

CS 553:Cloud Computing Assignment-2

Name: Priyank Shah

CWID:A20344797

1. Design Report

In Assignment-2 following are the parts that I have implemented:

- Setting up Virtual Cluster of c3.large contains 1 master and 1 slave, mounting EBS volume on master and slave node.
- Setting up Virtual Cluster of c3.large contains 1 master and 16 slave, mounting EBS volume on master and slave node.
- Implementing Hadoop Sort for 1 node and 16 nodes.
- Implementing Spark Sort for 1 node and 16 nodes.
- Implementing Shared Memory sort for Single Thread and Multiple thread.
- Comparison between Shared memory, Hadoop and Spark on 1 Node.
- Measuring Speed up and Performance evaluation of 1 node and 16 node cluster.

In Hadoop, I have generated input with gensort for 10 GB and 100 GB and run them on 1 and 16 nodes respectively. I have implemented the program in java. For Hadoop you need to generate jar file from java file, add class file to jar. Put generated data on hadoop file system and run the program with jar file. Get the output from hadoop file system to local file system and check with val sort for the sake of sorted output validity.

In Spark, I have generated input with gensort for 10 GB and 100 GB and run them on 1 and 16 nodes respectively. I have implemented the program in scala, which can be run in almost similar fashion as java. Put the scala file in the same folder where spark-shell resides. Open the spark-shell and run the program, it will generate the output.

In Shared Memory, I have generated input with gensort for 10 GB. I am taking user input for threads. Once program start, make sure to have enough space for intermediate files(temporary files). It will cut data in 20 MB Chunks. And perform quick sort. On temporary generated file it will imply merge sort and builds single output file.

Configuration and Installation:

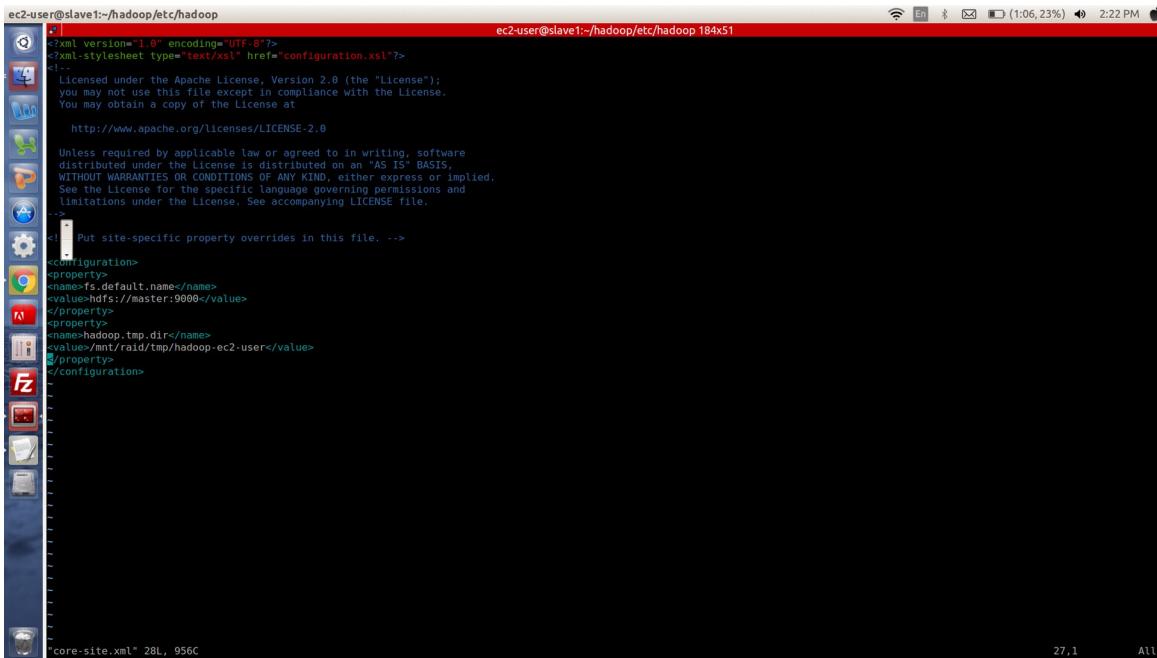
1. Hadoop:

Instance Type: c3.large
 RAM: 3.75 GB
 # of Instance: 1(for 10 GB) and 17(1 master and 16 slave)
 Storage: 32 GB
 OS: Amazon Linux AMI
 Java version: 1.7.0
 Hadoop Version: 2.7.2

Following are Files Changes which resides in /hadoop/etc/hadoop:

1. core-site.xml:

```
<configuration>
<property>
<name>fs.default.name</name>
<value>hdfs://master:9000</value>
</property>
<property>
<name>hadoop.tmp.dir</name>
<value>/mnt/raid/tmp/hadoop-ec2-user</value>
</property>
</configuration>
```



```
ec2-user@slave1:~/hadoop/etc/hadoop
[ec2-user@slave1:~/hadoop/etc/hadoop]$ cat core-site.xml
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!-- Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at
http://www.apache.org/licenses/LICENSE-2.0
Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License. See accompanying LICENSE file.
-->
<!-- Put site-specific property overrides in this file. -->
<configuration>
<property>
<name>fs.default.name</name>
<value>hdfs://master:9000</value>
</property>
<property>
<name>hadoop.tmp.dir</name>
<value>/mnt/raid/tmp/hadoop-ec2-user</value>
</property>
</configuration>
```

The terminal window shows the command 'cat core-site.xml' being run, followed by the XML content of the file. The XML defines two properties: 'fs.default.name' set to 'hdfs://master:9000' and 'hadoop.tmp.dir' set to '/mnt/raid/tmp/hadoop-ec2-user'. The window title is 'ec2-user@slave1:~/hadoop/etc/hadoop 184x51'.

2.hadoop-env.sh:

```
export JAVA_HOME=/usr/lib/jvm/java-1.7.0-openjdk-1.7.0.95.x86_64/
```

```
ec2-user@slave1:~/hadoop/etc/hadoop
# Licensed to the Apache Software Foundation (ASF) under one
# or more contributor license agreements.  See the NOTICE file
# distributed with this work for additional information regarding
# copyright ownership.  The ASF licenses this file
# to you under the Apache License, Version 2.0 (the
# "License"); you may not use this file except in compliance
# with the License.  You may obtain a copy of the License at
#
#     http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.

# Hadoop-specific environment variables here.

# The only required environment variable is JAVA_HOME. All others are
# optional; when running a distributed configuration it is best to
# set JAVA_HOME in this file, so that it is correctly defined on
# remote nodes.

# The Java implementation to use.
export JAVA_HOME=/usr/lib/jvm/java-1.7.0-openjdk-1.7.0.95.x86_64

# The Jsvc implementation to use. Jsvc is required to run secure datanodes
# that bind to privileged ports to provide authentication of data transfer
# protocol. Jsvc is not required if SASL is configured for authentication of
# data transfer protocol using non-privileged ports.
export JSVC_HOME=$JAVA_HOME

export HADOOP_CONF_DIR=$(HADOOP_CONF_DIR:"/etc/hadoop")

# Extra Java CLASSPATH elements. Automatically insert capacity-scheduler.
for f in $HADOOP_HOME/contrib/capacity-scheduler/*.jar; do
  if [ "$HADOOP_CLASSPATH" ]; then
    export HADOOP_CLASSPATH=$HADOOP_CLASSPATH:$f
  else
    export HADOOP_CLASSPATH=$f
  fi
done

# The maximum amount of heap to use, in MB. Default is 1000.
#export HADOOP_HEAPSIZE=
#export HADOOP_NAMENODE_INIT_HEAPSIZE="

# Extra Java runtime options. Empty by default.
export HADOOP_OPTS="$HADOOP_OPTS -Djava.net.preferIPv4Stack=true"

".hadoop-env.sh" 98L, 4260C
37,1      Top
```

3.hdfs-site.xml:

```
ec2-user@slave1:~/hadoop/etc/hadoop
<?xml version="1.0" encoding="UTF-8"?>
<!--
  Licensed under the Apache License, Version 2.0 (the "License");
  you may not use this file except in compliance with the License.
  You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License. See accompanying LICENSE file.
-->
<configuration>
  <!-- Put site-specific property overrides in this file. -->
  <!-->
  <property>
    <name>dfs.replication</name>
    <value>1</value>
  </property>
  <property>
    <name>dfs.permission</name>
    <value>false</value>
  </property>
  <property>
    <name>dfs.namenode.name.dir</name>
    <value>/mnt/raid/hdfs/namenode</value>
  </property>
  <property>
    <name>dfs.datanode.data.dir</name>
    <value>/mnt/raid/hdfs/datanode</value>
  </property>
</configuration>
"hdffs-site.xml" 40L, 1116C
32,1      All
```

```
<configuration>
<property>
<name>dfs.replication</name>
<value>1</value>
</property>
```

```

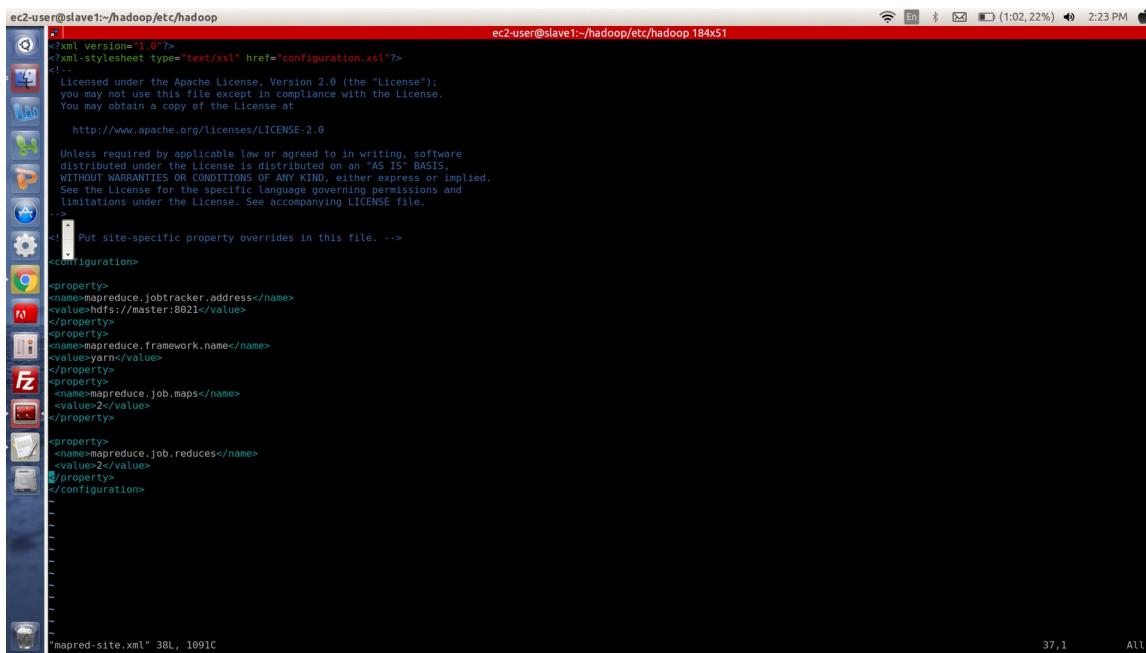
<property>
<name>dfs.permission</name>
<value>false</value>
</property>

<property>
<name>dfs.namenode.name.dir</name>
<value>/mnt/raid/hdfs/namenode</value>
</property>

<property>
<name>dfs.datanode.data.dir</name>
<value>/mnt/raid/hdfs/datanode</value>
</property>
</configuration>

```

4. mapred-site.xml:



```

ec2-user@slave1:~/hadoop/etc/hadoop
[ec2-user@slave1:~/hadoop/etc/hadoop]$ cat mapred-site.xml
xml version="1.0"?
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!-- Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at
http://www.apache.org/licenses/LICENSE-2.0
Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License. See accompanying LICENSE file. -->
<!-- Put site-specific property overrides in this file. -->
<configuration>
<property>
<name>mapreduce.jobtracker.address</name>
<value>hdfs://master:8021</value>
</property>
<property>
<name>mapreduce.framework.name</name>
<value>yarn</value>
</property>
<property>
<name>mapreduce.job.maps</name>
<value>2</value>
</property>
<property>
<name>mapreduce.job.reduces</name>
<value>2</value>
</property>
</configuration>

```

"mapred-site.xml" 38L, 1091C

```

<configuration>
<property>
<name>mapreduce.jobtracker.address</name>
<value>hdfs://master:8021</value>
</property>
<property>
<name>mapreduce.framework.name</name>
<value>yarn</value>
</property>

```

note: following properties are only application to 100 GB

experiments. Value is decided by number of threads available in instance to determine the optimum performance.

```
<property>
    <name>mapreduce.job.maps</name>
    <value>2</value>
</property>
<property>
    <name>mapreduce.job.reduces</name>
    <value>2</value>
</property>

</configuration>
```

5.yarn-site.xml

```
ec2-user@slave1:~/hadoop/etc/hadoop
ec2-user@slave1:~/hadoop/etc/hadoop 18x51
<!-- Site specific YARN configuration properties -->
<configuration>
    <!-- Site specific YARN configuration properties -->
    <property>
        <name>yarn.nodemanager.aux-services</name>
        <value>mapreduce_shuffle</value>
    </property>
    <property>
        <name>yarn.resourcemanager.resource-tracker.address</name>
        <value>master:8031</value>
    </property>
    <property>
        <name>yarn.resourcemanager.address</name>
        <value>master:8032</value>
    </property>
    <property>
        <name>yarn.resourcemanager.scheduler.address</name>
        <value>master:8030</value>
    </property>
    <property>
        <name>yarn.resourcemanager.admin.address</name>
        <value>master:8133</value>
    </property>
    <property>
        <name>yarn.resourcemanager.webapp.address</name>
        <value>master:8088</value>
    </property>
</configuration>
```

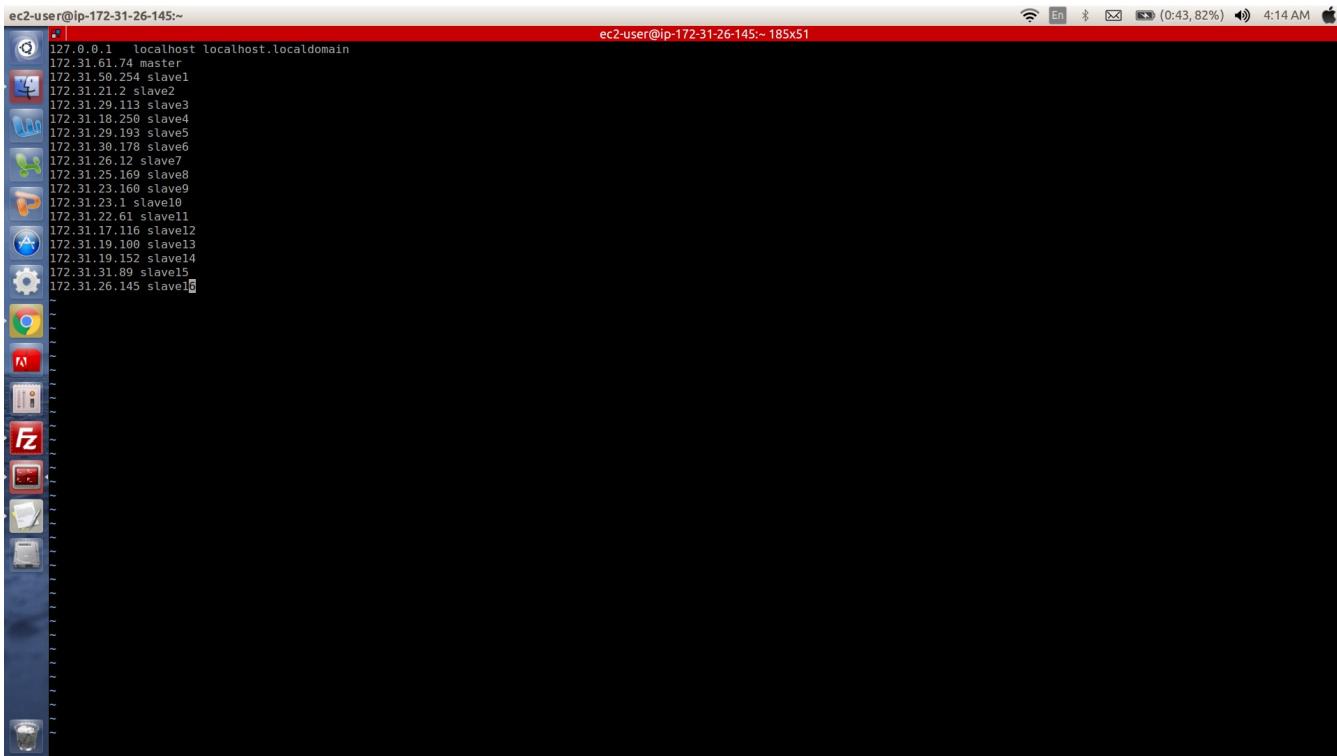
```
<configuration>
<!-- Site specific YARN configuration properties -->
<property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
</property>
<property>
    <name>yarn.resourcemanager.resource-
tracker.address</name>
    <value>master:8031</value>
</property>
<property>
    <name>yarn.resourcemanager.address</name>
    <value>master:8032</value>
```

```
</property>

<property>
    <name>yarn.resourcemanager.scheduler.address</name>
    <value>master:8030</value>
</property>
<property>
    <name>yarn.resourcemanager.admin.address</name>
    <value>master:8133</value>
</property>
<property>
    <name>yarn.resourcemanager.webapp.address</name>
    <value>master:8088</value>
</property>
</configuration>
```

Changes in system files are as follow:

6. etc/hosts:



The screenshot shows a terminal window on a Mac OS X desktop. The title bar says "ec2-user@ip-172-31-26-145:~". The window contains the following text:

```
127.0.0.1 localhost localhost.localdomain
172.31.61.74 master
172.31.50.254 slave1
172.31.21.2 slave2
172.31.29.113 slave3
172.31.18.250 slave4
172.31.29.193 slave5
172.31.30.178 slave6
172.31.26.12 slave7
172.31.25.160 slave8
172.31.23.160 slave9
172.31.23.1 slave10
172.31.22.61 slave11
172.31.17.116 slave12
172.31.19.100 slave13
172.31.19.152 slave14
172.31.31.89 slave15
172.31.26.145 slave16
```

7. `~/.ssh/config`

```

ec2-user@ip-172-31-26-145:~ ec2-user@ip-172-31-26-145:~ 185x51
[...]
Host master
HostName 52-207-246-88.compute-1.amazonaws.com
User ec2-user
IdentityFile ~/.ssh/Hadoop.pem

Host slave1
HostName ec2-54-88-86-1.compute-1.amazonaws.com
User ec2-user
IdentityFile ~/.ssh/Hadoop.pem

Host slave2
HostName ec2-54-164-109-21.compute-1.amazonaws.com
User ec2-user
IdentityFile ~/.ssh/Hadoop.pem

Host slave3
HostName ec2-54-174-6-23.compute-1.amazonaws.com
User ec2-user
IdentityFile ~/.ssh/Hadoop.pem

Host slave4
HostName ec2-54-174-13-249.compute-1.amazonaws.com
User ec2-user
IdentityFile ~/.ssh/Hadoop.pem

Host slave5
HostName ec2-54-174-15-49.compute-1.amazonaws.com
User ec2-user
IdentityFile ~/.ssh/Hadoop.pem

Host slave6
HostName ec2-52-99-82-88.compute-1.amazonaws.com
User ec2-user
IdentityFile ~/.ssh/Hadoop.pem

Host slave7
HostName ec2-52-99-123-165.compute-1.amazonaws.com
User ec2-user
IdentityFile ~/.ssh/Hadoop.pem

Host slave8
HostName ec2-54-174-12-210.compute-1.amazonaws.com
User ec2-user
IdentityFile ~/.ssh/Hadoop.pem

Host slave9
HostName ec2-54-174-13-65.compute-1.amazonaws.com
User ec2-user
IdentityFile ~/.ssh/Hadoop.pem

```

8. `~/.bashrc:`

```

ec2-user@master:~ ec2-user@master:~ 185x51
[...]
# Source global definitions
if [ -f /etc/bashrc ]; then
    . /etc/bashrc
fi

# User specific aliases and functions
# Set Hadoop-related environment variables
export HADOOP_HOME=$HOME/hadoop
export HADOOP_CONF_DIR=$HOME/hadoop/etc/hadoop
export HADOOP_NAMENODE=$HOME/hadoop
export HADOOP_COMMON_HOME=$HOME/hadoop
export HADOOP_HDFS_HOME=$HOME/hadoop
export YARN_HOME=$HOME/hadoop
# Set JAVA_HOME
export JAVA_HOME=/usr/lib/jvm/java-1.7.0-openjdk-1.7.0.95.x86_64/
export PATH=$PATH:$JAVA_HOME/bin
# Add Hadoop bin/ directory to PATH
export PATH=$PATH:$HOME/hadoop/bin
#export HADOOP_CLASSPATH=/usr/lib/jvm/java-1.7.0-openjdk-1.7.0.95.x86_64/lib/tools.jar

```

Installation Procedure:

1. Choose Amazon Machine Image from the Given list, preferably Amazon Linux AMI.
2. Select Instance Type, In our case choose c3.large
3. Select Number of instances and do "spot Request" if you are launching 16 instances. Bid proper price for instance and select the subnet that has the least price mentioned.
4. Add Storage as per requirement. For 10 GB experiment - 30 GB EBS volume is enough while for 100 GB experiment take at least 300 GB worth of space and on slave keep 60 GB EBS.

5. Configure Security Group, Select **All Traffic** to ensure successful connection between server and client.

6. Once Instance is launched follow the procedure:

7. Download java with following link:

```
sudo apt-get install java-1.7.0-openjdk  
sudo apt-get install java-devel
```

8. Download Hadoop from the following link:

```
 wget https://archive.apache.org/dist/hadoop/core/hadoop-2.7.2/hadoop-2.7.2.tar.gz
```

9. Unzip Hadoop with:

```
tar -xvf hadoop-2.7.2.tar.gz  
    rename it:  
mv hadoop-2.7.2 hadoop
```

10. Download gensor from following link:

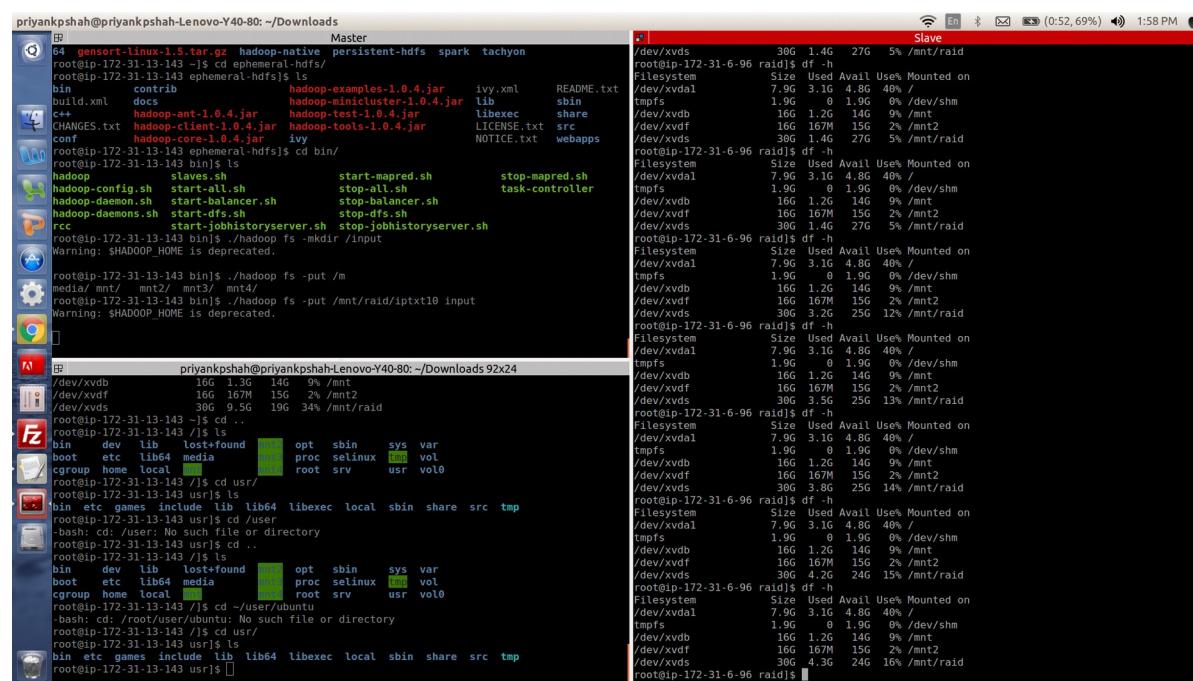
```
wget http://www.ordinal.com/try.cgi/gensort-linux-1.5.tar.gz
```

11. Follows the Configuration procedure from report file.

ScreenShots :

10 GB Experiment:

After Configuration done and data generated, Put data on hadoop file system.



Once data is on hadoop file system, open your browser and type, master_public_dns:50070. You can see that data is on the datanode(slave) and it is running.

Summary

Security is off.
Safemode is off.
23 files and directories, 81 blocks = 104 total filesystem object(s).
Heap Memory used 69.37 MB of 160.5 MB Heap Memory. Max Heap Memory is 889 MB.
Non Heap Memory used 30.83 MB of 31.94 MB Committed Non Heap Memory. Max Non Heap Memory is 214 MB.

Configured Capacity:	29.4 GB
DFS Used:	9.39 GB (31.92%)
Non DFS Used:	3.2 GB
DFS Remaining:	16.82 GB (57.2%)
Block Pool Used:	9.39 GB (31.92%)
DataNodes usages% (Min/Median/Max/stdDev):	31.92% / 31.92% / 31.92% / 0.00%
Live Nodes	1 (Decommissioned: 0)
Dead Nodes	0 (Decommissioned: 0)
Decommissioning Nodes	0
Total Datanode Volume Failures	0 (0 B)
Number of Under-Replicated Blocks	2
Number of Blocks Pending Deletion	0
Block Deletion Start Time	2012-03-27 17:18:17 UTC

Hadoop-Final gensor-t...tar.gz Show all downloads...

Hadoop Map&Reduce Task Running:

```
[ec2-user@ip-172-31-3-46:~] [ec2-user@master:~] [ec2-user@slave1:~]
```

Master

```
File: nohup.out
```

```
16/03/27 23:01:52 INFO mapreduce.Job: map 29% reduce 7%
16/03/27 23:01:53 INFO mapreduce.Job: map 30% reduce 7%
16/03/27 23:01:58 INFO mapreduce.Job: map 31% reduce 8%
16/03/27 23:02:08 INFO mapreduce.Job: map 31% reduce 9%
16/03/27 23:02:12 INFO mapreduce.Job: map 32% reduce 9%
16/03/27 23:02:16 INFO mapreduce.Job: map 33% reduce 9%
16/03/27 23:02:18 INFO mapreduce.Job: map 34% reduce 9%
16/03/27 23:02:20 INFO mapreduce.Job: map 35% reduce 9%
16/03/27 23:02:22 INFO mapreduce.Job: map 36% reduce 9%
16/03/27 23:02:23 INFO mapreduce.Job: map 37% reduce 9%
16/03/27 23:02:37 INFO mapreduce.Job: map 39% reduce 9%
16/03/27 23:02:41 INFO mapreduce.Job: map 40% reduce 9%
16/03/27 23:02:43 INFO mapreduce.Job: map 41% reduce 9%
16/03/27 23:02:43 INFO mapreduce.Job: map 42% reduce 9%
16/03/27 23:02:50 INFO mapreduce.Job: map 42% reduce 10%
16/03/27 23:02:53 INFO mapreduce.Job: map 42% reduce 11%
16/03/27 23:02:55 INFO mapreduce.Job: map 43% reduce 11%
16/03/27 23:02:56 INFO mapreduce.Job: map 43% reduce 12%
16/03/27 23:02:58 INFO mapreduce.Job: map 44% reduce 12%
```

Slave1

```
Filesystem Size Used Avail Use% Mounted on
/dev/xvda1 7.8G 2.5G 5.2G 33% /
devtmpfs 1.9G 64K 1.9G 1% /dev
tmpfs 1.9G 0 1.9G 0% /dev/shm
/dev/xvdb 30G 9.4G 19G 34% /mnt/raid
[ec2-user@slave1 hadoop]$ df -h
Filesystem Size Used Avail Use% Mounted on
/dev/xvda1 7.8G 2.5G 5.2G 33% /
devtmpfs 1.9G 64K 1.9G 1% /dev
tmpfs 1.9G 0 1.9G 0% /dev/shm
/dev/xvdb 30G 9.5G 19G 34% /mnt/raid
[ec2-user@slave1 hadoop]$ df -h
Filesystem Size Used Avail Use% Mounted on
/dev/xvda1 7.8G 2.5G 5.2G 33% /
devtmpfs 1.9G 64K 1.9G 1% /dev
tmpfs 1.9G 0 1.9G 0% /dev/shm
/dev/xvdb 30G 9.5G 19G 34% /mnt/raid
[ec2-user@slave1 hadoop]$ df -h
Filesystem Size Used Avail Use% Mounted on
/dev/xvda1 7.8G 2.5G 5.2G 33% /
devtmpfs 1.9G 64K 1.9G 1% /dev
tmpfs 1.9G 0 1.9G 0% /dev/shm
/dev/xvdb 30G 9.5G 19G 34% /mnt/raid
[ec2-user@slave1 hadoop]$
```

Hadoop Map&Reduce Task done:

```
ec2-user@ip-172-31-3-46:~
```

Master					
GNU nano 2.3.1		File: nohup.out			
16/03/27 23:13:02 INFO mapreduce.Job: map 100% reduce 98%					
16/03/27 23:13:08 INFO mapreduce.Job: map 100% reduce 99%					
16/03/27 23:13:11 INFO mapreduce.Job: map 100% reduce 100%					
16/03/27 23:13:14 INFO mapreduce.Job: job job_1459118954596_0001 completed successfully					
16/03/27 23:13:14 INFO mapreduce.Job: Counters: 51					
File System Counters					
FILE: Number of bytes read=29960314449					
FILE: Number of bytes written=40069222029					
FILE: Number of read operations=0					
FILE: Number of large read operations=0					
FILE: Number of write operations=0					
HDFS: Number of bytes read=10000311429					
HDFS: Number of bytes written=99000000000					
HDFS: Number of read operations=228					
HDFS: Number of large read operations=0					
HDFS: Number of write operations=2					
Job Counters					
Killed map tasks=4					
Launched map tasks=79					
Launched reduce tasks=1					
Get Help	Write Out	Read File	Prev Page	Cut Text	Cur Pos
Exit	Justify	Where Is	Next Page	Uncut Text	To Spell
Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/xvda1	7.8G	2.5G	5.2G	33%	/
devtmpfs	1.9G	64K	1.9G	1%	/dev
tmpfs	1.9G	0	1.9G	0%	/dev/shm
/dev/xvdb	30G	9.4G	19G	34%	/mnt/raid
[ec2-user@slave1 hadoop]\$ df -h					
Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/xvda1	7.8G	2.5G	5.2G	33%	/
devtmpfs	1.9G	64K	1.9G	1%	/dev
tmpfs	1.9G	0	1.9G	0%	/dev/shm
/dev/xvdb	30G	9.5G	19G	34%	/mnt/raid
[ec2-user@slave1 hadoop]\$ df -h					
Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/xvda1	7.8G	2.5G	5.2G	33%	/
devtmpfs	1.9G	64K	1.9G	1%	/dev
tmpfs	1.9G	0	1.9G	0%	/dev/shm
/dev/xvdb	30G	9.5G	19G	34%	/mnt/raid
[ec2-user@slave1 hadoop]\$ df -h					
Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/xvda1	7.8G	2.5G	5.2G	33%	/
devtmpfs	1.9G	64K	1.9G	1%	/dev
tmpfs	1.9G	0	1.9G	0%	/dev/shm
/dev/xvdb	30G	9.5G	19G	34%	/mnt/raid
[ec2-user@slave1 hadoop]\$					

Hadoop Valsort:

ec2-user@ip-172-31-3-46:~

```
-bash: ./hadoop: Is a directory
[ec2-user@ip-172-31-3-46 ~]$ hadoop fs -get /user/ec2-user/output_10gb/* /mnt/raid/
[ec2-user@ip-172-31-3-46 ~]$ cd /mnt/raid/
[ec2-user@ip-172-31-3-46 raid]$ ls
drwxrwxrwx 10 root root 0 Jan 18 11:40 part-r-000000 _SUCCESS tmp
[ec2-user@ip-172-31-3-46 raid]$ ls
drwxrwxrwx 10 root root 0 Jan 18 11:40 part-r-000000 _SUCCESS tmp
[ec2-user@ip-172-31-3-46 raid]$ mv part-r-000000 op.txt
[ec2-user@ip-172-31-3-46 raid]$ unix2dos op.txt
unix2dos: converting file op.txt to DOS format ...
unix2dos: can not write to output file
unix2dos: problem converting file op.txt
[ec2-user@ip-172-31-3-46 raid]$ unix2dos op.txt
unix2dos: converting file op.txt to DOS format ...
^[[A[ec2-user@ip-172-31-3-46 raid]$ cd ~
[ec2-user@ip-172-31-3-46 ~]$ ls
gensor* hadoop nohup.out Terasort.class Terasort.java Terasort$TSMapper.class Terasort$TSReducer.class ts.jar valsort
[bash: cd: gensor: Not a directory
[ec2-user@ip-172-31-3-46 ~]$ ./valsort /mnt/raid/op.txt
Records: 100000000
Checksum: 2faef2a1dfa8909
Duplicate keys: 0
SUCCESS - all records are in order
[ec2-user@ip-172-31-3-46 ~]$ 
```

ec2-user@master:~/ssh 185x24

```
/dev/xvda1 7.86 1.66 6.16 21% /
/dev/mpt5 1.96 64K 1.96 1% /dev
tmpfs 1.96 0 1.96 0% /dev/shm
/dev/xvdb 30G 9.4G 19G 34% /mnt/raid
[ec2-user@master raid]$ df -h
Filesystem Size Used Avail Use% Mounted on
/dev/xvda1 7.86 1.66 6.16 21% /
/dev/xvda1 7.86 1.66 6.16 21% /
/dev/mpt5 1.96 64K 1.96 1% /dev
tmpfs 1.96 0 1.96 0% /dev/shm
/dev/xvdb 30G 9.4G 19G 34% /mnt/raid
[ec2-user@master raid]$ df -h
Filesystem Size Used Avail Use% Mounted on
/dev/xvda1 7.86 1.66 6.16 21% /
/dev/mpt5 1.96 64K 1.96 1% /dev
tmpfs 1.96 0 1.96 0% /dev/shm
/dev/xvdb 30G 9.4G 19G 34% /mnt/raid
[ec2-user@master raid]$ cd ~/.ssh/
[ec2-user@master .ssh]$ ls
authorized_keys config Hadoop.pem id_rsa id_rsa.pub known_hosts
[ec2-user@master .ssh]$ vi config
[ec2-user@master .ssh]$ vi config
[ec2-user@master .ssh]$ ls
authorized_keys config Hadoop.pem id_rsa id_rsa.pub known_hosts
[ec2-user@master .ssh]$ 
```

100 GB Experiment:

16 nodes Running:

The screenshot shows the AWS EC2 Management Console interface. On the left, there's a sidebar with icons for EC2 Dashboard, Events, Tags, Reports, Limits, Instances, AMIs, Bundle Tasks, Elastic Block Store, Volumes, Snapshots, Network & Security, Security Groups, Elastic IPs, Placement Groups, Key Pairs, and Network Interfaces. The main area is titled 'Launch Instance' and shows a table of 17 instances. The table includes columns for Name, Instance ID, Instance Type, Availability Zone, Instance State, Status Checks, Alarm Status, and Public DNS. Most instances are labeled 'Slave' followed by a number, while one is labeled 'Master'. All instances are in the 'running' state with 2/2 checks passed. The Public DNS column lists various IP addresses such as ec2-54-165-26-20.cc, ec2-52-207-226-187, ec2-54-174-14-163.cc, etc.

Datanode Started:

```
ec2-user@master:~/hadoop/bin
Starting namenodes on [master]
master: starting namenode, logging to /home/ec2-user/hadoop/logs/hadoop-ec2-user-namenode-master.out
slave1: starting datanode, logging to /home/ec2-user/hadoop/logs/hadoop-ec2-user-datanode-slave1.out
slave2: starting datanode, logging to /home/ec2-user/hadoop/logs/hadoop-ec2-user-datanode-slave2.out
slave3: starting datanode, logging to /home/ec2-user/hadoop/logs/hadoop-ec2-user-datanode-slave3.out
slave4: starting datanode, logging to /home/ec2-user/hadoop/logs/hadoop-ec2-user-datanode-slave4.out
slave5: starting datanode, logging to /home/ec2-user/hadoop/logs/hadoop-ec2-user-datanode-slave5.out
slave6: starting datanode, logging to /home/ec2-user/hadoop/logs/hadoop-ec2-user-datanode-slave6.out
slave7: starting datanode, logging to /home/ec2-user/hadoop/logs/hadoop-ec2-user-datanode-slave7.out
slave8: starting datanode, logging to /home/ec2-user/hadoop/logs/hadoop-ec2-user-datanode-slave8.out
slave9: starting datanode, logging to /home/ec2-user/hadoop/logs/hadoop-ec2-user-datanode-slave9.out
slave10: starting datanode, logging to /home/ec2-user/hadoop/logs/hadoop-ec2-user-datanode-slave10.out
slave11: starting datanode, logging to /home/ec2-user/hadoop/logs/hadoop-ec2-user-datanode-slave11.out
slave12: starting datanode, logging to /home/ec2-user/hadoop/logs/hadoop-ec2-user-datanode-slave12.out
slave13: starting datanode, logging to /home/ec2-user/hadoop/logs/hadoop-ec2-user-datanode-slave13.out
slave14: starting datanode, logging to /home/ec2-user/hadoop/logs/hadoop-ec2-user-datanode-slave14.out
slave15: starting datanode, logging to /home/ec2-user/hadoop/logs/hadoop-ec2-user-datanode-slave15.out
slave16: starting datanode, logging to /home/ec2-user/hadoop/logs/hadoop-ec2-user-datanode-slave16.out
slave17: starting datanode, logging to /home/ec2-user/hadoop/logs/hadoop-ec2-user-datanode-slave17.out
slave18: starting datanode, logging to /home/ec2-user/hadoop/logs/hadoop-ec2-user-datanode-slave18.out
slave19: starting datanode, logging to /home/ec2-user/hadoop/logs/hadoop-ec2-user-datanode-slave19.out
Starting secondary namenodes [0.0.0.0]
0.0.0.0: 0.0.0.0@ec2-54-165-26-20.cc:50010
0.0.0.0: @ WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!
0.0.0.0: @ (DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=ec2-54-165-26-20.cc)(PORT=50010))
0.0.0.0: IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
0.0.0.0: Someone could be eavesdropping on you right now (man-in-the-middle attack)!
0.0.0.0: It is also possible that a host key has just been changed.
0.0.0.0: The fingerprint for the ECDSA key sent by the remote host is
0.0.0.0: 46:eb:7d:b7:66:11:40:f0:a1:0b:56:43:bd:c4:74:e1.
0.0.0.0: Please contact your system administrator.
0.0.0.0: Add correct host key in /home/ec2-user/.ssh/known_hosts to get rid of this message.
0.0.0.0: Offending ECDSA key in /home/ec2-user/.ssh/known_hosts:3
0.0.0.0: ECDSA host key for 0.0.0.0 has changed and you have requested strict checking.
0.0.0.0: Host key verification failed.
[ec2-user@master ~]$ ./start-yarn.sh
starting yarn daemons
starting resourcemanager, logging to /home/ec2-user/hadoop/logs/yarn-ec2-user-resourcemanager-master.out
slave1: starting nodemanager, logging to /home/ec2-user/hadoop/logs/yarn-ec2-user-nodemanager-slave1.out
slave2: starting nodemanager, logging to /home/ec2-user/hadoop/logs/yarn-ec2-user-nodemanager-slave2.out
slave3: starting nodemanager, logging to /home/ec2-user/hadoop/logs/yarn-ec2-user-nodemanager-slave3.out
slave4: starting nodemanager, logging to /home/ec2-user/hadoop/logs/yarn-ec2-user-nodemanager-slave4.out
slave5: starting nodemanager, logging to /home/ec2-user/hadoop/logs/yarn-ec2-user-nodemanager-slave5.out
slave6: starting nodemanager, logging to /home/ec2-user/hadoop/logs/yarn-ec2-user-nodemanager-slave6.out
slave7: starting nodemanager, logging to /home/ec2-user/hadoop/logs/yarn-ec2-user-nodemanager-slave7.out
slave8: starting nodemanager, logging to /home/ec2-user/hadoop/logs/yarn-ec2-user-nodemanager-slave8.out
slave9: starting nodemanager, logging to /home/ec2-user/hadoop/logs/yarn-ec2-user-nodemanager-slave9.out
slave10: starting nodemanager, logging to /home/ec2-user/hadoop/logs/yarn-ec2-user-nodemanager-slave10.out
slave11: starting nodemanager, logging to /home/ec2-user/hadoop/logs/yarn-ec2-user-nodemanager-slave11.out
slave12: starting nodemanager, logging to /home/ec2-user/hadoop/logs/yarn-ec2-user-nodemanager-slave12.out
slave13: starting nodemanager, logging to /home/ec2-user/hadoop/logs/yarn-ec2-user-nodemanager-slave13.out
slave14: starting nodemanager, logging to /home/ec2-user/hadoop/logs/yarn-ec2-user-nodemanager-slave14.out
slave15: starting nodemanager, logging to /home/ec2-user/hadoop/logs/yarn-ec2-user-nodemanager-slave15.out
```

Put Data on Hadoop filesystem:

```

ec2-user@master:~/hadoop/bin 92x51
[slave10: starting nodemanager, logging to /home/ec2-user/hadoop/logs/yarn-ec2-user-nodemanager]
[slave6: starting nodemanager, logging to /home/ec2-user/hadoop/logs/yarn-ec2-user-nodemanager]
[slave9: starting nodemanager, logging to /home/ec2-user/hadoop/logs/yarn-ec2-user-nodemanager]
[slave11: starting nodemanager, logging to /home/ec2-user/hadoop/logs/yarn-ec2-user-nodemanager]
[slave7: starting nodemanager, logging to /home/ec2-user/hadoop/logs/yarn-ec2-user-nodemanager]
[slave16: starting nodemanager, logging to /home/ec2-user/hadoop/logs/yarn-ec2-user-nodemanager]
[slave14: starting nodemanager, logging to /home/ec2-user/hadoop/logs/yarn-ec2-user-nodemanager]
[slave13: starting nodemanager, logging to /home/ec2-user/hadoop/logs/yarn-ec2-user-nodemanager]
[slave15: starting nodemanager, logging to /home/ec2-user/hadoop/logs/yarn-ec2-user-nodemanager]
[ec2-user@master sbin]$ jps
10267 NameNode
10916 Jps
10618 ResourceManager
[ec2-user@master sbin]$ ./yarn-daemon.sh start nodemanager
starting nodemanager, logging to /home/ec2-user/hadoop/logs/yarn-ec2-user-nodemanager-master
[ec2-user@master sbin]$ cd ..
[ec2-user@master hadoop]$ ls
bin etc include lib libexec LICENSE.txt logs NOTICE.txt README.txt sbin share
[ec2-user@master hadoop]$ cd bin/
[ec2-user@master bin]$ hadoop fs -ls /user/ec2-user/input/
ls: '/user/ec2-user/input/': No such file or directory
[ec2-user@master bin]$ hadoop fs -ls /user/ec2-user/
ls: '/user/ec2-user/': No such file or directory
[ec2-user@master bin]$ hadoop fs -ls /
[ec2-user@master bin]$ hdfs fs -ls /
[ec2-user@master bin]$ df -h
Error: Could not find or load main class fs
[ec2-user@master bin]$ df -h
Filesystem      Size   Used  Avail Use% Mounted on
/dev/xvda1     7.8G  1.6G  6.1G  21% /
devtmpfs        1.9G  64K  1.9G  1% /dev
tmpfs          1.9G  0    1.9G  0% /dev/shm
/dev/xvdb     596M  5.3G  51G  10% /mnt/raid
[ec2-user@master sbin]$ ls
[ec2-user@master raid]$ ls
[ec2-user@master raid]$ iptxt100 lost+found tmp
[ec2-user@master raid]$ ls -lth
total 94G
drwx----- 2 root   root  16K Mar 28 16:20 lost+found
drwxrwxr-x 3 ec2-user ec2-user 4.0K Mar 28 16:21 tmp
drwxrwxrwx 4 ec2-user ec2-user 4.0K Mar 28 16:21 . .
-rwxrwxr-x 1 ec2-user ec2-user 94G Mar 28 16:43 iptxt100
[ec2-user@master raid]$ cd
[ec2-user@master ~]$ cd hadoop/bin/
[ec2-user@master bin]$ hadoop fs -put /mnt/raid/iptxt100 input
put: input: No such file or directory
[ec2-user@master bin]$ hadoop fs -mkdir /user
[ec2-user@master bin]$ hadoop fs -mkdir /user/ec2-user
[ec2-user@master bin]$ hadoop fs -mkdir /user/ec2-user/input
[ec2-user@master bin]$ hadoop fs -put /mnt/raid/iptxt100 input

ec2-user@slave1:~$ df -h
Filesystem      Size   Used  Avail Use% Mounted on
/dev/xvda1     7.8G  1.6G  6.1G  21% /
devtmpfs        1.9G  64K  1.9G  1% /dev
tmpfs          1.9G  0    1.9G  0% /dev/shm
/dev/xvdb     596M  5.3G  51G  10% /mnt/raid
[ec2-user@slave1 ~]$ df -h
Filesystem      Size   Used  Avail Use% Mounted on
/dev/xvda1     7.8G  1.6G  6.1G  21% /
devtmpfs        1.9G  64K  1.9G  1% /dev
tmpfs          1.9G  0    1.9G  0% /dev/shm
/dev/xvdb     596M  5.3G  51G  10% /mnt/raid
[ec2-user@slave1 ~]$ 

ec2-user@slave5:~$ df -h
Filesystem      Size   Used  Avail Use% Mounted on
/dev/xvda1     7.8G  1.6G  6.1G  21% /
devtmpfs        1.9G  64K  1.9G  1% /dev
tmpfs          1.9G  0    1.9G  0% /dev/shm
/dev/xvdb     596M  5.3G  51G  10% /mnt/raid
[ec2-user@slave5 ~]$ df -h
Filesystem      Size   Used  Avail Use% Mounted on
/dev/xvda1     7.8G  1.6G  6.1G  21% /
devtmpfs        1.9G  64K  1.9G  1% /dev
tmpfs          1.9G  0    1.9G  0% /dev/shm
/dev/xvdb     596M  5.3G  51G  10% /mnt/raid
[ec2-user@slave5 ~]$ 

ec2-user@slave9:~$ df -h
Filesystem      Size   Used  Avail Use% Mounted on
/dev/xvda1     7.8G  1.6G  6.1G  21% /
devtmpfs        1.9G  64K  1.9G  1% /dev
tmpfs          1.9G  0    1.9G  0% /dev/shm
/dev/xvdb     596M  5.3G  51G  10% /mnt/raid
[ec2-user@slave9 ~]$ df -h
Filesystem      Size   Used  Avail Use% Mounted on
/dev/xvda1     7.8G  1.6G  6.1G  21% /
devtmpfs        1.9G  64K  1.9G  1% /dev
tmpfs          1.9G  0    1.9G  0% /dev/shm
/dev/xvdb     596M  5.3G  51G  10% /mnt/raid
[ec2-user@slave9 ~]$ 

ec2-user@slave13:~$ df -h
Filesystem      Size   Used  Avail Use% Mounted on
/dev/xvda1     7.8G  1.6G  6.1G  21% /
devtmpfs        1.9G  64K  1.9G  1% /dev
tmpfs          1.9G  0    1.9G  0% /dev/shm
/dev/xvdb     596M  5.3G  51G  10% /mnt/raid
[ec2-user@slave13 ~]$ 

```

Data Set on Datanode:

Overview 'master:9000' (active)

Started:	Mon Mar 28 20:56:15 UTC 2016
Version:	2.7.2, rb165c4f68a74250c792e23546c64604acf0e41
Compiled:	2016-01-26T00:08Z by jenkins from (detached from b165c4f)
Cluster ID:	CID-c0d01f1b-7233-4bc1-a3d9-2834740908b31
Block Pool ID:	BP-106206480-172.31.47.62-1459194526687

Summary

Security is off.
Safemode is off.
5 files and directories, 746 blocks = 751 total filesystem object(s).
Heap Memory used 60.27 MB of 451.5 MB Heap Memory. Max Heap Memory is 889 MB.
Non Heap Memory used 34.33 MB of 34.94 MB Committed Non Heap Memory. Max Non Heap Memory is 214 MB.

Configured Capacity:	942.94 GB
DFS Used:	93.86 GB (9.95%)
Non DFS Used:	49.06 GB
DFS Remaining:	800.01 GB (84.64%)
Block Pool Used:	93.86 GB (9.95%)
DataNodes usages% (Min/Median/Max/stdDev):	7.05% / 9.62% / 12.61% / 1.36%
Live Nodes	16 (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
Block Deletion Start Time	3/28/2016, 3:56:15 PM

DataNodes :

Google Chrome

Namenode information - Google Chrome

EC2 Management CS 553 (38 unres.) CS-553-Cloud-Co Namenode info MapReduce Job

ec2-54-175-146-89.compute-1.amazonaws.com:50070/dfshealth.html#tab-datanode

Datanode Information

In operation

Node	Last contact	Admin State	Capacity	Used	Non DFS Used	Remaining	Blocks	Block pool used	Failed Volumes	Version
slave8:50010 (172.31.50.113:50010)	2	In Service	58.93 GB	6.55 GB	9.88 GB	42.5 GB	54	6.55 GB (11.12%)	0	2.7.2
slave7:50010 (172.31.54.42:50010)	2	In Service	58.93 GB	5.67 GB	9.51 GB	43.76 GB	47	5.67 GB (9.62%)	0	2.7.2
slave1:50010 (172.31.50.216:50010)	2	In Service	58.93 GB	6.8 GB	10.39 GB	41.74 GB	56	6.8 GB (11.54%)	0	2.7.2
slave12:50010 (172.31.55.202:50010)	2	In Service	58.93 GB	5.42 GB	49.3 GB	4.21 GB	43	5.42 GB (9.19%)	0	2.7.2
slave16:50010 (172.31.53.247:50010)	2	In Service	58.93 GB	5.67 GB	10.21 GB	43.06 GB	47	5.67 GB (9.62%)	0	2.7.2
slave14:50010 (172.31.56.22:50010)	2	In Service	58.93 GB	7.43 GB	10.26 GB	41.24 GB	60	7.43 GB (12.81%)	0	2.7.2
slave3:50010 (172.31.51.34:50010)	2	In Service	58.93 GB	6.43 GB	34.44 GB	18.07 GB	53	6.43 GB (10.91%)	0	2.7.2
slave4:50010 (172.31.57.236:50010)	2	In Service	58.93 GB	5.92 GB	9.76 GB	43.25 GB	49	5.92 GB (10.05%)	0	2.7.2
slave15:50010 (172.31.57.77:50010)	2	In Service	58.93 GB	5.04 GB	9.25 GB	44.64 GB	40	5.04 GB (8.55%)	0	2.7.2
slave9:50010 (172.31.51.235:50010)	2	In Service	58.93 GB	5.54 GB	10.56 GB	42.81 GB	46	5.54 GB (9.41%)	0	2.7.2
slave5:50010 (172.31.51.177:50010)	2	In Service	58.93 GB	6.3 GB	9.88 GB	42.75 GB	50	6.3 GB (10.89%)	0	2.7.2
slave2:50010 (172.31.48.191:50010)	2	In Service	58.93 GB	4.16 GB	44.92 GB	9.85 GB	36	4.16 GB (7.05%)	0	2.7.2
slave11:50010 (172.31.48.46:50010)	2	In Service	58.93 GB	5.42 GB	9.95 GB	43.56 GB	45	5.42 GB (9.19%)	0	2.7.2
slave19:50010 (172.31.49.42:50010)	2	In Service	58.93 GB	6.93 GB	50.72 GB	1.28 GB	57	6.93 GB (11.76%)	0	2.7.2
slave13:50010 (172.31.57.6:50010)	2	In Service	58.93 GB	5.29 GB	9.58 GB	44.07 GB	42	5.29 GB (8.98%)	0	2.7.2
slave6:50010 (172.31.52.194:50010)	2	In Service	58.93 GB	5.3 GB	9.76 GB	43.88 GB	44	5.3 GB (8.99%)	0	2.7.2

Decommissioning

Node	Last contact	Under replicated blocks	Blocks with no live replicas	Under Replicated Blocks In files under construction
------	--------------	-------------------------	------------------------------	--------------------------------------------------------

Hadoop, 2015.

Hadoop Job Running:

Terminator

MapReduce Job job_1459198586907_0001 - Google Chrome

EC2 Management CS 553 (38 unres.) CS-553-Cloud-Co Namenode info MapReduce Job

ec2-54-175-146-89.compute-1.amazonaws.com:8088/proxy/application_1459198586907_0001/mapreduce/job/job_1459198586907_0001

Logged in as: dr.who

MapReduce Job job_1459198586907_0001

Job Overview

Job Name:	Tera sort	State:	RUNNING
Uberized:	false	Started:	Mon Mar 28 21:29:48 UTC 2016
Elapsed:	9min, 52sec		

ApplicationMaster

Attempt Number	Start Time	Node	Logs
1	Mon Mar 28 21:29:43 UTC 2016	slave6:8042	logs

Task Type

Map	Reduce	Progress	Total	Pending	Running	Complete
map 79%	reduce 16%	745	0	12	733	0
map 79%	reduce 16%	2	0	2	0	0

Attempt Type

Maps	Reduces	New	Running	Failed	Killed	Successful
0	0	12	0	0	1	733

ec2-user@mast:~/hadoop/bin

```
ec2-user@mast:~/hadoop/bin 80x24
16/03/28 21:37:04 INFO mapreduce.Job: map 76% reduce 15%
16/03/28 21:37:12 INFO mapreduce.Job: map 77% reduce 15%
16/03/28 21:37:16 INFO mapreduce.Job: map 77% reduce 16%
16/03/28 21:37:18 INFO mapreduce.Job: map 78% reduce 16%
16/03/28 21:37:21 INFO mapreduce.Job: map 79% reduce 16%
16/03/28 21:37:25 INFO mapreduce.Job: map 80% reduce 16%
16/03/28 21:37:34 INFO mapreduce.Job: map 81% reduce 16%
16/03/28 21:37:37 INFO mapreduce.Job: map 82% reduce 16%
16/03/28 21:37:42 INFO mapreduce.Job: map 83% reduce 17%
16/03/28 21:37:45 INFO mapreduce.Job: map 84% reduce 17%
16/03/28 21:37:51 INFO mapreduce.Job: map 85% reduce 17%
16/03/28 21:37:54 INFO mapreduce.Job: map 85% reduce 18%
16/03/28 21:38:01 INFO mapreduce.Job: map 86% reduce 18%
16/03/28 21:38:05 INFO mapreduce.Job: map 86% reduce 19%
16/03/28 21:38:10 INFO mapreduce.Job: map 87% reduce 19%
16/03/28 21:38:17 INFO mapreduce.Job: map 88% reduce 19%
16/03/28 21:38:19 INFO mapreduce.Job: map 88% reduce 20%
16/03/28 21:38:22 INFO mapreduce.Job: map 89% reduce 20%
16/03/28 21:38:25 INFO mapreduce.Job: map 90% reduce 20%
16/03/28 21:38:30 INFO mapreduce.Job: map 91% reduce 20%
16/03/28 21:38:35 INFO mapreduce.Job: map 92% reduce 20%
16/03/28 21:38:41 INFO mapreduce.Job: map 93% reduce 20%
16/03/28 21:38:47 INFO mapreduce.Job: map 92% reduce 6%
16/03/28 21:38:52 INFO mapreduce.Job: map 93% reduce 6%
```

Problem Faced:

1. Datanode was not generating on slave nodes, For this problem first delete the any current data in datanode. If problem persist, check path in hdfs-site.xml file. Check if sufficient permission is given to datanode directory to write datanode directory.
2. During the early phase of assignment, I didn't created alias for master and slave IP(s). So every time, I have to make changes to the hdfs-site file and yarn-site file for Master public DNS. Then I came to know about the alias in /etc/hosts file and ~/.ssh/config file which replaced to **master** and **slave** with their respective public DNS and private ips, in turns it decrease the amount of time taken to re-configure the master and slave node.
3. While doing 100 GB experiment, I attached 30 GB EBS volume to slaves, soon cluster ran out of space. At that time instead of terminating it and creating and configuring whole cluster, I found that we can attach EBS volume on running instance.
4. It was difficult to set 16 node cluster, especially manually configuring the files on each slave, I made a script which will copy different script and copy it to all the slaves and execute them remotely.

Methodology:

- First of all, Program will take input from the input file and store it as value.
- We need to sort the generated data on the base of key so, we will fetch first 10 char from the string and store it as key in Text data type field and rest in the another field as value.
- Mapper class sort the data which is key in our case before passing it to reducer class, so in second class that is TSDReducer we will fetch the key(0,10 - char) with its respective value(10,end - char) and store it in Text. We will iterate one by one value and store.
- We will write it back as (key , value). That is our output. Here Reducer class will work as combiner too.

Questionnaires:

- **conf/masters** that is resided in hadoop/etc/hadoop, stores the hostname of master node.
- **conf/slaves** that is resided in hadoop/etc/hadoop, stores the hostname of all the slave nodes.
- **conf/core-site.xml** that is resided in hadoop/etc/hadoop, it

overrides the default core parameters. It informs the Hadoop deamon where namenode runs in cluster. Contains common I/O settings related to HDFS and MapReduce.

- **conf/hdfs-site.xml** that is resided in hadoop/etc/hadoop, contains the setting of Namenode , Secondary Namenode and Datanode like number of replication, permission checking on HDFS.
- **conf/mapred-site.xml** that is resided in hadoop/etc/hadoop, contains the configuration settings of MapReduce Daemons, job tracker and task-tracker.
- **What is Master node ? What is Data node?**

Master node hosts the various storage and processing management services, described in list for entire Hadoop cluster, Namenode service are created on the top of Master Node.

Data node is responsible for storing actual data in HDFS and processing them, Datanode service are created on the top of Data node.

- **Why we need to set unique port in shared environment service? Error or side-effect of using same port?**

Each service uses different port to uniquely identified by callee. So it will not throw error , but it will not able to perform either of the service that has the same port numbers.

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

In hadoop/etc/hadoop there is a file called mapred-site.xml . We can add the value as number of mappers and reducers for **mapreduce.job.maps** and **mapreduce.job.reduces**.

- **Changes made while scaling from 1 node to 16 Node:**

Particularly for this experiment we are using 10 GB dataset for 1 slave and 100 GB dataset for 16 slaves. So we need more space on both Master and Slave for 16 node Experiments. I would suggest at least 250 GB of space on Master Node. That is once you put data on Hadoop file system, delete the generated data to let output store.

Otherwise, We need to mention all the 16 nodes private ip in/etc/hosts file and public dns in ~/.ssh/config file.

Add slaves public dns in known_host file resides in same location as config file.

Mention hostname/alias as per /etc/hosts in hadoop/etc/hadoop/slaves file.

Change as mentioned above in configuration file in etc/hadoop/mapred-site.xml.

2. Spark Benchmarking:

Instance Type: c3.large
 RAM: 3.75 GB
 # of Instances: 1(for 10 GB) and 17(1 master and 16 slave)
 Storage: 32 GB
 OS: Amazon Linux AMI
 Java version: 1.7.0
 Spark Version: 1.6.1

Installation Procedure:

1. Choose Amazon Machine Image from the Given list, preferably Amazon Linux AMI.
2. Select Instance Type, In our case choose c3.large
3. Select Number of instances and do "spot Request" if you are launching 16 instances. Bid proper price for instance and select the subnet that has the least price mentioned.
4. Add Storage as per requirement. For 10 GB experiment - 30 GB EBS volume is enough, for 100 GB experiment have at least 250 GB space and on slave take 60 GB EBS.
5. Configure Security Group, Select **All Traffic** to ensure successful connection between server and client.
6. Once Instance is launched follow the procedure to download necessary files:

Download Java:

```

sudo apt-get update
sudo apt-get install openjdk-7-jre-headless
sudo apt-get install openjdk-7-jdk

```

Download Spark and unzip it:

```

wget http://mirrors.ocf.berkeley.edu/apache/spark/spark-1.6.1/spark-1.6.1-bin-hadoop2.6.tgz

```

Download Scala:

```

wget http://www.scala-lang.org/files/archive/scala-2.10.4.tgz

```

7. Create local folder and unzip scala:

```

sudo mkdir /usr/local/src/scala
sudo tar xvf scala-2.10.4.tgz -C /usr/local/src/scala/

```

8. Open **bashrc** and add following parameter:

```

vi .bashrc
export SCALA_HOME=/usr/local/src/scala/scala-2.10.4
export PATH=$SCALA_HOME/bin:$PATH

```

source the bashrc:

```

. ./bashrc

```

Check if Scala is properly configured with
Scala -version

9. Export your Access Key and Access ID:

```
export AWS_ACCESS_KEY_ID=<Access Key ID>
export AWS_SECRET_ACCESS_KEY=<Secret Access Key>
```

These keys can be generated by following steps:

go to your aws account, Click on your name and click on Security Credential.Click on Access Keys, Generate a New one and Store it somewhere safe.

10. Once Spark is Configured as mentioned in report, do the following steps:

for 1 Node Run following command:

```
./spark-1.6.1/ec2/spark-ec2 -k Spark -i Spark.pem --slaves=1
--instance-type=c3.large --ebs-vol-size=30 --spot-price=0.04 launch
spark_10gb
```

for 16 Node Run following command:

```
./spark-1.6.1/ec2/spark-ec2 -k Spark -i Spark.pem --slaves=16
--instance-type=c3.large -m c3.4xlarge --spot-price=0.15 launch
spark_100gb
```

11. Change the configuration files as following in Master node launched from Main node:

1. ephermal/conf/core-site.xml

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

<property>
  <name>fs.default.name</name>
<value>hdfs://MASTER's_PUBLIC_DNS:9000</value>
</property>

<property>
  <name>fs.defaultFS</name>
<value>hdfs://MASTER's_PUBLIC_DNS:9000</value>
</property>
</configuration>
```

2. `ephemeral/conf/hdfs-site.xml`

Changes are only in following properties:

```
<property>
    <name>dfs.replication</name>
    <value>1</value>
</property>

<property>
    <name>dfs.data.dir</name>
    <value>/mnt/raid/</value>
</property>
```

3. `spark/conf/core-site.xml`

Changes are only in following properties:

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

<property>
    <name>fs.default.name</name>
    <value>hdfs://MASTER's PUBLIC_DNS:9000</value>
</property>
```

12. Once Configuration are done, copy it in all the slaves:

```
./copy-dir ~/ephemeral-hdfs/conf/
./copy-dir ~/spark/conf/
```

13. Stop all the services with following command:

```
ephemeral-hdfs/bin/./stop-dfs.sh
spark/sbin/./stop-all.sh
```

14. Format the namenode:

```
ephemeral-hdfs/bin/.hadoop namenode -format
```

15. Start all the Service with following commands:

```
cd spark/sbin/ → ./start-all.sh
cd ephemeral-hdfs/bin/ → ./start-dfs.sh
```

ScreenShots :**10 GB Experiments :****Spark Cluster on AWS:**

The screenshot shows the AWS EC2 Management Console in Google Chrome. The left sidebar navigation bar includes links for EC2 Dashboard, Events, Tags, Reports, Limits, Instances, Spot Requests, Reserved Instances, Scheduled Instances, Commands, Dedicated Hosts, AMIs, and Load Balancing. The main content area displays a table of instances. The table has columns for Name, Instance ID, Instance Type, Availability Zone, Instance State, Status Checks, Alarm Status, and Public DNS. Three instances are listed:

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS
spark_10gb-master-i-eb35bb70	i-eb35bb70	c3.large	us-east-1d	running	2/2 checks passed	None	ec2-52-91-137-63.cc
spark_10gb-slave-1-i-9435bb0f	i-9435bb0f	c3.large	us-east-1d	running	2/2 checks passed	None	ec2-54-152-73-60.cc
	i-dfb1bd26	c3.large	us-east-1e	running	2/2 checks passed	None	ec2-54-84-207-44.cc

Below the table, a detailed view of the master instance (i-eb35bb70) is shown. The 'Description' tab is selected, displaying the following details:

- Instance ID: i-eb35bb70
- Instance state: running
- Instance type: c3.large
- Private DNS: ip-172-31-24-34.ec2.internal
- Private IPs: 172.31.24.34
- Secondary private IPs: vpc-f5f199b
- VPC ID: subnet-4d9d5215
- Subnet ID: subnet-4d9d5215
- Network interfaces: eth0
- Public DNS: ec2-52-91-137-63.compute-1.amazonaws.com
- Public IP: 52.91.137.63
- Elastic IP: -
- Availability zone: us-east-1d
- Security groups: spark_10gb-master, view rules
- Scheduled events: No scheduled events
- AMI ID: spark.ami.pvnm.v9 (ami-5bb18832)
- Platform: -
- IAM role: -

Cluster Formation Started:

```

ubuntu@ip-172-31-16-253:~/spark-1.6.1-bin-hadoop2.6/ec2
priyankpshah@priyankpshah-Lenovo-Y40-80:~/netbeans-7.0.1/bin x  ubuntu@ip-172-31-16-253:~/spark-1.6.1-bin-hadoop2.6/ec2
priyankpshah@priyankpshah-Lenovo-Y40-80:~/Downloads$ ssh -i "Spark.pem" ubuntu@ip-172-31-16-253.ubuntu@ip-172-31-16-253:~/spark-1.6.1-bin-hadoop2.6/ec2
Welcome to Ubuntu 14.04.3 LTS (GNU/Linux 3.13.0-74-generic x86_64)

 * Documentation:  https://help.ubuntu.com/
 * System information as of Sun Mar 20 08:43:06 UTC 2016
System load:  0.08      Processes:          124
Usage of /:   25.1% of 7.74GB   Users logged in:    1
Memory usage: 1%           IP address for eth0: 172.31.16.253
Swap usage:   0%
Graph this data and manage this system at:
  https://landscape.canonical.com/
Get cloud support with Ubuntu Advantage Cloud Guest:
  http://www.ubuntu.com/business/services/cloud

Last login: Sun Mar 20 08:42:42 2016 from 208-59-157-46.c3-0.mcm-ubr1.chi-mcm.il
.cable.rcn.com
ubuntu@ip-172-31-16-253:~$ ls
scala-2.10.4.tgz          spark-1.6.1-bin-hadoop2.6.tgz
spark-1.6.1-bin-hadoop2.6  Spark.pem
ubuntu@ip-172-31-16-253:~$ cd spark-1.6.1-bin-hadoop2.6/ec2/
ubuntu@ip-172-31-16-253:~/spark-1.6.1-bin-hadoop2.6/ec2$ ls
deploy_generic.lib  README  spark-ec2  spark_ec2.py  Spark.pem
ubuntu@ip-172-31-16-253:~/spark-1.6.1-bin-hadoop2.6/ec2$ chmod 600 Spark.pem
ubuntu@ip-172-31-16-253:~/spark-1.6.1-bin-hadoop2.6/ec2$ export AWS_ACCESS_KEY_ID=AKIAITC8GWRXTXULWCL
ubuntu@ip-172-31-16-253:~/spark-1.6.1-bin-hadoop2.6/ec2$ export AWS_SECRET_ACCESS_KEY=kDzV2c0xfd60jvKN/y0vNH5xyYlwCKWDnuSw
ubuntu@ip-172-31-16-253:~/spark-1.6.1-bin-hadoop2.6/ec2$ ./spark-ec2 -k Spark -i Spark.pem -s 1 --instance-type=d2.xlarge launch spark_runcluster
Setting up security groups...
Creating security group spark_runcluster-master
Creating security group spark_runcluster-slaves
Searching for existing cluster spark_runcluster in region us-east-1...
Spark AMI: ami-35b1885c
Launching instances...
Launched 1 slave in us-east-1e, regid = r-d58d0b2c
Launched master in us-east-1e, regid = r-c08c0a39
Waiting for AWS to propagate instance metadata...
Waiting for cluster to enter 'ssh-ready' state...

```

Namenode Format:

```

priyankpshah@priyankpshah-Lenovo-Y40-80: ~/Downloads
Done!
ubuntu@ip-172-31-40-105:~$ ls
64 gnsort-linux-1.5.tar.gz gpl-2.0.txt MANIFEST.MF scala-2.10.4.tgz spark-1.6.1 spark-1.6.1-bin-hadoop2.6.tgz Spark.pem sparkterasort.scala
ubuntu@ip-172-31-40-105:~$ [REDACTED]

16/03/27 17:42:10 INFO namenode.NameNode: STARTUP_MSG:
*****STARTUP MSG: Starting NameNode
STARTUP MSG: host = ip-172-31-24-34.ec2.internal/172.31.24.34
STARTUP MSG: args = [-format]
STARTUP MSG: version = 1.0.4
STARTUP MSG: build = https://svn.apache.org/repos/ast/hadoop/common/branches/branch-1.0 -r 1393290; compiled by 'hortonfo' on Wed Oct 3 05:13:58 UTC 2012
*****
16/03/27 17:42:10 INFO util.GSet: VM type      = 64-bit
16/03/27 17:42:10 INFO util.GSet: % max memory = 17.78 MB
16/03/27 17:42:10 INFO util.GSet: capacity     = 2^21 = 2097152 entries
16/03/27 17:42:10 INFO util.GSet: recommended=2097152, actual=2097152
16/03/27 17:42:10 INFO namenode.FSNamesystem: fsOwner=root
16/03/27 17:42:11 INFO namenode.FSNamesystem: supergroup=supergroup
16/03/27 17:42:11 INFO namenode.FSNamesystem: isPermissionEnabled=false
16/03/27 17:42:11 INFO namenode.FSNamesystem: dfs.block.invalidate.limit=100
16/03/27 17:42:11 INFO namenode.FSNamesystem: isAccessTokenEnabled=false accessKeyUpdateInterval=0 min(s), accessTokenLifetime=0 min(s)
16/03/27 17:42:11 INFO namenode.NameNode: Caching file names occurring more than 10 times
16/03/27 17:42:11 INFO common.Storage: Image file of size 110 saved in 0 seconds.
16/03/27 17:42:11 INFO common.Storage: Storage directory /mnt/raid/dfs/name has been successfully formatted.
16/03/27 17:42:11 INFO namenode.NameNode: SHUTDOWN_MSG:
*****
SHUTDOWN_MSG: Shutting down Namenode at ip-172-31-24-34.ec2.internal/172.31.24.34
root@ip-172-31-24-34 bin]$ [REDACTED]

16/03/27 17:21:58 INFO DataNucleus.Datastore: The class "org.apache.hadoop.hive.metastore.model.MFieldSchema" is tagged as "embedded-only" so does not have its own datastore table.
16/03/27 17:21:58 INFO DataNucleus.Datastore: The class "org.apache.hadoop.hive.metastore.model.MOrder" is tagged as "embedded-only" so does not have its own datastore table.
16/03/27 17:22:00 INFO DataNucleus.Datastore: The class "org.apache.hadoop.hive.metastore.model.MFieldSchema" is tagged as "embedded-only" so does not have its own datastore table.
16/03/27 17:22:00 INFO DataNucleus.Datastore: The class "org.apache.hadoop.hive.metastore.model.MOrder" is tagged as "embedded-only" so does not have its own datastore table.
16/03/27 17:22:01 INFO metastore.MetaStoreDirectSql: Using direct SQL, underlying DB is DERBY
16/03/27 17:22:01 WARN metastore.ObjectStore: Version information not found in metastore. hive.metastore.schema.verification is not enabled so recording the schema version 1.2.0
16/03/27 17:22:01 WARN metastore.ObjectStore: Failed to get database default, returning NoSuchObjectException
16/03/27 17:22:01 INFO metastore.HiveMetaStore: Added admin role in metastore
16/03/27 17:22:01 INFO metastore.HiveMetaStore: Added public role in metastore
16/03/27 17:22:02 INFO metastore.HiveMetaStore: No user is added in admin role, since config is empty
16/03/27 17:22:02 INFO metastore.HiveMetaStore: 0: get_all_databases
16/03/27 17:22:02 INFO HiveMetaStore.audit: ugi=root ip=unknown-ip-addr cmd=get_all_databases
16/03/27 17:22:02 INFO metastore.HiveMetaStore: 0: get_functions: db=default pat=
16/03/27 17:22:02 INFO HiveMetaStore.audit: ugi=root ip=unknown-ip-addr cmd=get_functions: db=default pat=
16/03/27 17:22:02 INFO DataNucleus.Datastore: The class "org.apache.hadoop.hive.metastore.model.MResourceUri" is tagged as "embedded-only" so does not have its own datastore table.

```

Service Started on master/slave:

```

priyankpshah@priyankpshah-Lenovo-Y40-80: /tmp
Done!
ubuntu@ip-172-31-40-105:~$ ls
64 gnsort-linux-1.5.tar.gz gpl-2.0.txt MANIFEST.MF scala-2.10.4.tgz spark-1.6.1 spark-1.6.1-bin-hadoop2.6.tgz Spark.pem sparkterasort.scala
ubuntu@ip-172-31-40-105:~$ [REDACTED]

starting namenode, logging to /mnt/ephemeral-hdfs/logs/hadoop-root-namenode-ip-172-31-24-34.ec2.internal.out
ec2-54-152-73-60.compute-1.amazonaws.com: Warning: $HADOOP_HOME is deprecated.
ec2-54-152-73-60.compute-1.amazonaws.com: starting datanode, logging to /mnt/ephemeral-hdfs/logs/hadoop-root-datanode-ip-172-31-21-41.ec2.internal.out
ec2-52-91-137-63.compute-1.amazonaws.com: Warning: $HADOOP_HOME is deprecated.
ec2-52-91-137-63.compute-1.amazonaws.com: starting secondarynamenode, logging to /mnt/ephemeral-hdfs/logs/hadoop-root-secondarynamenode-ip-172-31-24-34.ec2.internal.out
root@ip-172-31-24-34 bin]$ jps
28899 Jps
28840 SecondaryNameNode
28659 NameNode
27367 SparkSubmit
5768 TachyonMaster
root@ip-172-31-24-34 bin]$ cd ..
root@ip-172-31-24-34 ephemeral-hdfs]$ ls
bin   CHANGES.txt  docs  hadoop-core-1.0.4.jar  hadoop-test-1.0.4.jar  ivy.xml  LICENSE.txt  sbin  webapps
build.xml  conf  hadoop-ant-1.0.4.jar  hadoop-mapreduce-client-jobclient-1.0.4.jar  lib  NOTICE.txt  share
c++  contrib  hadoop-client-1.0.4.jar
root@ip-172-31-24-34 ephemeral-hdfs]$ cd ..
root@ip-172-31-24-34 ~$ ks
4885 Worker
26266 Jps
4758 DataNode
32 64 ephemeral-hdfs gnsort gnsort
5079 TachyonWorker
root@ip-172-31-24-34 ~$ cd spark
root@ip-172-31-24-34 spark]$ ls
5384 CoarseGrainedExecutorBackend
bin  CHANGES.txt  conf  data  ec2  example
root@ip-172-31-24-34 sbin]$ ls
4885 Worker
26359 DataNode
26438 TaskTracker
Slaves.sh
root@ip-172-31-24-34 sbin]$ ls
5079 TachyonWorker
26488 Jps
root@ip-172-31-24-34 sbin]$ ./start-all.sh
26611 Jps
root@ip-172-31-24-34 sbin]$ ./start-all.sh
5079 TachyonWorker
26725 Jps
root@ip-172-31-24-34 sbin]$ ./start-all.sh
26652 DataNode
5079 TachyonWorker
root@ip-172-31-21-41 ~$ jps
28840 SecondaryNameNode
28659 NameNode
28999 Jps
28927 Master
27367 SparkSubmit
5768 TachyonMaster
root@ip-172-31-24-34 sbin]$ [REDACTED]

spark  spark-ec2  tachyon
r.sh  stop-slaves.sh
service.sh  stop-thriftserver.sh
.sh

```

10 GB Job Completed:

Spark Jobs (2)

Total Uptime: 28 min
Scheduling Mode: FIFO
Completed Jobs: 2

Event Timeline

Completed Jobs (2)

Job Id	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
1	saveAsTextFile at <console>:26	2016/03/27 19:26:10	13 min	2/2	150/150
0	sortByKey at <console>:23	2016/03/27 19:18:11	2.6 min	1/1	75/75

10 GB output:

```
priyankpshah@priyankpshah-Lenovo-Y40-80: ~/Downloads
Master
blk_3656988163785466233_1001.meta
blk_3716376761093718544
blk_3716376761093718544_1001.meta
blk_392497644266332959_1001.meta
blk_392497644266332959_1001.meta
blk_3984962160257936664
blk_3984962160257936664_1001.meta
blk_44858899556672853
blk_44858899556672853_1001.meta
blk_4491235397312873455_1001.meta
blk_4491235397312873455_1001.meta
blk_-470145/26924485211
blk_-8801385180646698292
blk_-8801385180646698292_1001.meta
blk_8929071995644869269
blk_8929071995644869269_1001.meta
blk_-8991541316492150571
blk_-8991541316492150571_1001.meta
blk_9002528176370568702
blk_9002528176370568702_1001.meta
blk_-999942694838294186
blk_-999942694838294186_1001.meta
dncp_block_verification.log.curr
blk_-470145/26924485211
subdir0
root@ip-172-31-6-96:~$ current]$ cd ..
root@ip-172-31-6-96:~$ ls
blocksBeingWritten current detach in_use.lock lost+found storage
root@ip-172-31-6-96:~$ ls
total 44K
drwx----- 2 root root 16K Mar 27 18:33 lost+found
-rw-r--r-- 1 root root 0 Mar 27 18:39 in_use.lock
-rw-r--r-- 1 root root 157 Mar 27 18:39 storage
drwxr-xr-x 2 root root 4,0K Mar 27 18:39 tmp
drwxr-xr-x 2 root root 4,0K Mar 27 18:39 detach
drwxr-xr-x 66 root root 12K Mar 27 19:01 current
drwxr-xr-x 2 root root 4,0K Mar 27 19:02 blocksBeingWritten
root@ip-172-31-6-96:~$ cd
root@ip-172-31-6-96:~$ cd ephemeral-hdfs/bin/hadoop fs -ls /
-bash: cd: ephemeral-hdfs/bin/hadoop: Not a directory
root@ip-172-31-6-96:~$ ephemeral-hdfs/bin/hadoop fs -ls /
Warning: $HADOOP_HOME is deprecated.

Found 3 items
drwxr-xr-x - root supergroup 0 2016-03-27 18:55 /input
drwxr-xr-x - root supergroup 0 2016-03-27 19:04 /tmp
drwxr-xr-x - root supergroup 0 2016-03-27 18:56 /user
root@ip-172-31-13-143:~$ ephemeral-hdfs/bin/hadoop fs -ls /input/
Warning: $HADOOP_HOME is deprecated.

root@ip-172-31-13-143:~$ ephemeral-hdfs/bin/hadoop fs -ls /user/root/input
Warning: $HADOOP_HOME is deprecated.

Found 1 items
-rw-r--r-- 1 root supergroup 10000000000 2016-03-27 18:56 /user/root/input
root@ip-172-31-13-143:~$ ls
32 ephemeral-hdfs gpl-2.0.txt mapreduce scala spark-ec2
root@ip-172-31-13-143:~$ [REDACTED]
```

100 GB Experiment:

Amazon AWS Cluster:

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS
spark_100gb-master-i-7a525d83	i-7a525d83	c3.large	us-east-1e	running	2/2 checks passed	None	ec2-52-91-131-254
spark_100gb-slave-i-c5b25d32	i-c5b25d32	c3.large	us-east-1e	running	2/2 checks passed	None	ec2-54-172-80-220
spark_100gb-slave-i-cd515e34	i-cd515e34	c3.large	us-east-1e	running	2/2 checks passed	None	ec2-54-172-106-100
spark_100gb-slave-i-cf515e36	i-cf515e36	c3.large	us-east-1e	running	2/2 checks passed	None	ec2-54-172-82-99
spark_100gb-slave-i-c2525d3b	i-c2525d3b	c3.large	us-east-1e	running	2/2 checks passed	None	ec2-54-172-79-205
spark_100gb-slave-i-c4515e3d	i-c4515e3d	c3.large	us-east-1e	running	2/2 checks passed	None	ec2-52-91-117-204
spark_100gb-slave-i-c4525d3d	i-c4525d3d	c3.large	us-east-1e	running	2/2 checks passed	None	ec2-54-172-85-230
spark_100gb-slave-i-c6525d3f	i-c6525d3f	c3.large	us-east-1e	running	2/2 checks passed	None	ec2-54-172-9-12
spark_100gb-slave-i-c7515e3e	i-c7515e3e	c3.large	us-east-1e	running	2/2 checks passed	None	ec2-54-172-80-96
spark_100gb-slave-i-c9515e30	i-c9515e30	c3.large	us-east-1e	running	2/2 checks passed	None	ec2-54-172-83-194
spark_100gb-slave-i-4b525db2	i-4b525db2	c3.large	us-east-1e	running	2/2 checks passed	None	ec2-52-90-22-223
spark_100gb-slave-i-4d525db4	i-4d525db4	c3.large	us-east-1e	running	2/2 checks passed	None	ec2-54-172-78-92
spark_100gb-slave-i-4f525db6	i-4f525db6	c3.large	us-east-1e	running	2/2 checks passed	None	ec2-54-172-86-5
spark_100gb-slave-i-7f515e86	i-7f515e86	c3.large	us-east-1e	running	2/2 checks passed	None	ec2-54-152-60-138
spark_100gb-slave-i-41525db8	i-41525db8	c3.large	us-east-1e	running	2/2 checks passed	None	ec2-52-91-203-10
spark_100gb-slave-i-44525dbd	i-44525dbd	c3.large	us-east-1e	running	2/2 checks passed	None	ec2-54-172-81-31
spark_100gb-slave-i-47525dbe	i-47525dbe	c3.large	us-east-1e	running	2/2 checks passed	None	ec2-52-90-13-111

Cluster Configured and Started:

```
ubuntu@ip-172-31-21-244: ~
[1] 100gb@ip-172-31-21-244: ~ 185x51
ubuntu@ip-172-31-21-244: ~ 185x51
Shutting down GANGLIA gmond: [FAILED]
Starting GANGLIA gmond: [OK]
Connection to ec2-54-164-87-239.compute-1.amazonaws.com closed.
Shutting down GANGLIA gmond: [FAILED]
Starting GANGLIA gmond: [OK]
Connection to ec2-52-87-171-142.compute-1.amazonaws.com closed.
Shutting down GANGLIA gmond: [FAILED]
Starting GANGLIA gmond: [OK]
Connection to ec2-54-88-154-88.compute-1.amazonaws.com closed.
Shutting down GANGLIA gmond: [FAILED]
Starting GANGLIA gmond: [OK]
Connection to ec2-52-90-23-197.compute-1.amazonaws.com closed.
Shutting down GANGLIA gmond: [FAILED]
Starting GANGLIA gmond: [OK]
Connection to ec2-54-88-97-104.compute-1.amazonaws.com closed.
Shutting down GANGLIA gmond: [FAILED]
Starting GANGLIA gmond: [OK]
Connection to ec2-52-90-166-78.compute-1.amazonaws.com closed.
Shutting down GANGLIA gmond: [FAILED]
Starting GANGLIA gmond: [OK]
Connection to ec2-52-201-213-135.compute-1.amazonaws.com closed.
Shutting down GANGLIA gmond: [FAILED]
Starting GANGLIA gmond: [OK]
Connection to ec2-54-89-120-241.compute-1.amazonaws.com closed.
Shutting down GANGLIA gmond: [FAILED]
Starting GANGLIA gmond: [OK]
Connection to ec2-52-23-238-31.compute-1.amazonaws.com closed.
Shutting down GANGLIA gmond: [FAILED]
Starting GANGLIA gmond: [OK]
Connection to ec2-52-91-242-113.compute-1.amazonaws.com closed.
Shutting down GANGLIA gmond: [FAILED]
Starting GANGLIA gmond: [OK]
Connection to ec2-52-91-217-145.compute-1.amazonaws.com closed.
Shutting down GANGLIA gmond: [FAILED]
Starting GANGLIA gmond: [OK]
Connection to ec2-52-91-171-34.compute-1.amazonaws.com closed.
Shutting down GANGLIA gmond: [FAILED]
Starting GANGLIA gmond: [OK]
Connection to ec2-52-87-174-178.compute-1.amazonaws.com closed.
Shutting down GANGLIA gmond: [FAILED]
Starting GANGLIA gmond: [OK]
Connection to ec2-52-90-53-113.compute-1.amazonaws.com closed.
Shutting down GANGLIA gmond: [FAILED]
Starting GANGLIA gmond: [OK]
Connection to ec2-54-152-200-20.compute-1.amazonaws.com closed.
Shutting down GANGLIA gmond: [FAILED]
Starting GANGLIA gmond: [OK]
Connection to ec2-54-86-148-25.compute-1.amazonaws.com closed.
Shutting down GANGLIA gmetad: [FAILED]
Starting GANGLIA gmetad: [OK]
Stopping httpd: [FAILED]
```

Datanode Running on 16 Slaves:

```
priyankpshah@priyankpshah-Lenovo-Y40-80: ~/Downloads
root@ip-172-31-37-44 bin]$ ./start-dfs.sh
=====
root@ip-172-31-37-44 bin]$ ./start-dfs.sh
Warning: $HADOOP_HOME is deprecated.

starting namenode, logging to /mnt/ephemeral-hdfs/logs/hadoop-root-namenode-ip-172-31-37-44.ec2.internal.out
ec2-52-3-242-138.compute-1.amazonaws.com: Warning: $HADOOP_HOME is deprecated.
ec2-52-91-125-212.compute-1.amazonaws.com: Warning: $HADOOP_HOME is deprecated.
ec2-54-84-11-133.compute-1.amazonaws.com: Warning: $HADOOP_HOME is deprecated.
ec2-52-91-122-219.compute-1.amazonaws.com: Warning: $HADOOP_HOME is deprecated.
ec2-52-91-160-197.compute-1.amazonaws.com: Warning: $HADOOP_HOME is deprecated.
ec2-54-86-44-186.compute-1.amazonaws.com: Warning: $HADOOP_HOME is deprecated.
ec2-52-87-153-96.compute-1.amazonaws.com: Warning: $HADOOP_HOME is deprecated.
ec2-52-90-22-89.compute-1.amazonaws.com: Warning: $HADOOP_HOME is deprecated.
ec2-52-3-220-126.compute-1.amazonaws.com: Warning: $HADOOP_HOME is deprecated.
ec2-52-91-124-95.compute-1.amazonaws.com: Warning: $HADOOP_HOME is deprecated.
ec2-52-91-126-226.compute-1.amazonaws.com: Warning: $HADOOP_HOME is deprecated.
ec2-52-91-135-185.compute-1.amazonaws.com: Warning: $HADOOP_HOME is deprecated.
ec2-54-84-75-37.compute-1.amazonaws.com: Warning: $HADOOP_HOME is deprecated.
ec2-52-90-13-73.compute-1.amazonaws.com: Warning: $HADOOP_HOME is deprecated.
ec2-54-84-55-131.compute-1.amazonaws.com: Warning: $HADOOP_HOME is deprecated.
ec2-52-3-253-122.compute-1.amazonaws.com: Warning: $HADOOP_HOME is deprecated.
ec2-54-84-75-37.compute-1.amazonaws.com: starting datanode, logging to /mnt/ephemeral-hdfs/logs/hadoop-root-datanode-ip-172-31-43-25.ec2.internal.out
ec2-52-91-125-212.compute-1.amazonaws.com: starting datanode, logging to /mnt/ephemeral-hdfs/logs/hadoop-root-datanode-ip-172-31-36-86.ec2.internal.out
ec2-54-86-44-186.compute-1.amazonaws.com: starting datanode, logging to /mnt/ephemeral-hdfs/logs/hadoop-root-datanode-ip-172-31-33-253.ec2.internal.out
ec2-52-91-160-197.compute-1.amazonaws.com: starting datanode, logging to /mnt/ephemeral-hdfs/logs/hadoop-root-datanode-ip-172-31-33-0.ec2.internal.out
ec2-52-91-122-219.compute-1.amazonaws.com: starting datanode, logging to /mnt/ephemeral-hdfs/logs/hadoop-root-datanode-ip-172-31-45-68.ec2.internal.out
ec2-52-90-22-89.compute-1.amazonaws.com: starting datanode, logging to /mnt/ephemeral-hdfs/logs/hadoop-root-datanode-ip-172-31-47-65.ec2.internal.out
ec2-52-3-220-126.compute-1.amazonaws.com: starting datanode, logging to /mnt/ephemeral-hdfs/logs/hadoop-root-datanode-ip-172-31-33-130.ec2.internal.out
ec2-52-91-124-95.compute-1.amazonaws.com: starting datanode, logging to /mnt/ephemeral-hdfs/logs/hadoop-root-datanode-ip-172-31-42-219.ec2.internal.out
ec2-52-91-135-185.compute-1.amazonaws.com: starting datanode, logging to /mnt/ephemeral-hdfs/logs/hadoop-root-datanode-ip-172-31-46-43.ec2.internal.out
ec2-54-84-11-133.compute-1.amazonaws.com: starting datanode, logging to /mnt/ephemeral-hdfs/logs/hadoop-root-datanode-ip-172-31-32-227.ec2.internal.out
ec2-52-87-153-96.compute-1.amazonaws.com: starting datanode, logging to /mnt/ephemeral-hdfs/logs/hadoop-root-datanode-ip-172-31-32-66.ec2.internal.out
ec2-54-84-55-131.compute-1.amazonaws.com: starting datanode, logging to /mnt/ephemeral-hdfs/logs/hadoop-root-datanode-ip-172-31-36-54.ec2.internal.out
ec2-52-3-253-122.compute-1.amazonaws.com: starting datanode, logging to /mnt/ephemeral-hdfs/logs/hadoop-root-datanode-ip-172-31-37-143.ec2.internal.out
```

Workers:

Google Chrome

Spark Master at spark://ec2-52-90-154-182.compute-1.amazonaws.com:7077 - Google Chrome

EC2 Management ▾ Spark Master at ▾

Spark 1.6.1

Spark Master at spark://ec2-52-90-154-182.compute-1.amazonaws.com:7077

URL: spark://ec2-52-90-154-182.compute-1.amazonaws.com:7077
 REST URL: spark://ec2-52-90-154-182.compute-1.amazonaws.com:6066 (cluster mode)

Alive Workers: 16

Cores in use: 32 Total, 32 Used
 Memory in use: 42.7 GB Total, 38.3 GB Used
 Applications: 1 Running, 0 Completed
 Drivers: 0 Running, 0 Completed
 Status: ALIVE

Workers

Worker Id	Address	State	Cores	Memory
worker-20160328035622-17231.1.235-50708	172.31.10.15:7441	ALIVE	2 (2 Used)	2.7 GB (2.4 GB Used)
worker-20160328035622-17231.10.15-57441	172.31.12.81:55563	ALIVE	2 (2 Used)	2.7 GB (2.4 GB Used)
worker-20160328035622-17231.13.82-46320	172.31.13.82:46320	ALIVE	2 (2 Used)	2.7 GB (2.4 GB Used)
worker-20160328035622-17231.2.69-46864	172.31.2.69:46864	ALIVE	2 (2 Used)	2.7 GB (2.4 GB Used)
worker-20160328035622-17231.3.239-50805	172.31.3.239:50805	ALIVE	2 (2 Used)	2.7 GB (2.4 GB Used)
worker-20160328035622-17231.3.255-35793	172.31.3.255:35793	ALIVE	2 (2 Used)	2.7 GB (2.4 GB Used)
worker-20160328035622-17231.4.94-49839	172.31.4.94:49839	ALIVE	2 (2 Used)	2.7 GB (2.4 GB Used)
worker-20160328035622-17231.5.252-32944	172.31.5.252:32944	ALIVE	2 (2 Used)	2.7 GB (2.4 GB Used)
worker-20160328035622-17231.5.65-43795	172.31.5.65:43795	ALIVE	2 (2 Used)	2.7 GB (2.4 GB Used)
worker-20160328035622-17231.6.144-36399	172.31.6.144:36399	ALIVE	2 (2 Used)	2.7 GB (2.4 GB Used)
worker-20160328035622-17231.6.163-45128	172.31.6.163:45128	ALIVE	2 (2 Used)	2.7 GB (2.4 GB Used)
worker-20160328035622-17231.6.27-44034	172.31.6.27:44034	ALIVE	2 (2 Used)	2.7 GB (2.4 GB Used)
worker-20160328035622-17231.7.239-44945	172.31.7.239:44945	ALIVE	2 (2 Used)	2.7 GB (2.4 GB Used)
worker-20160328035622-17231.8.114-34180	172.31.8.114:34180	ALIVE	2 (2 Used)	2.7 GB (2.4 GB Used)
worker-20160328035622-17231.8.89-38423	172.31.8.89:38423	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-20160328044609-0000	(kill) Spark shell	32	2.4 GB	2016/03/28 04:46:09	root	RUNNING	3.9 min

Completed Applications

Job Finished on 16 nodes:

Event Timeline

Jobs

Job Id	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
1	saveAsTextFile at <console>-26	2016/03/28 04:50:58	9.6 min	2/2	1490/1490
0	sortByKey at <console>-23	2016/03/28 04:49:36	49 s	1/1	745/745

Output from first File:

3. Shared Memory Benchmarking:

Shared Memory is basically program written in java for sorting the data generated by gensort. Purpose of writing and executing this program is to understand the concept of multi-threading and how execution of large files affected w.r.t. time complexity and space complexity. Goal of developing this program is to optimize it as much as possible, to achieve least execution time.

Methodology:

- In this program, I am taking value of threads as an input from user. User of this program have to manually generate the data in side /mnt/raid/ path with the name of file as "input" and empty file as "output" on given path. Temporary files will be generated on system path.
- I am using quick sort approach for sorting the key. First of all program will fetch the data in chunks of 20 Mb and pass it to quicksort function which applies quick sort.
- After performing quick sort on chunk it will be stored as temp file inside system's tmp folder. Once quick sort is performed and tmp files are generated, it will gather tmp files and sort it with merge sort which check one by one line of each file and put the in ascending order.
- In order to run the program for 10 GB dataset, I have increased to system limit of total number of open files to 10000 for safer side. Which can be check with "ulimit -a" command and can be set with "ulimit -n LIMIT" command, where LIMIT is the number of files you want to open at a time.
- Temp files will be automatically deleted when system exits the program run.

2. Performance Evaluation:

1 node Comparison:
Data Size: 10 GB

	1 Node Time	Speed Up	Throughput
Hadoop	836 Sec	1.00	97.990 Mbps
Spark	936 Sec	1.11	87.521 Mbps
Shared Memory	940 Sec	1.12	87.149 Mbps

16 node Comparison:
Data Size: 100 GB

	Node Time	Speed Up	Throughput
Hadoop	1622 Sec	1.00	505.055 Mbps
Spark	625 Sec	0.38	1310.720 Mbps

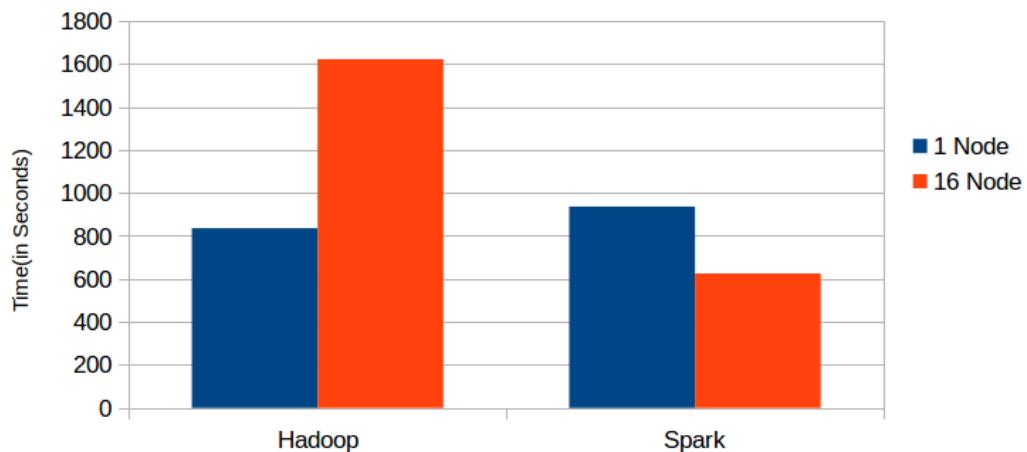
Shared Memory:

	Node Time	Speed Up	Throughput
1 Thread	940 Sec	1.00	87.149 Mbps
2 Thread	907 Sec	0.95	90.320 Mbps
4 Thread	946 Sec	1.08	86.598 Mbps
8 Thread	970 Sec	1.10	84.454 Mbps

Spark Vs Hadoop:

Execution Time Comparision

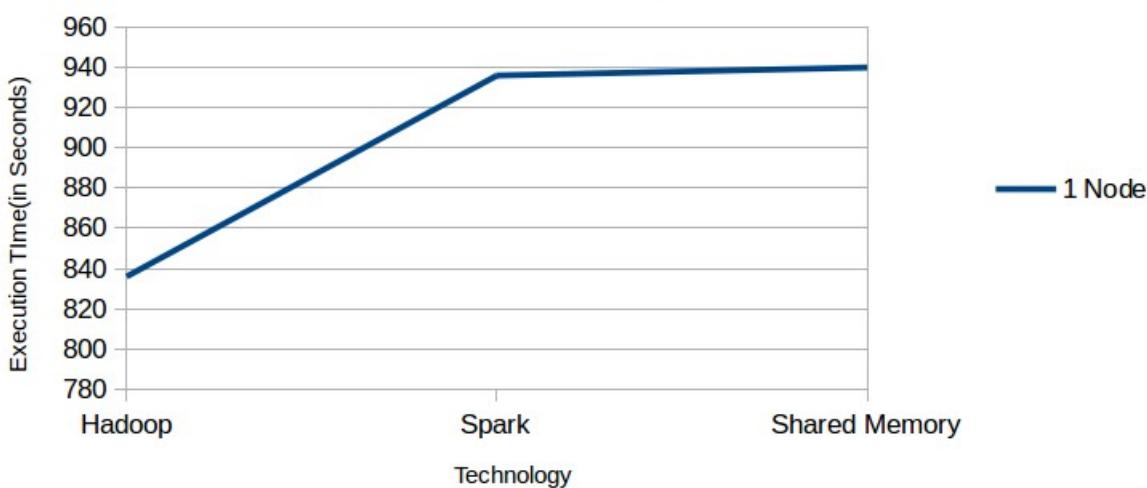
Hadoop Vs Spark

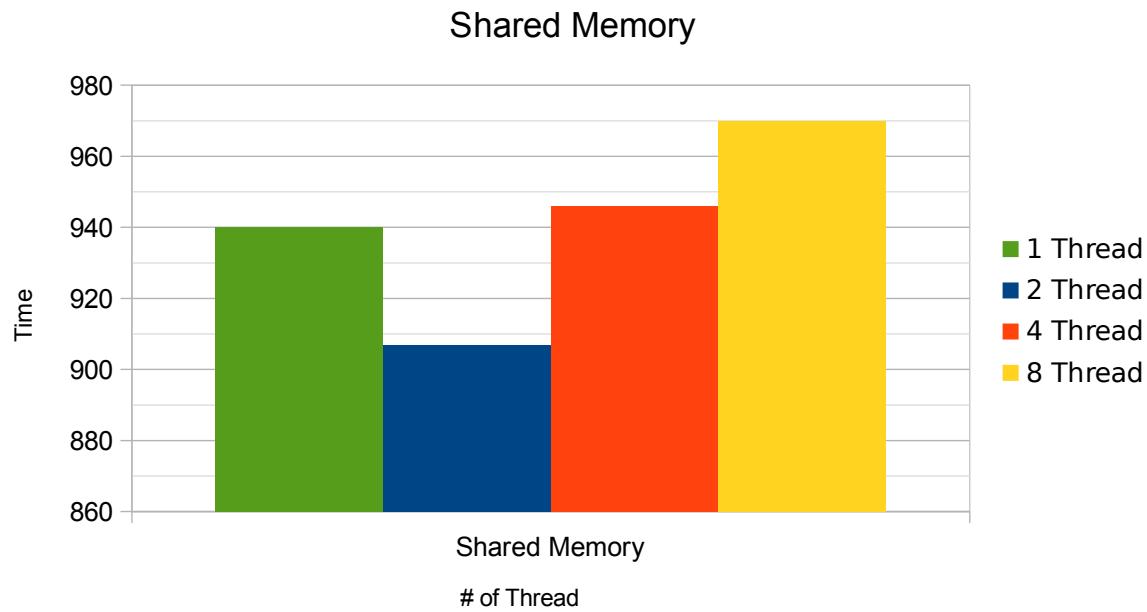
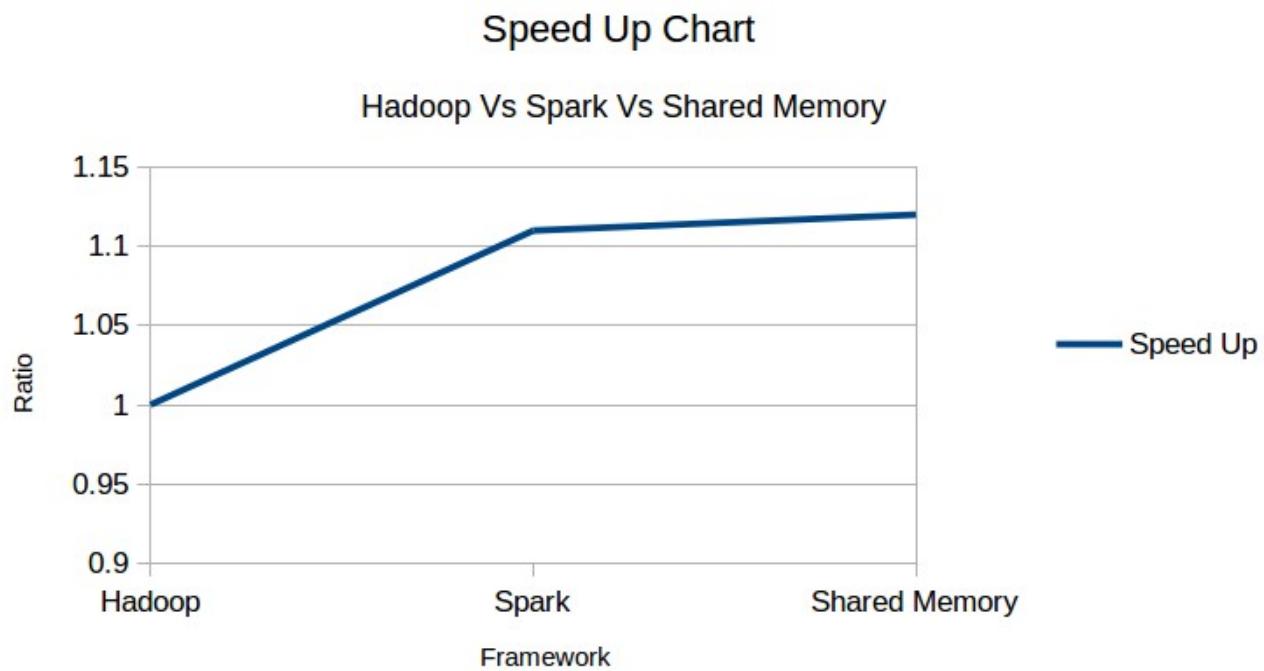
**Hadoop Vs Spark Vs Shared Memory:**

Below Chart shows the comparison between Hadoop , Spark and Shared Memory for 10 GB Data Set:

Hadoop vs Spark vs Shared Memory Experiment

10 GB dataset - A Node



Shared Memory Performance Comparison:**Speed Up Comparison:**

Conclusion:

- From this experiment I conclude that, In Spark Program runs on 16 Node much faster than 1 node.
- Initially, in 10 GB experiment Spark program was bit slower but over the time Spark results outnumbered the Hadoop Results.
- In Shared Memory, program with 2 thread runs faster than the program for 1 thread. Program for 4 node runs slower than the 2 nodes because c3.large have only 2 cores. Program for 8 thread runs even more slower than the 4 thread programs.
- Overall, Hadoop took least time for 10 GB Experiment. Spark took least time for 100 GB experiment. Shared Memory with 2 thread took the least time in 1,2,4 and 8 thread experiments.
- Overall Shared Memory is the best in case of 10 GB. It has the highest speed up ratio.
- Spark runs 100 times faster than the Hadoop on memory and 10 times faster when running with the disk. So at 100 Node Spark will be more efficient in management and execution.
- At 1000 nodes, Same will happen, Spark will perform better than the hadoop but required much more memory for intermediate data. So its better to with Hadoop for 1000 node.

References:

- 1) <https://hadoop.apache.org/docs/r1.2.1/api/org/apache/hadoop/mapred/Mapper.html>
- 2) <https://hadoop.apache.org/docs/r2.7.1/api/org/apache/hadoop/examples/terasort/package-summary.html>
- 3) <http://www.itadmintools.com/2012/10/throughput-calculator.html>
- 4) <http://tldp.org/LDP/solrhe/Securing-Optimizing-Linux-RH-Edition-v1.3/x4733.html>
- 5) <http://stackoverflow.com/questions/1318980/how-to-iterate-over-a-treemap>
- 6) <http://tldp.org/LDP/solrhe/Securing-Optimizing-Linux-RH-Edition-v1.3/x4733.html>