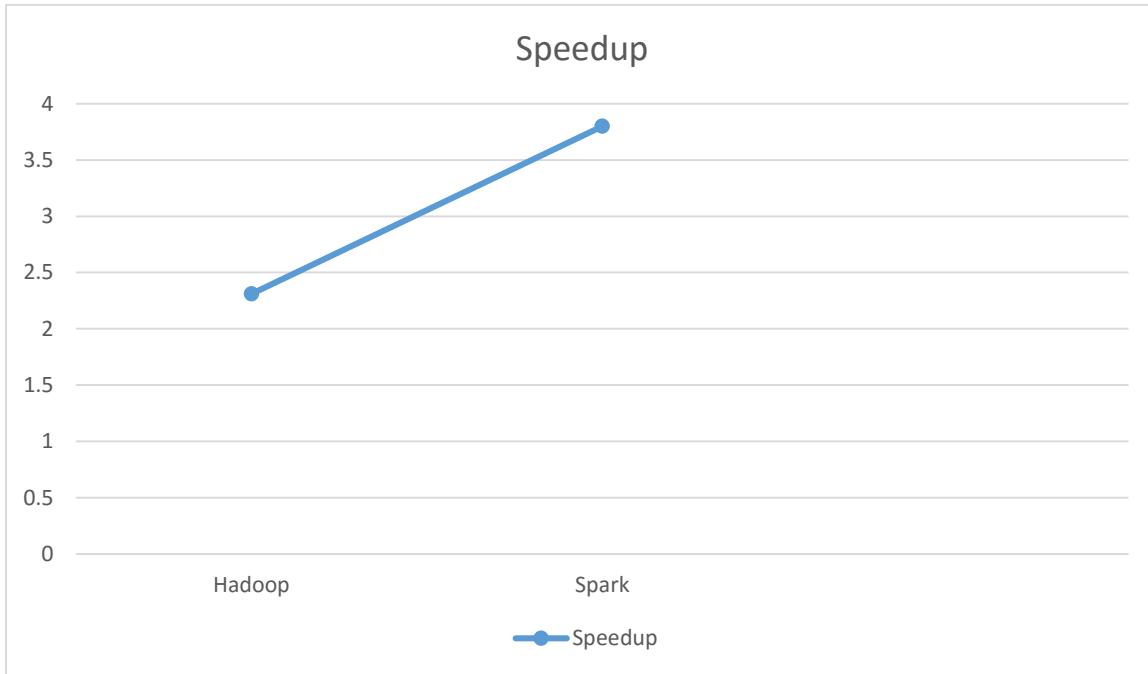
Graph showing the increased throughput of spark compared to Hadoop



Graph showing the decrease in time of execution of Spark as compared to Hadoop

Speed up Shared Memory

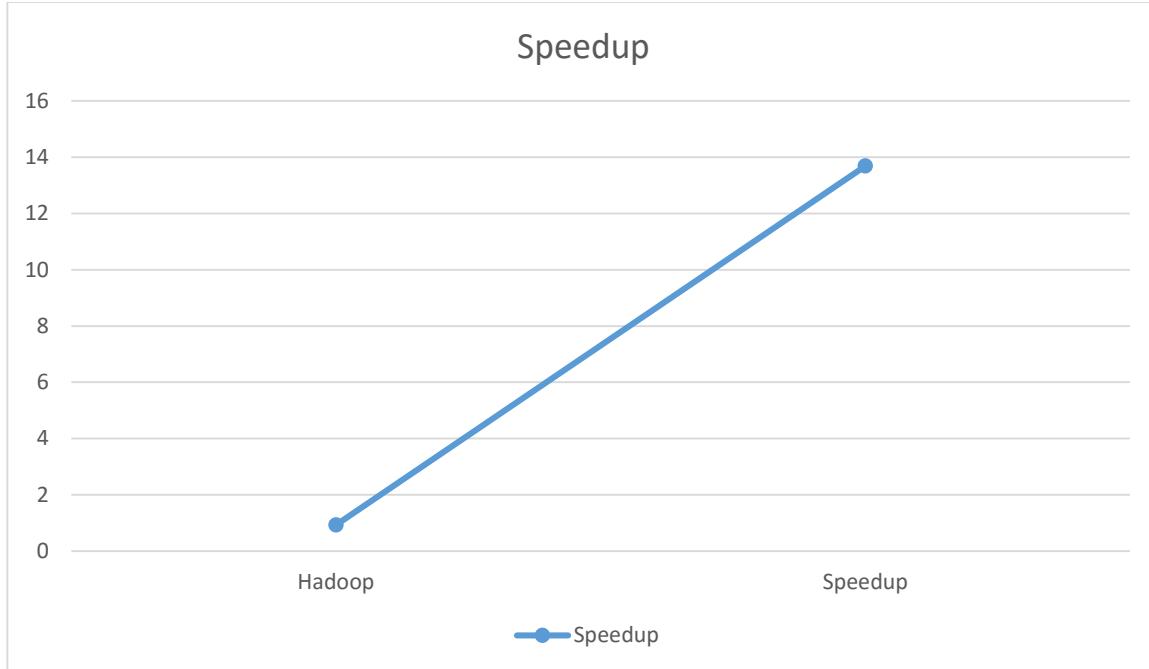
Name	Throughput (1 node)	Speed Up
Shared Memory	2.98 MB/s	
Hadoop	6.90 MB/s	2.31
Spark	11.33 MB/s	3.80



Graph showing the high increase in speedup of spark as compared to Hadoop and shared memory

Speedup Spark and Speedup Hadoop

Name	Throughput (1 node)	Throughput (16 node)	Speed Up
Hadoop	6.90 MB/s	6.46 MB/s	0.93
Spark	11.33 MB/s	155.15 MB/s	13.69



Graph showing the increased speed up of spark as compared to Hadoop

It can be concluded that Hadoop has more throughput than the shared memory sort and is better than normal programming practices

Spark is definitely an improvement over Hadoop in throughput and the speed up attained when more nodes are added

9)TeraSort

Terasort was attempted for shared memory. It was taking too much time to execute so it was not completed to save money The completed code has been submitted with the assignment

10)Conclusions

Environment Details

OS: Ubuntu Server 14.04 LTS (HVM), SSD Volume Type - ami-9abea4fb 64 bit

Java: 1.7.0_95

Hadoop: 2.7.2

Spark: Spark 1.6.1

Difficulties Faced:

-Shared Memory:

The external sort algorithm was tricky to implement

-Hadoop

Pseudo Mode was easy but setting up a cluster was hard especially because lot of repetitive tasks were there

-Spark

Spark environment was easy to setup but the program was hard to write mainly because I decided to do it in python and I was new to python

It was definitely an experience to do this assignment. The challenges involved were huge as the learning curve was very high. But in the end it felt nice to have done a sorting over data bigger than memory, setting up clusters and writing codes on that