

CS553: Cloud Computing

Programming Assignment 2: Sorting Application

Assignment Report

cwid: A20327606

INDEX

1. Shared Memory

- 1.1 Introduction
- 1.2 Experiment 1: 1GB (c3.large)
- 1.3 Experiment 2: 10GB (c3.large)
- 1.4 Observations and graphs
- 1.4 Experiment 3: 1TB (d2.xlarge) **Extra credit**

2. Hadoop

- 1.1 Introduction
- 1.2 Experiment 1: 10 GB (single node)
- 1.3 Experiment 2: 100 GB (multinode)

3. Spark

- 1.1 Introduction
- 1.2 Experiment 1: 10 GB (single node)
- 1.3 Experiment 2: 100 GB (multinode)

4. Performance Evaluation and conclusions

Shared Memory

1.1 Introduction

Problem Statement:

The goal of this experiment is to sort a large dataset which contains one record per line of 100 bytes and we need to sort them on the basis of ASCII values of first 10 bytes. The major challenge here is the memory size. Since we have a very less memory which is much smaller value than the total file size.

Methodology:

The logic here is to divide the total number of records into small chunks which can fit and be sorted within memory space. Thus I have implemented external merge sort which first divides the size into smaller chunks, then individually sorts it and finally merges all the chunks. For sorting the individual chunks, I have used quick sort.

Environment details:

Type of EC2 instance used:	c3.large
Region:	N. Virginia
OS:	Ubuntu Server 14.04 LTS (HVM), SSD Volume Type – ami-fce3c696
Implementation Language:	JAVA
JAVA version	java version "1.7.0_95" OpenJDK Runtime Environment (IcedTea 2.6.4) (7u95-2.6.4-0ubuntu0.14.04.1) OpenJDK 64-Bit Server VM (build 24.95-b01, mixed mode)

Installation and experiment running steps:

1. Connect to AWS instance through ssh terminal.
2. check the names of disks using “lslbk” command (xvdb,xvdc and xvdd)
3. mount the disks on a folder using the script mount.sh
4. transfer the files (java files, gensort and valsrt) to the mounted location
5. run the gensort to generate input file of required size using following command:

./gensort -a num_of_records input.txt

where, num_of_records = 10000000 for 1GB
= 1000000000 for 10 GB
= 10000000000 for 1TB

6. now run the script SharedMemory.sh which would install java, compile the code, run it and then save the output to log.txt

7. (optional) The code can also be run without using script by directly running the following commands:

```
javac FileSorter.java  
java -Xmx3G FileSorter
```

8. Now the output file is generated in the same folder which contains sorted records

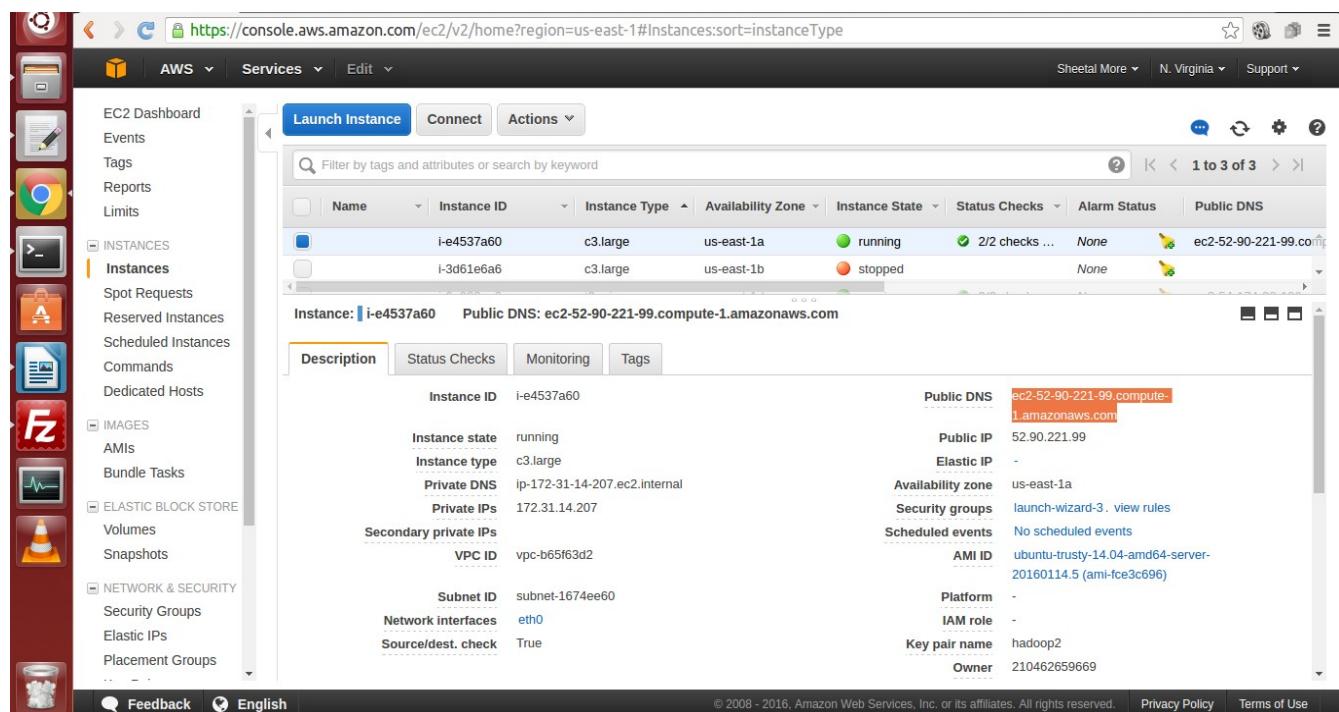
9. in order to verify the code, run the valsrt using command:

```
./valsrt output.txt
```

10. The head and tail of the output file can be found using head and tail command as shown in screenshots below:

1.2 Experiment 1 with 1GB Dataset

1. Aws with ipaddress:



The screenshot shows the AWS Management Console interface for the EC2 service. The left sidebar has a red background and lists various services: EC2 Dashboard, Events, Tags, Reports, Limits, Instances, Spot Requests, Reserved Instances, Scheduled Instances, Commands, Dedicated Hosts, Images, AMIs, Bundle Tasks, Elastic Block Store, Volumes, Snapshots, Network & Security, Security Groups, Elastic IPs, Placement Groups, and Feedback. The main content area is titled "Instances" and shows a table of running instances. One instance is selected, showing its detailed configuration. The instance details include:

Instance ID	Name	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS
i-e4537a60		c3.large	us-east-1a	running	2/2 checks ...	None	ec2-52-90-221-99.compute-1.amazonaws.com
i-3d61e6a6		c3.large	us-east-1b	stopped		None	

The selected instance (i-e4537a60) is shown in more detail with the following configuration:

Description	Value
Instance ID	i-e4537a60
Instance state	running
Instance type	c3.large
Private DNS	ip-172-31-14-207.ec2.internal
Private IPs	172.31.14.207
Secondary private IPs	
VPC ID	vpc-b65f63d2
Subnet ID	subnet-1674ee60
Network interfaces	eth0
Source/dest. check	True
Public DNS	ec2-52-90-221-99.compute-1.amazonaws.com
Public IP	52.90.221.99
Elastic IP	-
Availability zone	us-east-1a
Security groups	launch-wizard-3, view rules
Scheduled events	No scheduled events
AMI ID	ubuntu-trusty-14.04-amd64-server-20160114.5 (ami-fce3c696)
Platform	-
IAM role	-
Key pair name	hadoop2
Owner	210462659669

2. gensort

```
ubuntu@ip-172-31-14-207:/mnt/raid$ ./gensort -a 10000000 input1GB.txt
ubuntu@ip-172-31-14-207:/mnt/raid$ ls
FileSorter.java  gensort  input1GB.txt  lost+found  valsort
ubuntu@ip-172-31-14-207:/mnt/raid$
```

3. running java program:

a) 4 threads: error screenshot for out of memort heap space

```
Writer:
ubuntu@ip-172-31-14-207:/mnt/raid$ ./gensort -a 10000000 input1GB.txt
ubuntu@ip-172-31-14-207:/mnt/raid$ ls
FileSorter.java  gensort  input1GB.txt  lost+found  valsort
ubuntu@ip-172-31-14-207:/mnt/raid$ javac FileSorter.java
ubuntu@ip-172-31-14-207:/mnt/raid$ java FileSorter
Exception in thread "Thread-3" java.lang.OutOfMemoryError: Java heap space
Exception in thread "Thread-1" java.lang.OutOfMemoryError: Java heap spacejava.io.IOException: Stream closed

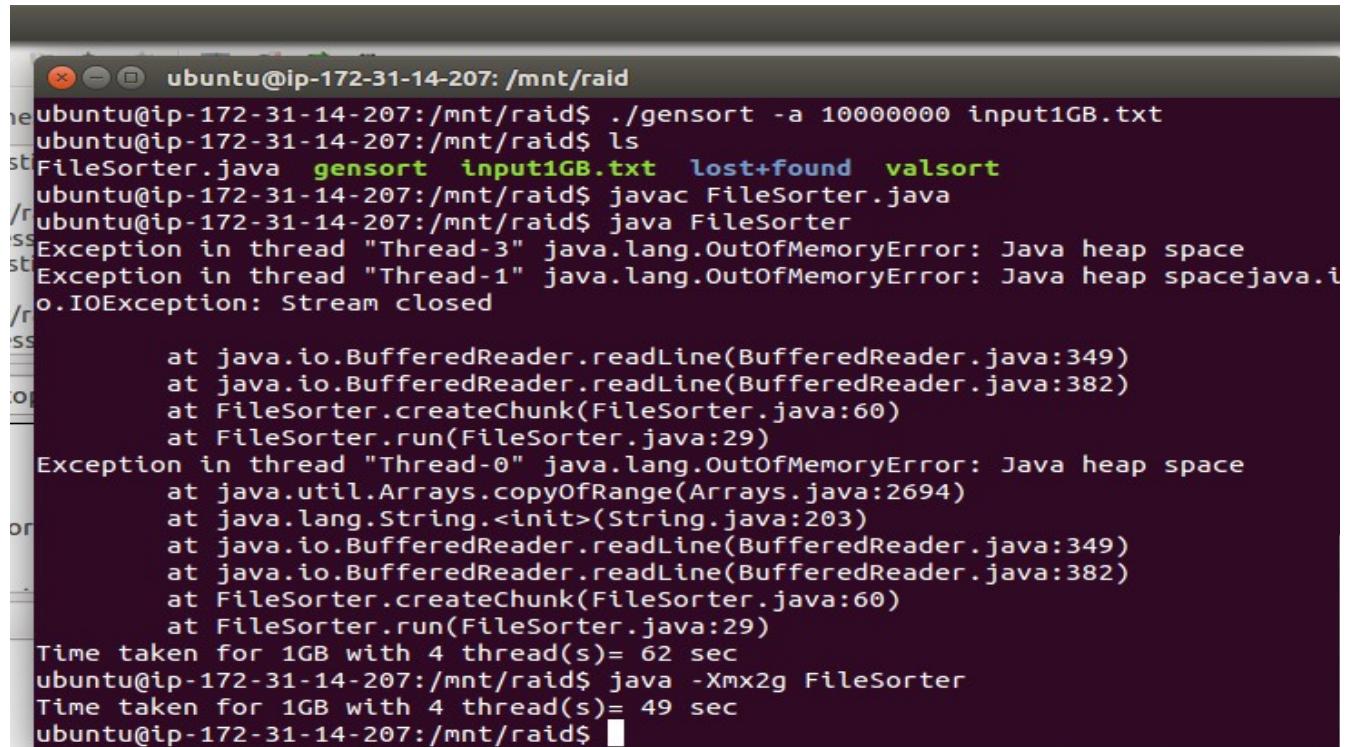
        at java.io.BufferedReader.readLine(BufferedReader.java:349)
        at java.io.BufferedReader.readLine(BufferedReader.java:382)
        at FileSorter.createChunk(FileSorter.java:60)
        at FileSorter.run(FileSorter.java:29)
Exception in thread "Thread-0" java.lang.OutOfMemoryError: Java heap space
        at java.util.Arrays.copyOfRange((Arrays.java:2694)
        at java.lang.String.<init>(String.java:203)
        at java.io.BufferedReader.readLine(BufferedReader.java:349)
        at java.io.BufferedReader.readLine(BufferedReader.java:382)
        at FileSorter.createChunk(FileSorter.java:60)
        at FileSorter.run(FileSorter.java:29)
Time taken for 1GB with 4 thread(s)= 62 sec
ubuntu@ip-172-31-14-207:/mnt/raid$ java -Xmx2g FileSorter
```

As seen in above document, I got an error due to heap space which I then solved by increasing the heap size. The command used is as follows:

```
java -Xmx2g FileSorter
```

Thus allocating 2GB RAM which solved the problem and I got the output as follows:

Actual time: 49 secs



```
ubuntu@ip-172-31-14-207:/mnt/raid$ ./gensort -a 10000000 input1GB.txt
ubuntu@ip-172-31-14-207:/mnt/raid$ ls
FileSorter.java  gensort  input1GB.txt  lost+found  valsort
ubuntu@ip-172-31-14-207:/mnt/raid$ javac FileSorter.java
ubuntu@ip-172-31-14-207:/mnt/raid$ java FileSorter
Exception in thread "Thread-3" java.lang.OutOfMemoryError: Java heap space
Exception in thread "Thread-1" java.lang.OutOfMemoryError: Java heap space
java.io.IOException: Stream closed
        at java.io.BufferedReader.readLine(BufferedReader.java:349)
        at java.io.BufferedReader.readLine(BufferedReader.java:382)
        at FileSorter.createChunk(FileSorter.java:60)
        at FileSorter.run(FileSorter.java:29)
Exception in thread "Thread-0" java.lang.OutOfMemoryError: Java heap space
        at java.util.Arrays.copyOfRange((Arrays.java:2694)
        at java.lang.String.<init>(String.java:203)
        at java.io.BufferedReader.readLine(BufferedReader.java:349)
        at java.io.BufferedReader.readLine(BufferedReader.java:382)
        at FileSorter.createChunk(FileSorter.java:60)
        at FileSorter.run(FileSorter.java:29)
Time taken for 1GB with 4 thread(s)= 62 sec
ubuntu@ip-172-31-14-207:/mnt/raid$ java -Xmx2g FileSorter
Time taken for 1GB with 4 thread(s)= 49 sec
ubuntu@ip-172-31-14-207:/mnt/raid$
```

b) 2 threads:

```
ubuntu@ip-172-31-14-207: /mnt/raid$ javac FileSorter.java
ubuntu@ip-172-31-14-207: /mnt/raid$ java -Xmx3g FileSorter
Time taken for 1GB with 2 thread(s)= 43 sec
ubuntu@ip-172-31-14-207: /mnt/raid$ ls
FileSorter.class  gensort      log.txt      output1GB.txt
FileSorter.java   input1GB.txt  lost+found  valsrt
ubuntu@ip-172-31-14-207: /mnt/raid$ ./valsrt output1GB.txt
Records: 10000000
Checksum: 4c499aee14ae2b
Duplicate keys: 0
SUCCESS - all records are in order
ubuntu@ip-172-31-14-207: /mnt/raid$
```

3. 1 thread:

```
ubuntu@ip-172-31-14-207: /mnt/raid$ javac FileSorter.java
ubuntu@ip-172-31-14-207: /mnt/raid$ java -Xmx3g FileSorter
Time taken for 1GB with 1 thread(s)= 37 sec
ubuntu@ip-172-31-14-207: /mnt/raid$ ./valsrt output1GB.txt
Records: 10000000
Checksum: 4c499aee14ae2b
Duplicate keys: 0
SUCCESS - all records are in order
ubuntu@ip-172-31-14-207: /mnt/raid$
```

head and tail

```

ubuntu@ip-172-31-14-207:/mnt/raid$ ls
FileSorter.class FileSorter.java gensort input1GB.txt log.txt lost+found output1GB.txt valsrt
ubuntu@ip-172-31-14-207:/mnt/raid$ head output1GB.txt
    "Oluve 000000000000000000000000000000001228D4 77778888000022224444D0DDDDDEEEE0000000000CCCC7777DDDD
, K4a-:v 000000000000000000000000000000001B8132 5555EEE8888999444FFFF1111CCCEEE1111EEEE6666FFFF
.FuD\ju 00000000000000000000000000000000797631 5555DDBBBB0000777222211122224444D0DDDD099996666
;5Yhtc 000000000000000000000000000000007D3DF5 2222AAACCCCFFFAAAA44445555EEE44442222DD09992222
=2C9{- 000000000000000000000000000000000809EE 5555D0D1111CCC9998BBB0000BBBBCCCCFFFC44443333
N)M9?sp 00000000000000000000000000000000429597 5555FFF00007775559991111CCCC66669999AAAEEE8888
P0X?rs 000000000000000000000000000000004162E 88883339999FFF1111CCCC8888CCC9999EEE0000003333
[xq\\$% 0000000000000000000000000000000097A5F0 6666666EEEDDD7777FFF00005555FFFFF88885551111
rAmQq4v 000000000000000000000000000000008180D3 BBBB111111119999FFF88884444CCCC
!&))Jf3; 000000000000000000000000000000004602E1 4444FFFCCCC88888888CCCFFFC4444CCCC
ubuntu@ip-172-31-14-207:/mnt/raid$ 
ubuntu@ip-172-31-14-207:/mnt/raid$ tail output1GB.txt
~~~%A NB_t 0000000000000000000000000000003DE5EC FFFF7777EEEBBBB4444EEEEEEE3333999900005555
~~~-c_Q(> 00000000000000000000000000000000832611 9999FFF1111777333777000111144444440000BBBB6666
~~~8Y)Fq1* 000000000000000000000000000000007424AC BBBB1111CCCCEE888800000007773333333DD22225555
~~~>D=QT 00000000000000000000000000000000674C0F 9999DD000055556666CCCC22220000FFFEEEDDDFFF0000
~~~]JA(j$ 000000000000000000000000000000006BCBE 111122223334445559999AAAABB89999FFFDDDBBBB3333
~~~]Zp,#/+ 000000000000000000000000000000003B95A CCC8888EEEAAEEE3333333777000FFFCCCC66667777
~~~_jOpix 0000000000000000000000000000000011E5D4 1111999911115555BBB1111000222EEE6666BBBB7777DDDD
~~~nt=ZH[N 00000000000000000000000000000000332A13 4444111BBBBB8888333777FFF44445555553330000CCCC
~~~s/Pq,-E 000000000000000000000000000000006B930 2222DDDDDD7777111EEECCCC7777BBB844488811111111
~~~zba_Tt 000000000000000000000000000000007F94F BBBBCCCC666655559999FFF8888AAA11116666AAAABB0000
ubuntu@ip-172-31-14-207:/mnt/raid$ 

```

1.3 Experiment 2: SharedMemory 10GB on c3.large instance

a) 1 thread:

```

ubuntu@ip-172-31-14-207:/mnt/raid$ ls
FileSorter.class FileSorter.java gensort input10GB.txt lost+found valsrt
ubuntu@ip-172-31-14-207:/mnt/raid$ javac FileSorter.java
ubuntu@ip-172-31-14-207:/mnt/raid$ java -Xmx3g FileSorter >> log.txt
ubuntu@ip-172-31-14-207:/mnt/raid$ 
ubuntu@ip-172-31-14-207:/mnt/raid$ ls
FileSorter.class FileSorter.java gensort input10GB.txt log.txt lost+found output10GB.txt valsrt
ubuntu@ip-172-31-14-207:/mnt/raid$ cat log.txt
Time taken for 10GB with 1 thread(s)= 665 sec
Records: 100000000
Checksum: 2faf20d000e8d8
Duplicate keys: 0
SUCCESS - all records are in order
ubuntu@ip-172-31-14-207:/mnt/raid$ 

```

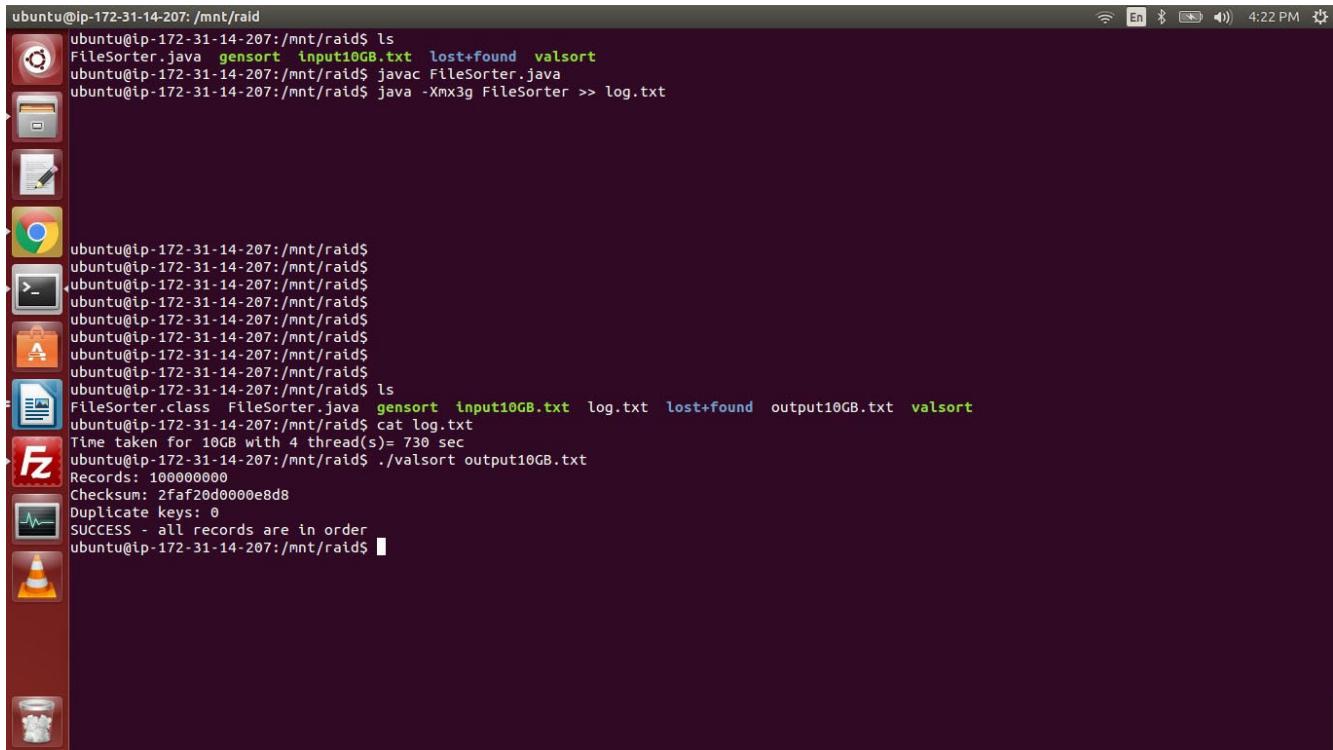
b) 2 threads:

```
ubuntu@ip-172-31-14-207:/mnt/raid$ ls
ubuntu@ip-172-31-14-207:/mnt/raid$ FileSorter.java gensort input10GB.txt lost+found valsort
ubuntu@ip-172-31-14-207:/mnt/raid$ javac FileSorter.java
ubuntu@ip-172-31-14-207:/mnt/raid$ java -Xmx3g FileSorter >> log.txt

ubuntu@ip-172-31-14-207:/mnt/raid$ ls
ubuntu@ip-172-31-14-207:/mnt/raid$ FileSorter.class FileSorter.java gensort input10GB.txt log.txt lost+found output10GB.txt valsort
ubuntu@ip-172-31-14-207:/mnt/raid$ cat log.txt
Time taken for 10GB with 2 thread(s): 659 sec
Records: 100000000
Checksum: 2faf2d0000e8d8
Duplicate keys: 0
SUCCESS - all records are in order
ubuntu@ip-172-31-14-207:/mnt/raid$
```

```
ubuntu@ip-172-31-14-207:/mnt/raid$ ls
ubuntu@ip-172-31-14-207:/mnt/raid$ FileSorter.java gensort input10GB.txt log.txt lost+found output10GB.txt valsort
ubuntu@ip-172-31-14-207:/mnt/raid$ cat log.txt
Time taken for 10GB with 2 thread(s): 659 sec
Records: 100000000
Checksum: 2faf2d0000e8d8
Duplicate keys: 0
SUCCESS - all records are in order
ubuntu@ip-172-31-14-207:/mnt/raid$ head output10GB.txt
    "0!uve 000000000000000000000000000000001228D4 7777888000022224444DDDDDDDEEE00000000CCCC7777DDDD
    PMD32= 00000000000000000000000000000000340CC1 FFFEEE6666CCCBBB9999333555DDDDDD77778886666
    ^3CO], 00000000000000000000000000000000158C5C5 5555AAA9999EEE88882229999CCCDD066665554442222
    !&3[J] 000000000000000000000000000000002145D78 8888BBBBDD01111CCC5556660BBBB1111FEEEDDD22229999
    !=U#, 9 0000000000000000000000000000000019072E3 33332222FFFFBBBB0000FFFAAA6666555333DD03333CCC
    !Of[[Tid 00000000000000000000000000000003CAAB4B 9999FFFF55553337777CCC4444BBBB7777EEEEE8888DDDD4444
    !f6Suy2 00000000000000000000000000000000000003ABFD8 EEEE55555556666AAA5555BBBBDD000011116666000000DD
    #KNIpq. 000000000000000000000000000000003B36FB9 111100003334441116666666AAAAAAA00001111CCCCEEE
    #'^c1'~ 000000000000000000000000000000002EDC5C8 8888AAA11114444FFFF77773333EEE44440000FFF9999999
    $"'Q)] 000000000000000000000000000000005F1265D CCCC6666EEE22220000DDDDAAA88886666BBBB00006666AAA
ubuntu@ip-172-31-14-207:/mnt/raid$ tail output10GB.txt
~~~ug2k#=U 000000000000000000000000000000002C06745 9999111DDDD22211110000FFFFEEEEEFFF33337777CCCC2222
~~~v/0&Qm 000000000000000000000000000000004789701 CCCCB8883333FFF000000000099991111FFF777744446666
~~~yK0l:gE 00000000000000000000000000000000204884F CCCCI111444488882226666BBBB888855557777EEE88880000
~~~yk^H_il 00000000000000000000000000000000463D064 44440000FFF333399944447777DDDFFFFAAA1118888000
~~~yl;C'XE 000000000000000000000000000000005BD0211 2222EEE33330000222111CCCCFFF555577774444BBBB6666
~~~zbA_ Tt 000000000000000000000000000000007F9F4F BBBBCCCC66665559999FFF8888AAA11116666AAAABB0000
~~~zeO^FEG 00000000000000000000000000000001E06130 4444CCCCBBB99992228885558888CCCCFFF00001111111
~~~}GxjhWHI 00000000000000000000000000000000CA1345 777711118888AAAAAA222111BBBB000222BBBCCC2222
~~~}P;jgqg 0000000000000000000000000000000040DA3E4 4444FFF444466663333EEE88888888DDDEEEE44442222DD
~~~}ku|k+p 0000000000000000000000000000005E4A0AA 0000666655551111BBBB88889999AAA55550000333355557777
ubuntu@ip-172-31-14-207:/mnt/raid$
```

c) 4 threads:



A screenshot of a Ubuntu desktop environment. On the left, there is a vertical dock containing icons for various applications: Dash, Home, Applications, Places, System, Help, Terminal, Nautilus, Gedit, Google Chrome, Terminal (another instance), FileSorter, FileSorter.class, and VLC. The main window shows a terminal session with the following command history:

```
ubuntu@ip-172-31-14-207:/mnt/raid$ ls
FileSorter.java  gensort  input10GB.txt  lost+found  valsrt
ubuntu@ip-172-31-14-207:/mnt/raid$ javac FileSorter.java
ubuntu@ip-172-31-14-207:/mnt/raid$ java -Xmx3g FileSorter >> log.txt

ubuntu@ip-172-31-14-207:/mnt/raid$ ls
ubuntu@ip-172-31-14-207:/mnt/raid$ FileSorter.class  FileSorter.java  gensort  input10GB.txt  log.txt  lost+found  output10GB.txt  valsrt
ubuntu@ip-172-31-14-207:/mnt/raid$ cat log.txt
Time taken for 10GB with 4 thread(s)= 730 sec
ubuntu@ip-172-31-14-207:/mnt/raid$ ./valsrt output10GB.txt
Records: 100000000
Checksum: 2faf20d0000e8d8
Duplicate keys: 0
SUCCESS - all records are in order
ubuntu@ip-172-31-14-207:/mnt/raid$
```

d) 8 threads

```

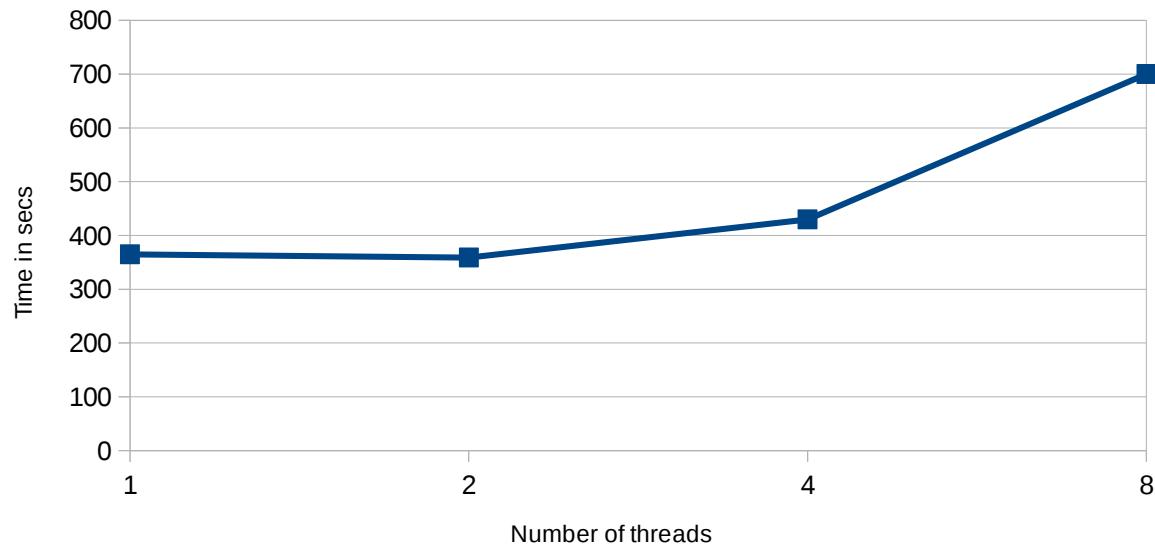
ubuntu@ip-172-31-14-207:/mnt/raid$ 
ubuntu@ip-172-31-14-207:/mnt/raid$ 
ubuntu@ip-172-31-14-207:/mnt/raid$ 
ubuntu@ip-172-31-14-207:/mnt/raid$ 
ubuntu@ip-172-31-14-207:/mnt/raid$ 
ubuntu@ip-172-31-14-207:/mnt/raid$ 
ubuntu@ip-172-31-14-207:/mnt/raid$ 
ubuntu@ip-172-31-14-207:/mnt/raid$ ls
FileSorter.class FileSorter.java gensort input10GB.txt log.txt lost+found output10GB.txt valsrt
ubuntu@ip-172-31-14-207:/mnt/raid$ cat log.txt
Time taken for 10GB with 8 thread(s) = 780 sec
ubuntu@ip-172-31-14-207:/mnt/raid$ ./valsrt output10GB.txt
Records: 100000000
Checksum: 2faf20d000e8d8
Duplicate keys: 0
SUCCESS - all records are in order
ubuntu@ip-172-31-14-207:/mnt/raid$ head output10GB.txt
    "0!uve 000000000000000000000000000000001228D4 7777888800002224444DDDDDDDEEE00000000CCCC7777DDDD
    PMd32= 0000000000000000000000000000003440CC1 FFFFE6666CCCCBBB89993335555DDDDDD777788886666
    ^3C0], 000000000000000000000000000000158C5C5 5555AAA9999EEE88822229999CCCCDDDD666655554442222
    !&3/J/] 00000000000000000000000000000002145D78 8888BBBBDD0111CCCC5555666BBBB1111EEEDDD022229999
    !=U#,9 0000000000000000000000000000000019072E3 33332222FFFFBBBB0000FFFAAAA6666555333DDDD333CCCC
    !of[ITd 0000000000000000000000000000003CAAB4B 9999FFF55553337777CCC4444BBBB7777EEEBBBBDDD4444
    !f6Suy2 0000000000000000000000000000003ABFD84 EEEE55555556666AAA5555BBBB0DD0000111666600000DD
    #&Nipq. 00000000000000000000000000000003B36FB9 11100003334441116666666AAAAAAA00001111CCCCEEE
    #'^cl'~ 0000000000000000000000000000002EDC5C8 8888AAA11114444FFF7773333EEE44440000FFF9999999
    $`-'Q)] 00000000000000000000000000000005F1265 DCCCC6666EEEE22220000DDDAAAA88886666BBBB00006666AAA
ubuntu@ip-172-31-14-207:/mnt/raid$ 
ubuntu@ip-172-31-14-207:/mnt/raid$ tail output10GB.txt
~~~uq2k#=U 00000000000000000000000000000002C06745 99991111DDDD22211110000FFFEEEEFFFF33337777CCCC2222
~~~v/0&Qm 00000000000000000000000000000004709701 CCCC88883333FFF0000000000099991111FFF77744446666
~~~ykol:gE 000000000000000000000000000000204884F CCCC1111444488882226666BBBB88885557777EEEBBBB0000
~~~yk^H.il 0000000000000000000000000000000463D004 44440000FFF333399944447777DDDDFFFAAA11188880DD
~~~yl;C'XE 00000000000000000000000000000005BD0211 2222EEE33330000222111CCCCFFF5557774444BBBB6666
~~~zba_Tt 00000000000000000000000000000007F9F4F BBBB8CCCC66665559999FFF8888AAA1116666AAAABB0000
~~~ze0^FEG 00000000000000000000000000001E06130 4444CCCCBBBB999922288885558888CCCCFFF00001111111
~~~}GxjhWHI 00000000000000000000000000000000CA1345 777711118888AAAAAA222111BBBB000222BBBBC2222
~~~}P;jgog 00000000000000000000000000000040DA3E4 4444FFF444466663333EEE88888888DDDEEEE44442222DD
~~~}ku|ksp 0000000000000000000000000000005E4A0AA 0000666655551111BBBB88889999AAA555500033355557777
ubuntu@ip-172-31-14-207:/mnt/raid$ 

```

Observations:

Number of threads	Time taken (secs)
1	365
2	359
4	430
8	700

Shared Memory 10GB c3.large instance



As seen above the least time required for sorting is 359 secs for 2 number of threads.

1.4: Experiment Shared Memory 1TB

The screenshot shows the AWS EC2 Management Console interface. On the left, there's a sidebar with various service icons. The main area displays a list of instances. Two instances are visible: one named `i-0749b1e68fdb509d3` which is `running`, and another named `i-0e7e1e12dea680998` which is `terminated`. Below the list, detailed information for the running instance is provided.

Attribute	Value
Instance ID	i-0749b1e68fdb509d3
Instance state	running
Instance type	d2.xlarge
Private DNS	ip-172-31-54-79.ec2.internal
Private IPs	172.31.54.79
VPC ID	vpc-f36329b
Subnet ID	subnet-5add6e70
Network interfaces	eth0
Source/dest. check	True
EBS-optimized	True
Root device type	ebs
Root device	/dev/xvda
Block device	/dev/xvdb
Public DNS	ec2-52-207-242-161.compute-1.amazonaws.com
Public IP	52.207.242.161
Elastic IP	-
Availability zone	us-east-1d
Security groups	launch-wizard-4, view rules
Scheduled events	No scheduled events
AMI ID	amzn-ami-hvm-2016.03.0.x86_64-gp2 (ami-08111162)
Platform	-
IAM role	-
Key pair name	akash
Owner	846027423990
Launch time	March 29, 2016 at 4:53:23 PM UTC-5 (less than one hour)
Termination protection	False
Lifecycle	normal
Monitoring	basic

The screenshot shows a terminal window on an Amazon Linux instance. The user has run the command `./gensort -a 10000000000 input1TB.txt`. The terminal output shows the progress of the sort operation, indicating it is currently at 10000000000 bytes.

```
ec2-user@ip-172-31-54-79:/mnt/raid$ ls
[ec2-user@ip-172-31-54-79 ~]$ ls
[ec2-user@ip-172-31-54-79 ~]$ cd /mnt/raid
[ec2-user@ip-172-31-54-79 raid]$ ls
gensort  lost+found  SharedMemory.sh  valsort
[ec2-user@ip-172-31-54-79 raid]$ chmod 777 gensort
[ec2-user@ip-172-31-54-79 raid]$ ./gensort -a 10000000000 input1TB.txt
```



```
ec2-user@ip-172-31-54-79:/mnt/raid
Records: 10000000000
Checksum: 15b72afc9567fda0c
Duplicate keys: 0
SUCCEED - all records are in order
[ec2-user@ip-172-31-54-79 raid]$ [ec2-user@ip-172-31-54-79 raid]$ [ec2-user@ip-172-31-54-79 raid]$ ls FileSorter.class FileSorter.java gensort input1TB.txt log.txt lost+found output1TB.txt SharedMemory.sh valsort [ec2-user@ip-172-31-54-79 raid]$ [ec2-user@ip-172-31-54-79 raid]$ [ec2-user@ip-172-31-54-79 raid]$ head output1TB.txt
"01uve 000000000000000000000000000000001228D4 777788880000222444DDDDDDDEEE00000000CCC7777DDDD
,K4a-.v: 000000000000000000000000000000001B8132 5555EEE88889999444FFFF1111CCCCEEE1111EEEE6666FFFF
.FuD\ju 00000000000000000000000000000000797631 5555DDDBBBB0000777722211122224444DDDDDD99996666
;5YThct 00000000000000000000000000000000703F5 2222AAAACCCCFFFFAAA44455555EEE44442222DDDD99992222
=2G9{-: 00000000000000000000000000000000809EE 5555DDD1111CCCC9999BBB8000BBBBCCCCFFFC44443333
NJM9?sp 00000000000000000000000000000000429597 5555FFF000077755559999111CCCC66669999AAAAEEE8888
POX7RsP 0000000000000000000000000000000041162E 88883339999FF1111CCCC8888CCC9999EEEEDDD00003333
[Xq]\$% 0000000000000000000000000000000097A5F0 66666666EEEEDDD7777FFF00005555FFFFFF888855551111
`rAmQg4v 000000000000000000000000000000008180D3 BBBB111111119999FFFFFFFFF4444BBBBB88884444CCCC
!&))f3; 000000000000000000000000000000004602E1 4444FFFFCCCC88888888CCCCFFCCCEEE5555666666666666
[ec2-user@ip-172-31-54-79 raid]$ [ec2-user@ip-172-31-54-79 raid]$ [ec2-user@ip-172-31-54-79 raid]$ [ec2-user@ip-172-31-54-79 raid]$ tail output1TB.txt
~~~NB_0t 000000000000000000000000000000003DE5C FFFF7777EEEBBBB4444EEEEEE33399999DD99990005555
~~~-c-CQ(> 00000000000000000000000000000000832611 9999FFF11117777333377700011114444444000BBBBB6666
~~~-c-YFql* 00000000000000000000000000000000742A4C BBBBB1111CCCCEEE888800000007773333333DD22225555
~~~-Dd@D 00000000000000000000000000000000674C9 99999DD000055556666CCC22220000FFEEEEDDDDDFF0000
~~~-JA()j$ 00000000000000000000000000000000608CBE 1111222233344455559999AAAABBBB9999FFFDDBBBB3333
~~~-Zp.#/+ 000000000000000000000000000000003B9A5A CCC8888EEEEAAAEEE333333337770000FFFFCCCC66667777
~~~-JQepix 0000000000000000000000000000000011E5D4 1111999911115555BBB1110000222EEE6666BBB7777DDDD
~~~-nt=ZH[N 00000000000000000000000000000000323A12 4444111BBB88883333777FF444455555553330000CCCC
~~~-s/Pq,-E 000000000000000000000000000000006B6E930 2222DDDDDD77771111EEECCCC7777BBB4444888811111111
~~~-zb_A_Tt 000000000000000000000000000000007F94F BBBBCCCC666655559999FF8888AAA11116666AAABBBB0000
[ec2-user@ip-172-31-54-79 raid]$ [ec2-user@ip-172-31-54-79 raid]$ [ec2-user@ip-172-31-54-79 raid]$ [ec2-user@ip-172-31-54-79 raid]$ [ec2-user@ip-172-31-54-79 raid]$
```

Total time taken = 54720 secs

HADOOP SORT APPLICATION

2.1. Introduction:

Problem Statement:

The goal of this experiment is to sort a large dataset which contains one record per line of 100 bytes and we need to sort them on the basis of ASCII values of first 10 bytes. If we use the traditional methodology of sorting then it would not only require higher space, but will also take higher time.

What is master node? What is slave node?

Hadoop consists of master and slaves in which master is responsible for storing large dataset and doing parallel computations across nodes (i.e. slaves). It has the Namenode, JobTracker, TaskTracker, whereas the slave node acts as the Datanode.

Slave nodes are the ones that actually perform the computations in parallel and are responsible for executing the actual logic of the problem statement.

Configuration files:

1. conf/masters: In this configuration file, we mention the ip address of the master node. This also optionally consists of the hostnames of the secondary node servers in case the master (namenode) fails.
2. conf/slaves: In this configuration file of masters, we mention the ip addresses of all the slave nodes present in the cluster. And in case of slaves, it would consist of localhost.
3. conf/core-site.xml: This file consists of the metadata which specifies the hadoop where the namenode (master) is located, thus we mention the ip address of the master node. We also mention the port number for the master and mostly used is 8020.
4. conf/hdfs-site.xml: This file specifies the configuration for hadoop which consists of the master, secondary name node and data nodes. We also mention the number of replications to keep for the data to be retrieved in case of failure.
5. conf/map-red.xml: This file mentions the location of job tracker. The property mapred.job.tracker takes the value of the master ip address and port number for the communication purpose.

Setting unique ports:

The reason we set unique ports is that we do not want it to match with the well known ports, else there would be problem when using the common processes like ssh etc. Also, if each user has same port number then they won't be able to use the hadoop deamons since it would then clash with the 2 users.

Changing mapper and reducers:

In order to change the number of mapper and reducers, we need to edit the configuration file mapred-site.xml in which there is a parameter *mapreduce.job.maps* and *mapreduce.job.maps*

Methodology:

- One of the ways of solving above problem efficiently is to use Hadoop framework which is used for distributed storage and distributed processing.
- Hadoop used the MapReduce Programming model.
- The storage system of Hadoop is called the HDFS (Hadoop Distributed File System) and the actual processing is the Map-reduce which has 2 phases Map and reduce.
- Hadoop works by distributing a large dataset into number of chunks and then distributing those on various nodes in the cluster which then runs in parallel.

Runtime environment settings:

Type of EC2 instance used	c3.large
Region	N. Virginia
OS and kernel	Linux version 4.4.5-15.26.amzn1.x86_64 (mockbuild@gobi-build-60007) (gcc version 4.8.3 20140911 (Red Hat 4.8.3-9) (GCC))
Implementation Language	JAVA
JAVA version	java version "1.8.0_73" Java(TM) SE Runtime Environment (build 1.8.0_73-b02) Java HotSpot(TM) 64-Bit Server VM (build 25.73-b02, mixed mode)
Hadoop Version	Hadoop 2.7.2

Installation steps to set up virtual cluster:

1. Request a spot instance of c3.large on aws. Transfer the script file to the newly created instance.
2. Run the script file which would then install java, hadoop on the master and update the configuration files as well.
3. Create an image of this instance and then add EBS volume of 400 GB.
4. Run the EBS script to mount the 400 GB volume on a folder /mnt/raid.
5. Using the AMI created in step 3, launch 16 instances (spot requests) along with 4000GB EBS volume and note down their ip addresses.
6. Edit the slaves files in configuration files of hadoop and add the ip addresses of slaves.
7. The EBS volumes of slaves are mounted using the pssh (parallel ssh command) which would then mount on /mnt/raid folder of all slaves.
8. We create a tmp directories on all the instances which would be used to store the intermediate data generated during running the hadoop framework.
9. The hdfs namenode has to be formatted before we begin the hdfs since it would then remove all the intermediate data.
10. Next is to run the hdfs using the script file “start-dfs.sh”, for the newer versions we also need to run the start-yarn.sh file
11. Verify if the cluster is created using the *jps* command which shows the services running on master and slave.

Experiment 1: Hadoop d2.xlarge single node 1GB

In this experiment I used d2.xlarge instance to sort 1GB file on single node (one master, one slave). The results are as shown in screenshots:

The screenshot shows the AWS Management Console EC2 Instances page. The left sidebar navigation bar includes links for EC2 Dashboard, Events, Tags, Reports, Limits, Instances (selected), Spot Requests, Reserved Instances, Scheduled Instances, Commands, Dedicated Hosts, Images (AMIs), and Elastic Block Store (Volumes, Snapshots). The main content area displays a table of 8 instances. The 'Master' instance is highlighted with a blue selection box. The table columns include Name, Instance ID, Instance Type, Availability Zone, Instance State, Status Checks, Alarm Status, and Public DNS. The Master instance has the following details:

Description	Value
Instance ID	i-61369ffa
Instance state	running
Instance type	d2.xlarge
Private DNS	ip-172-31-20-29.ec2.internal
Private IPs	172.31.20.29
Public DNS	ec2-52-207-223-244.compute-1.amazonaws.com
Public IP	52.207.223.244
Elastic IP	-
Availability zone	us-east-1b
Security groups	launch-wizard-1, view rules

```
Terminal
ubuntu@ip-172-31-20-29: /mnt/raid
System information as of Sun Mar 20 06:45:17 UTC 2016

System load:  0.0          Processes:           153
Usage of /:   23.5% of 7.74GB  Users logged in:    1
Memory usage: 1%
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 06:32:42 2016 from c-73-208-126-251.hsd1.il.comcast.net
ubuntu@ip-172-31-20-29:~$ cd /mnt
ubuntu@ip-172-31-20-29:/mnt$ cd /raid
-bash: cd: /raid: No such file or directory
ubuntu@ip-172-31-20-29:/mnt$ cd raid
ubuntu@ip-172-31-20-29:/mnt/raid$ ls
gensort  HadoopAssignment.jar  lost+found  tmp  valsort
ubuntu@ip-172-31-20-29:/mnt/raid$ chmod 777 gensort
ubuntu@ip-172-31-20-29:/mnt/raid$ ./gensort -a 10000000 input1GB.txt
ubuntu@ip-172-31-20-29:/mnt/raid$ █
```

Illustration A: gensort

The screenshot shows a Linux desktop environment with a terminal window open. The terminal window title is "ubuntu@ip-172-31-20-29:~". The terminal output shows the execution of a Hadoop job:

```
ubuntu@ip-172-31-20-29:~$ date
Sun Mar 20 06:58:34 UTC 2016
ubuntu@ip-172-31-20-29:~$ hadoop jar /mnt/raid/HadoopAssignment.jar HadoopAssignment /input /output
Picked up _JAVA_OPTIONS: -Xmx28g
16/03/20 06:58:43 INFO Configuration.deprecation: session.id is deprecated. Instead, use dfs.metrics.session-id
16/03/20 06:58:43 INFO jvm.JvmMetrics: Initializing JVM Metrics with processName=JobTracker, sessionId=
16/03/20 06:58:44 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
16/03/20 06:58:44 INFO input.FileInputFormat: Total input paths to process : 1
16/03/20 06:58:44 INFO mapreduce.JobSubmitter: number of splits:8
16/03/20 06:58:44 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_local99098305_0001
16/03/20 06:58:44 INFO mapreduce.Job: The url to track the job: http://localhost:8080/
16/03/20 06:58:44 INFO mapreduce.Job: Running job: job_local99098305_0001
16/03/20 06:58:44 INFO mapred.LocalJobRunner: OutputCommitter set in config null
16/03/20 06:58:44 INFO output.FileOutputCommitter: File Output Committer Algorithm version is 1
16/03/20 06:58:44 INFO mapred.LocalJobRunner: OutputCommitter is org.apache.hadoop.mapreduce.lib.output.FileOutputCommitter
16/03/20 06:58:44 INFO mapred.LocalJobRunner: Waiting for map tasks
16/03/20 06:58:44 INFO mapred.LocalJobRunner: Starting task: attempt_local99098305_0001_m_000000_0
```

Illustration B: running jar using hadoop command

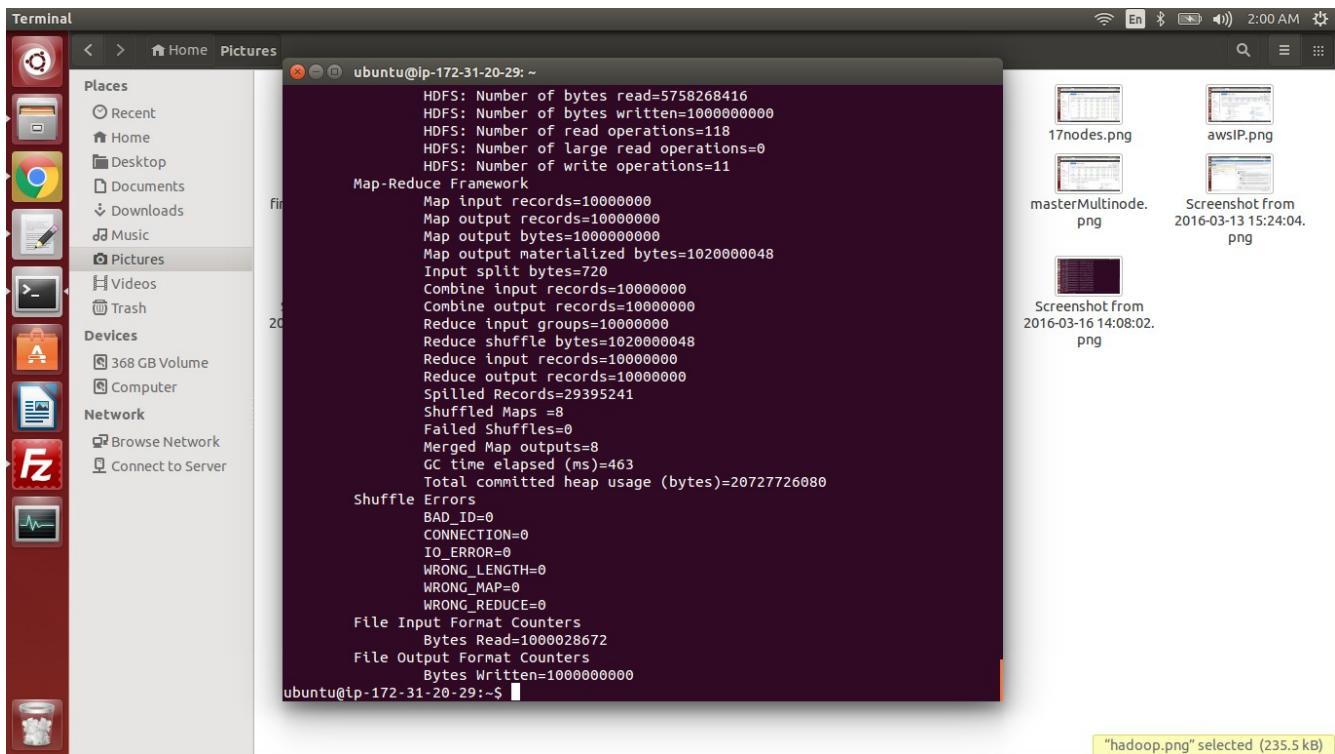


Illustration C: hadoop 1GB output

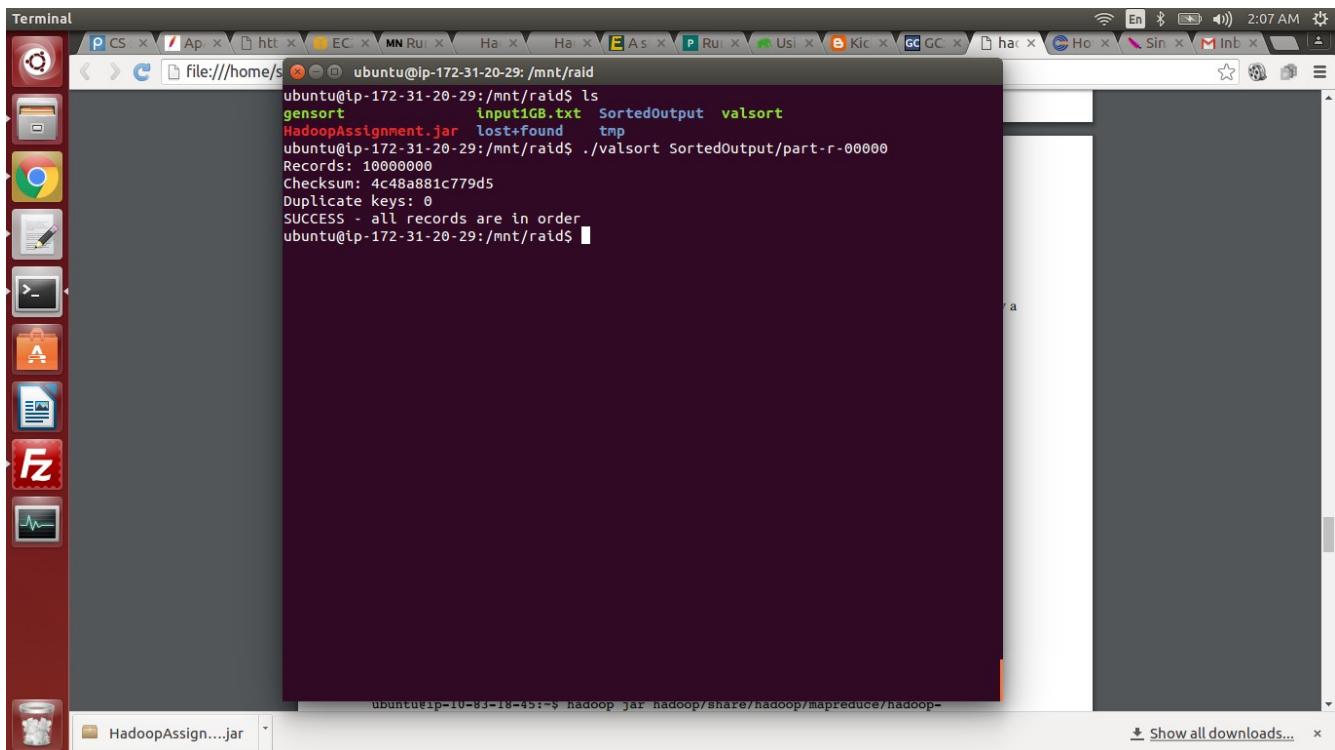


Illustration D: hadoop 1GB valsort

Observations:

Total time taken = 128 secs

Experiment 2: Hadoop c3.large single node 10GB

C3 large single node

The screenshot shows the AWS EC2 Management Console in Google Chrome. The left sidebar has icons for EC2 Dashboard, Events, Tags, Reports, Limits, Instances (selected), Spot Requests, Reserved Instances, Scheduled Instances, Commands, Dedicated Hosts, Images (AMIs), Bundle Tasks, Elastic Block Store (Volumes, Snapshots), Network & Security (Security Groups, Elastic IPs, Placement Groups), and CloudWatch Metrics. The main content area shows a table of instances. One row is selected for an instance named 'Master' with Instance ID i-b0122734, running in us-east-1a. The detailed view on the right shows the following information:

Attribute	Value
Instance ID	i-b0122734
Public DNS	ec2-54-164-43-189.compute-1.amazonaws.com
Public IP	54.164.43.189
Elastic IP	-
Availability zone	us-east-1a
Security groups	launch-wizard-1 . view rules
Scheduled events	No scheduled events
AMI ID	ubuntu-trusty-14.04-amd64-server-20160114.5 (ami-fce3c696)
Platform	-
IAM role	-
Keypair name	hadoop2

The screenshot shows the WinSCP interface. On the left, there's a file browser window showing local files like 'hs_err_pid18260.log', 'valsort', 'gensort', 'input.txt', 'output1TB.txt', 'FinalOutput.txt', 'input1GB.txt', 'out.txt', 'output.txt', and 'output1GB.txt'. The right side shows a terminal window with the following command history and output:

```

Command: put "/home/sheetal/Desktop/Cloud/Assignment2/SharedMemory/aws/gensort" "gensort"
Status: local:/home/sheetal/Desktop/Cloud/Assignment2/SharedMemory/aws/gensort => remote:/mnt/raid/gensort
Status: File transfer successful, transferred 65.6 KB in 1 second
Status: Retrieving directory listing...
Command: ls
Status: Listing directory /mnt/raid
Status: Directory listing successful
Status: Disconnected from server
Local site: /home/sheetal/Desktop/Cloud/Assignment2/SharedMemory/aws
Filesize  Filetype  Last modified
Selected 1 file. Total size: 141.1 KB
Selected 1 file. Total size: 4.1 KB

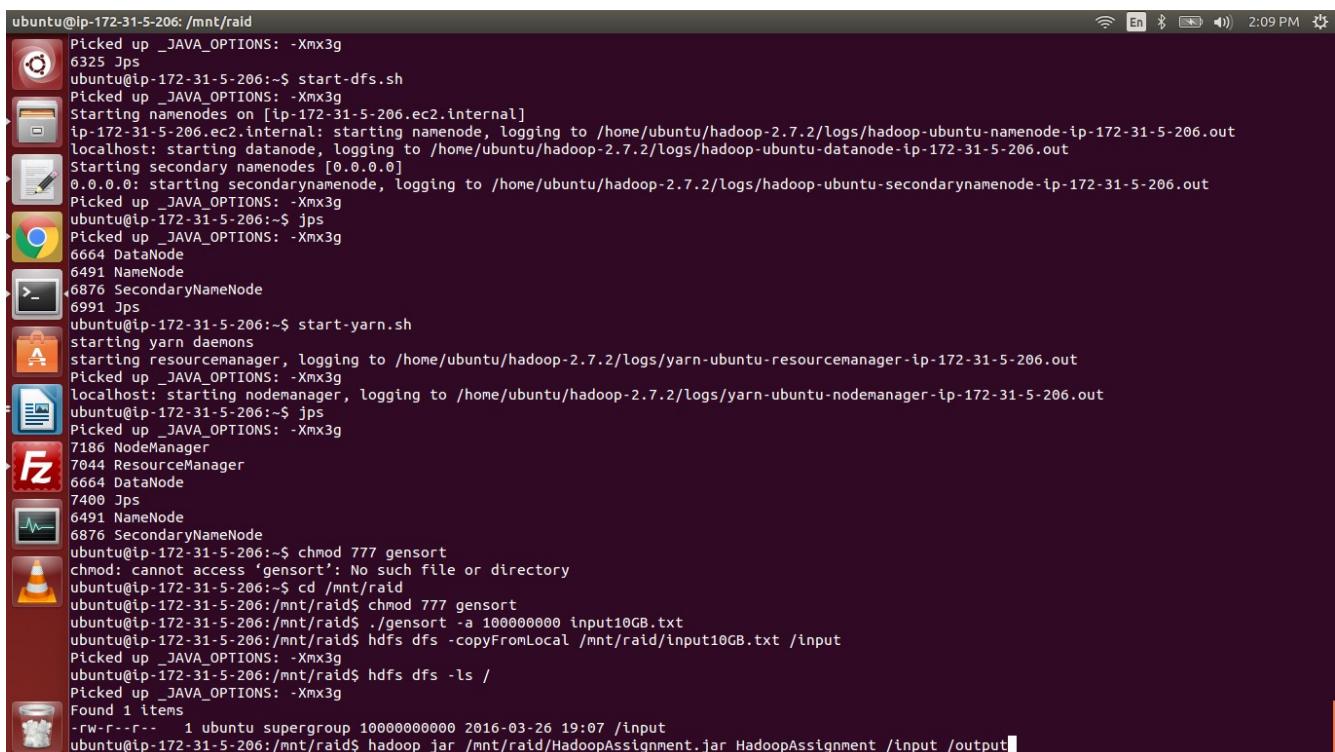
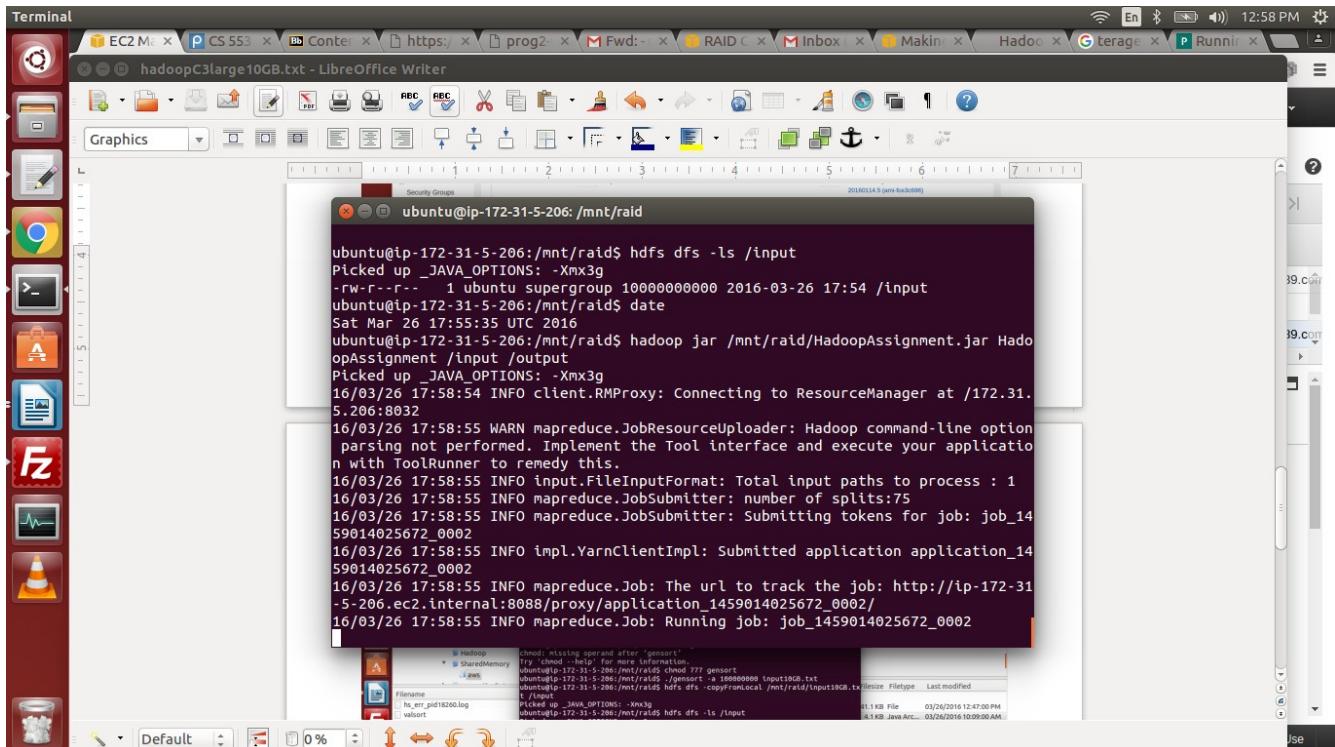
```

The terminal also displays Java options and HDFS commands:

```

ubuntu@ip-172-31-5-206:~/mnt/raid$ cd /mnt/raid
ubuntu@ip-172-31-5-206:~/mnt/raid$ chmod gensort
chmod: missing operand after 'gensort'
Try 'chmod --help' for more information.
ubuntu@ip-172-31-5-206:~/mnt/raid$ chmod 777 gensort
ubuntu@ip-172-31-5-206:~/mnt/raid$ ./gensort -a 1000000000 input10GB.txt
ubuntu@ip-172-31-5-206:~/mnt/raid$ hdfs dfs -copyFromLocal /mnt/raid/input10GB.txt /input
Picked up _JAVA_OPTIONS: -Xmx3g
ubuntu@ip-172-31-5-206:~/mnt/raid$ hdfs dfs -ls /input
Picked up _JAVA_OPTIONS: -Xmx3g
ubuntu@ip-172-31-5-206:~/mnt/raid$ hdfs dfs -ls /input
Picked up _JAVA_OPTIONS: -Xmx3g
-rw-r--r-- 1 ubuntu supergroup 10000000000 2016-03-26 17:54 /input
ubuntu@ip-172-31-5-206:~/mnt/raid$ clear
ubuntu@ip-172-31-5-206:~/mnt/raid$ hdfs dfs -ls /input
Picked up _JAVA_OPTIONS: -Xmx3g
ubuntu@ip-172-31-5-206:~/mnt/raid$ hdfs dfs -ls /input
-rw-r--r-- 1 ubuntu supergroup 10000000000 2016-03-26 17:54 /input
ubuntu@ip-172-31-5-206:~/mnt/raid$ date
Sat Mar 26 17:55:35 UTC 2016
ubuntu@ip-172-31-5-206:~/mnt/raid$ hadoop jar /mnt/raid/HadoopAssignment.jar HadoopAssignment /input /output

```



running sorting application - time to be noted 19:10:21

```
ubuntu@ip-172-31-5-206:/mnt/raid$ hdfs dfs -ls /
Picked up _JAVA_OPTIONS: -Xmx3g
Found 1 items
-rw-r--r-- 1 ubuntu supergroup 10000000000 2016-03-26 19:07 /input
ubuntu@ip-172-31-5-206:/mnt/raid$ date
Sat Mar 26 19:10:14 UTC 2016
ubuntu@ip-172-31-5-206:/mnt/raid$ 
ubuntu@ip-172-31-5-206:/mnt/raid$ 
ubuntu@ip-172-31-5-206:/mnt/raid$ 
ubuntu@ip-172-31-5-206:/mnt/raid$ 
ubuntu@ip-172-31-5-206:/mnt/raid$ hadoop jar /mnt/raid/HadoopAssignment.jar HadoopAssignment /input /output
Picked up _JAVA_OPTIONS: -Xmx3g
16/03/26 19:10:21 INFO client.RMProxy: Connecting to ResourceManager at /172.31.5.206:8032
16/03/26 19:10:22 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
16/03/26 19:10:22 INFO input.FileInputFormat: Total input paths to process : 1
16/03/26 19:10:23 INFO mapreduce.JobSubmitter: number of splits:75
16/03/26 19:10:23 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1459018744034_0001
16/03/26 19:10:23 INFO mapreduce.Job: The url to track the job: http://ip-172-31-5-206.ec2.internal:8088/proxy/application_1459018744034_0001/
16/03/26 19:10:23 INFO mapreduce.Job: Running job: job_1459018744034_0001
16/03/26 19:10:33 INFO mapreduce.Job: Job job_1459018744034_0001 running in uber mode : false
16/03/26 19:10:33 INFO mapreduce.Job: map 0% reduce 0%
16/03/26 19:11:01 INFO mapreduce.Job: map 1% reduce 0%
16/03/26 19:11:04 INFO mapreduce.Job: map 2% reduce 0%
16/03/26 19:11:07 INFO mapreduce.Job: map 3% reduce 0%
16/03/26 19:11:12 INFO mapreduce.Job: map 4% reduce 0%
16/03/26 19:11:26 INFO mapreduce.Job: map 5% reduce 0%
16/03/26 19:11:38 INFO mapreduce.Job: map 6% reduce 0%
16/03/26 19:11:41 INFO mapreduce.Job: map 7% reduce 0%
16/03/26 19:11:45 INFO mapreduce.Job: map 8% reduce 0%
16/03/26 19:12:14 INFO mapreduce.Job: map 9% reduce 0%
16/03/26 19:12:17 INFO mapreduce.Job: map 10% reduce 0%
16/03/26 19:12:20 INFO mapreduce.Job: map 11% reduce 0%
16/03/26 19:12:24 INFO mapreduce.Job: map 12% reduce 0%
16/03/26 19:12:39 INFO mapreduce.Job: map 13% reduce 0%
16/03/26 19:12:51 INFO mapreduce.Job: map 14% reduce 0%
16/03/26 19:12:55 INFO Mapreduce.Job: map 15% reduce 0%
16/03/26 19:12:58 INFO mapreduce.Job: map 16% reduce 0%
```

```
ubuntu@ip-172-31-5-206:/mnt/raid
Total time spent by all maps in occupied slots (ms)=4943220
Total time spent by all reduces in occupied slots (ms)=957836
Total time spent by all map tasks (ms)=4943220
Total time spent by all reduce tasks (ms)=957836
Total vcore-milliseconds taken by all map tasks=4943220
Total vcore-milliseconds taken by all reduce tasks=957836
Total megabyte-milliseconds taken by all map tasks=5061857280
Total megabyte-milliseconds taken by all reduce tasks=980824064
Map-Reduce Framework
  Map input records=100000000
  Map output records=100000000
  Map output bytes=10000000000
  Map output materialized bytes=10200000450
  Input split bytes=7125
  Combine input records=100000000
  Combine output records=100000000
  Reduce input groups=100000000
  Reduce shuffle bytes=10200000450
  Reduce input records=100000000
  Reduce output records=100000000
  Spilled Records=396636763
  Shuffled Maps =75
  Failed Shuffles=0
  Merged Map outputs=75
  GC time elapsed (ms)=87210
  CPU time spent (ms)=1474730
  Physical memory (bytes) snapshot=19277725696
  Virtual memory (bytes) snapshot=144797569024
  Total committed heap usage (bytes)=15632171008
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=10000303104
File Output Format Counters
  Bytes Written=10000000000
ubuntu@ip-172-31-5-206:/mnt/raid$
```

```
ubuntu@ip-172-31-5-206:/mnt/raid$ Input split bytes=7125
Combine input records=100000000
Combine output records=100000000
Reduce input groups=100000000
Reduce shuffle bytes=102000000450
Reduce input records=100000000
Reduce output records=100000000
Spilled Records=396636763
Shuffled Maps =75
Failed Shuffles=0
Merged Map outputs=75
GC time elapsed (ms)=87210
CPU time spent (ms)=1474730
Physical memory (bytes) snapshot=19277725696
Virtual memory (bytes) snapshot=144797569024
Total committed heap usage (bytes)=15632171008
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=10000303104
File Output Format Counters
Bytes Written=10000000000
ubuntu@ip-172-31-5-206:/mnt/raid$ hdfs dfs -ls /
Picked up _JAVA_OPTIONS: -Xmx3g
Found 3 items
-rw-r--r-- 1 ubuntu supergroup 10000000000 2016-03-26 19:07 /input
drwxr-xr-x - ubuntu supergroup 0 2016-03-26 19:30 /output
drwx----- - ubuntu supergroup 0 2016-03-26 19:10 /tmp
ubuntu@ip-172-31-5-206:/mnt/raid$ hdfs dfs -ls /output
Picked up _JAVA_OPTIONS: -Xmx3g
Found 2 items
-rw-r--r-- 1 ubuntu supergroup 0 2016-03-26 19:30 /output/_SUCCESS
-rw-r--r-- 1 ubuntu supergroup 10000000000 2016-03-26 19:30 /output/part-r-00000
ubuntu@ip-172-31-5-206:/mnt/raid$ date
Sat Mar 26 19:32:28 UTC 2016
ubuntu@ip-172-31-5-206:/mnt/raid$
```

time to be noted above in creation of output file – 19:30

```
ubuntu@ip-172-31-5-206:/mnt/raid$ File Input Format Counters
Bytes Read=10000303104
File Output Format Counters
Bytes Written=10000000000
ubuntu@ip-172-31-5-206:/mnt/raid$ hdfs dfs -ls /
Picked up _JAVA_OPTIONS: -Xmx3g
Found 3 items
-rw-r--r-- 1 ubuntu supergroup 10000000000 2016-03-26 19:07 /input
drwxr-xr-x - ubuntu supergroup 0 2016-03-26 19:30 /output
drwx----- - ubuntu supergroup 0 2016-03-26 19:10 /tmp
ubuntu@ip-172-31-5-206:/mnt/raid$ hdfs dfs -ls /output
Picked up _JAVA_OPTIONS: -Xmx3g
Found 2 items
-rw-r--r-- 1 ubuntu supergroup 0 2016-03-26 19:30 /output/_SUCCESS
-rw-r--r-- 1 ubuntu supergroup 10000000000 2016-03-26 19:30 /output/part-r-00000
ubuntu@ip-172-31-5-206:/mnt/raid$ date
Sat Mar 26 19:32:28 UTC 2016
ubuntu@ip-172-31-5-206:/mnt/raid$ hdfs dfs -copyToLocal /output /mnt/raid
Picked up _JAVA_OPTIONS: -Xmx3g
-copyToLocal: Unknown command
ubuntu@ip-172-31-5-206:/mnt/raid$ hdfs dfs -copyToLocal /output /mnt/raid
Picked up _JAVA_OPTIONS: -Xmx3g
ubuntu@ip-172-31-5-206:/mnt/raid$ ls
gensort HadoopAssignment.jar input10GB.txt lost+found output tmp
ubuntu@ip-172-31-5-206:/mnt/raid$ cd /mnt/raid/output
ubuntu@ip-172-31-5-206:/mnt/raid/output$ ls
part-r-00000 _SUCCESS
ubuntu@ip-172-31-5-206:/mnt/raid/output$ cd ..
ubuntu@ip-172-31-5-206:/mnt/raid$ ls
gensort HadoopAssignment.jar input10GB.txt lost+found output tmp valsrt
ubuntu@ip-172-31-5-206:/mnt/raid$ chmod 777 valsrt
ubuntu@ip-172-31-5-206:/mnt/raid$ ./valsrt output/part-r-00000

Records: 100000000
Checksum: 2fb0574596d67c8
Duplicate keys: 0
SUCCESS - all records are in order
ubuntu@ip-172-31-5-206:/mnt/raid$
```

```

ubuntu@ip-172-31-5-206:/mnt/raid$ gensort HadoopAssignment.jar input10GB.txt lost+found output tmp
ubuntu@ip-172-31-5-206:/mnt/raid$ cd /mnt/raid/output
ubuntu@ip-172-31-5-206:/mnt/raid/output$ ls
part-r-00000 SUCCESS
ubuntu@ip-172-31-5-206:/mnt/raid/output$ cd ..
ubuntu@ip-172-31-5-206:/mnt/raid$ ls
gensort HadoopAssignment.jar input10GB.txt lost+found output tmp valsrt
ubuntu@ip-172-31-5-206:/mnt/raids chmod 777 valsrt
ubuntu@ip-172-31-5-206:/mnt/raid$ ./valsrt output/part-r-00000

Records: 100000000
Checksum: 2fb0574596d67c8
Duplicate keys: 0
SUCCESS - all records are in order
ubuntu@ip-172-31-5-206:/mnt/raids
ubuntu@ip-172-31-5-206:/mnt/raids
ubuntu@ip-172-31-5-206:/mnt/raids head output/part-r-00000
    "0!uve      000000000000000000000000000000001228D4 77778888000022224444DDDDDDDEEE00000000CCCC7777DDDD
    PMd32=    000000000000000000000000000000003440CC1 FFFFEEEE6666CCCCBBBB99993335555DDDDDD777788886666
    ^3C0],   00000000000000000000000000000000158C5C5 5555AAA9999EEE888822229999CCCCDD666655554442222
    !&S3/] ]  000000000000000000000000000000002145D78 8888BBBBDD111CCCC5555666888B111EEEEDDD22229999
    !=U#_9    0000000000000000000000000000000019072E3 33332222FFFFBBBB0000FFFFAA66665553333DDDD3333CCCC
    !f6[ITd   000000000000000000000000000000003CAA848 9999FFFFF5553333777CCCC4444BBBB7777EEE888888888888
    !f6$uy2   000000000000000000000000000000003ABFD84 EEE55555556666AAA5555BBBBDD00001111666600000DDDD
    #%%NIpq.  000000000000000000000000000000003B36FB9 11100000333444411166666666AAAAAAA00001111CCCCEEE
    #'^cl`~   000000000000000000000000000000002EDC5C8 8888AAA11114444FFFF7777333EEE44440000FFF99999999
    $`-'Q)]  000000000000000000000000000000005F1265D CCCC6666EEEE22220000DDDAAA88886666BBBB0006666AAA
ubuntu@ip-172-31-5-206:/mnt/raids
ubuntu@ip-172-31-5-206:/mnt/raids tail output/part-r-00000
~~~uq2k#=U  000000000000000000000000000000002C06745 999911110DD022221110000FFFFEEEEEFFFF33337777CCCC2222
~~~v/0&0nm  000000000000000000000000000000004709701 CCCCB8883333FFF000000000009999111FFF77774446666
~~~yKol;gE  00000000000000000000000000000000204884F CCC11114444888822226666BBBB88885557777EEE8888888888
~~~yKH.il  00000000000000000000000000000000463D004 4440000FFF3333999944447777DDDFFFFFAAA111188880DD
~~~yl;C'XE  00000000000000000000000000000000580D211 2222EEE33330002221111CCCCFF555577774444BBBB6666
~~~zBa_Tt   000000000000000000000000000000007F9F4F BBBBCCCC666655559999FFFF8888AAA11116666AAAABB80000
~~~ze0^FEG  000000000000000000000000000000001E06130 4444CCCCB888999922288885558888CCCCFF000011111111
~~~]GxjWHI  00000000000000000000000000000000CA1345 777711118888AAAAAA2222111BBBB0002222BBBBCCCC2222
~~~]P;jg0g   0000000000000000000000000000000040DA3E4 4444FFFF444466663333EEE88888888DDDEEEE44442222DDDD
~~~]Ku|k-p   000000000000000000000000000000005E4AOAA 00066665555111BBBB88889999AAA55550000333355557777
ubuntu@ip-172-31-5-206:/mnt/raids"

```

Total time taken = 1179 secs

Hadoop Multinode

Experiment: 100 GB dataset on cluster of 16 nodes of c3.large instances

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS	Public IP
slave1	i-f2b68476	c3.large	us-east-1a	running	2/2 checks ...	None	ec2-54-210-176-242.co...	54.210.176.242
slave2	i-e2b78566	c3.large	us-east-1a	running	2/2 checks ...	None	ec2-54-210-113-128.co...	54.210.113.128
slave3	i-f1b68475	c3.large	us-east-1a	running	2/2 checks ...	None	ec2-54-175-71-22.comp...	54.175.71.22
slave4	i-3b68477	c3.large	us-east-1a	running	2/2 checks ...	None	ec2-54-210-85-16.comp...	54.210.85.16
slave5	i-84b78500	c3.large	us-east-1a	running	2/2 checks ...	None	ec2-54-175-242-138.co...	54.175.242.138
slave6	i-44b78500	c3.large	us-east-1a	running	2/2 checks ...	None	ec2-54-210-156-133.co...	54.210.156.133
slave7	i-fcb78578	c3.large	us-east-1a	running	2/2 checks ...	None	ec2-54-210-176-160.co...	54.210.176.160
slave8	i-8ab7850e	c3.large	us-east-1a	running	2/2 checks ...	None	ec2-54-210-134-79.com...	54.210.134.79
slave9	i-46b785c2	c3.large	us-east-1a	running	2/2 checks ...	None	ec2-54-89-112-180.com...	54.89.112.180
slave10	i-85b78501	c3.large	us-east-1a	running	2/2 checks ...	None	ec2-54-210-110-159.co...	54.210.110.159
slave11	i-45b785c1	c3.large	us-east-1a	running	2/2 checks ...	None	ec2-54-210-156-108.co...	54.210.156.108
slave12	i-87b78503	c3.large	us-east-1a	running	2/2 checks ...	None	ec2-54-210-117-156.co...	54.210.117.156
slave13	i-47b785c3	c3.large	us-east-1a	running	2/2 checks ...	None	ec2-54-209-175-117.co...	54.209.175.117
slave14	i-2cb684a8	c3.large	us-east-1a	running	2/2 checks ...	None	ec2-54-210-86-229.com...	54.210.86.229
slave15	i-8bb7850f	c3.large	us-east-1a	running	2/2 checks ...	None	ec2-54-88-1-243.compu...	54.88.1.243
slave16	i-e3b78567	c3.large	us-east-1a	running	2/2 checks ...	None	ec2-54-209-53-35.comp...	54.209.53.35
Master	i-9bd1e31f	c3.large	us-east-1a	running	2/2 checks ...	None	ec2-54-173-189-205.co...	54.173.189.205

Description	Value
Instance ID	i-9bd1e31f
Instance state	running
Instance type	c3.large
Private DNS	ip-172-31-1-121.ec2.internal
Private IPs	172.31.1.121
Secondary private IPs	
VPC ID	vpc-b65f63d2
Subnet ID	subnet-1674ee60
Public DNS	ec2-54-173-189-205.compute-1.amazonaws.com
Public IP	54.173.189.205
Elastic IP	-
Availability zone	us-east-1a
Security groups	launch-wizard-1 . view rules
Scheduled events	No scheduled events
AMI ID	amzn-ami-hvm-2016.03.0.x86_64-gp2 (ami-08111162)
Platform	-

```
ec2-user@ip-172-31-1-121:~$ sheetal@sheetal-VPCEH15FD:~$ cd /home/sheetal/Desktop/Cloud/Assignment2/SharedMemory/aws
sheetal@sheetal-VPCEH15FD:~/Desktop/Cloud/Assignment2/SharedMemory/aws$ ssh -i h
adoop2.pem ec2-user@ec2-54-173-189-205.compute-1.amazonaws.com
Last login: Tue Mar 29 01:00:26 2016 from 208-59-149-165.c3-0.mcm-ubr1.chi-mcm.i
l.cable.rcn.com
[ec2-user@ip-172-31-1-121 ~]$ clear
[ec2-user@ip-172-31-1-121 ~]$ hdfs dfs -ls /
Picked up _JAVA_OPTIONS: -Xmx2g
Found 3 items
-rw-r--r-- 1 ec2-user supergroup 1000000000 2016-03-29 00:27 /input
-rw-r--r-- 1 ec2-user supergroup 10000000000 2016-03-28 19:06 /input1
drwxr-xr-x  - ec2-user supergroup          0 2016-03-29 00:54 /output
[ec2-user@ip-172-31-1-121 ~]$ hadoop jar /mnt/raid/HadoopAssignment.jar HadoopAssignment /input1 /output^C
[ec2-user@ip-172-31-1-121 ~]$ date
Tue Mar 29 01:08:21 UTC 2016
[ec2-user@ip-172-31-1-121 ~]$ hdfs dfs -rm -r /output
Picked up _JAVA_OPTIONS: -Xmx2g
16/03/29 01:09:12 INFO fs.TrashPolicyDefault: Namenode trash configuration: Deletion interval = 0 minutes, Emptier interval = 0 minutes.
Deleted /output
[ec2-user@ip-172-31-1-121 ~]$ hdfs dfs -ls /
Picked up _JAVA_OPTIONS: -Xmx2g
Found 2 items
-rw-r--r-- 1 ec2-user supergroup 1000000000 2016-03-29 00:27 /input
-rw-r--r-- 1 ec2-user supergroup 10000000000 2016-03-28 19:06 /input1
[ec2-user@ip-172-31-1-121 ~]$ hadoop jar /mnt/raid/HadoopAssignment.jar HadoopAssignment /input1 /output
```

```
ec2-user@ip-172-31-1-121:~$ Making 1 from 100000 sampled records
Computing partitions took 764ms
Spent 1152ms computing partitions.
16/03/29 02:33:48 INFO client.RMProxy: Connecting to ResourceManager at /172.31.1.121:8032
16/03/29 02:33:49 INFO mapreduce.JobSubmitter: number of splits:745
16/03/29 02:33:50 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1459218786274_0001
16/03/29 02:33:50 INFO impl.YarnClientImpl: Submitted application application_1459218786274_0001
16/03/29 02:33:50 INFO mapreduce.Job: The url to track the job: http://ip-172-31-1-121.ec2.internal:8088/proxy/application_1459218786274_0001/
16/03/29 02:34:01 INFO mapreduce.Job: Job job_1459218786274_0001 running in uber mode : false
16/03/29 02:34:01 INFO mapreduce.Job: map 0% reduce 0%
16/03/29 02:34:20 INFO mapreduce.Job: map 2% reduce 0%
16/03/29 02:34:21 INFO mapreduce.Job: map 4% reduce 0%
16/03/29 02:34:26 INFO mapreduce.Job: map 5% reduce 0%
16/03/29 02:34:27 INFO mapreduce.Job: map 6% reduce 0%
16/03/29 02:34:35 INFO mapreduce.Job: map 7% reduce 0%
16/03/29 02:34:36 INFO mapreduce.Job: map 8% reduce 0%
16/03/29 02:34:43 INFO mapreduce.Job: map 9% reduce 0%
16/03/29 02:34:53 INFO mapreduce.Job: map 10% reduce 0%
16/03/29 02:34:56 INFO mapreduce.Job: map 11% reduce 0%
16/03/29 02:34:57 INFO mapreduce.Job: map 12% reduce 0%
16/03/29 02:34:59 INFO mapreduce.Job: map 13% reduce 0%
16/03/29 02:35:03 INFO mapreduce.Job: map 14% reduce 0%
16/03/29 02:35:07 INFO mapreduce.Job: map 15% reduce 0%
16/03/29 02:35:10 INFO mapreduce.Job: map 16% reduce 0%
16/03/29 02:35:12 INFO mapreduce.Job: map 17% reduce 0%
16/03/29 02:35:24 INFO mapreduce.Job: map 18% reduce 0%
16/03/29 02:35:27 INFO mapreduce.Job: map 19% reduce 0%
16/03/29 02:35:29 INFO mapreduce.Job: map 20% reduce 0%
16/03/29 02:35:32 INFO mapreduce.Job: map 21% reduce 0%
16/03/29 02:35:34 INFO mapreduce.Job: map 21% reduce 1%
16/03/29 02:35:35 INFO mapreduce.Job: map 22% reduce 1%
16/03/29 02:35:38 INFO mapreduce.Job: map 23% reduce 1%
16/03/29 02:35:41 INFO mapreduce.Job: map 24% reduce 1%
16/03/29 02:35:44 INFO mapreduce.Job: map 25% reduce 1%
16/03/29 02:35:55 INFO mapreduce.Job: map 26% reduce 1%
16/03/29 02:35:58 INFO mapreduce.Job: map 27% reduce 1%
16/03/29 02:36:00 INFO mapreduce.Job: map 28% reduce 1%
16/03/29 02:36:02 INFO mapreduce.Job: map 29% reduce 1%
16/03/29 02:36:04 INFO mapreduce.Job: map 30% reduce 1%
```

```
ec2-user@ip-172-31-1-121:~$ Physical memory (bytes) snapshot=225078706176
Virtual memory (bytes) snapshot=2238865739776
Total committed heap usage (bytes)=188944482304
    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=1000000000000
    File Output Format Counters
        Bytes Written=1000000000000
16/03/29 04:22:03 INFO terasort.TeraSort: done
[ec2-user@ip-172-31-1-121 ~]$ date
Tue Mar 29 04:24:48 UTC 2016
[ec2-user@ip-172-31-1-121 ~]$ hdfs dfs -la /
Picked up _JAVA_OPTIONS: -Xmx1g
-la: Unknown command
[ec2-user@ip-172-31-1-121 ~]$ hdfs dfs -ls /
Picked up _JAVA_OPTIONS: -Xmx1g
Found 3 items
-rw-r--r-- 1 ec2-user supergroup 100000000000 2016-03-28 19:06 /input1
drwxr-xr-x - ec2-user supergroup 0 2016-03-29 04:22 /output
drwxr----- - ec2-user supergroup 0 2016-03-29 02:33 /tmp
[ec2-user@ip-172-31-1-121 ~]$ hdfs dfs -ls /output
Picked up _JAVA_OPTIONS: -Xmx1g
Found 3 items
-rw-r--r-- 1 ec2-user supergroup 0 2016-03-29 04:22 /output/_SUCCESS
-rw-r--r-- 10 ec2-user supergroup 0 2016-03-29 02:33 /output/_partition.lst
-rw-r--r-- 1 ec2-user supergroup 100000000000 2016-03-29 04:22 /output/part-r-00000
[ec2-user@ip-172-31-1-121 ~]$ hdfs dfs -copyToLocal /output /mnt/raid
Picked up _JAVA_OPTIONS: -Xmx1g
```

```
ec2-user@ip-172-31-1-121:/mnt/raid
-rwxrwxrwx 1 ec2-user ec2-user 134558 Mar 28 16:20 valsort
[ec2-user@ip-172-31-1-121 raid]$ ./valsort output/part-r-00000

Records: 1000000000
Checksum: 1dc615efb9dfe11
① Duplicate keys: 0
SUCCESS - all records are in order
[ec2-user@ip-172-31-1-121 raid]$ 
[ec2-user@ip-172-31-1-121 raid]$ 
[ec2-user@ip-172-31-1-121 raid]$ 
[ec2-user@ip-172-31-1-121 raid]$ 
[ec2-user@ip-172-31-1-121 raid]$ 
[ec2-user@ip-172-31-1-121 raid]$ 
[ec2-user@ip-172-31-1-121 raid]$ head output/part-r-00000
!+ABV 00000000000000000000000000000017F7E829 EEEEEE3333444411112222888833334444666633332222DDDEEEE
"Olive 0000000000000000000000000000000022244440DDDDDDDEEE000000000CCC7777DDDD
%!S(U 000000000000000000000000000000002E6C821C 22223333777444555511119999CCCC4444EEEFFFF11115555
&5rx/X 00000000000000000000000000000000398BC288 5555CCCCBBB99999999DD011100001111EEE7777DDDD9999
'ic%So 0000000000000000000000000000000031F06B7D EEEEEEBBBBAAA8888BDDDDDD777722224444111166664444AAAA
*#G1Io 000000000000000000000000000000003B5E85A1 1111AAA9999CCCCBBB111199991111333399991111AAAG666
,(Ght_ 000000000000000000000000000000002D0172DC 1111CCCC1111DDDCCCCEEE9999CCCC8888CCCCFFFF55555555
*#vym3 0000000000000000000000000000000026D61578 DDD7777AAAEEEEE6666AAA2222CCC5555555522229999
2C->8d 0000000000000000000000000000000026C79E66 444400001111CCC6666BBB555577776666CCC2222AAABB
PMD32= 000000000000000000000000000000003440CC1 FFFFFFFEE6666CCCCBBB9999333355550DDDDDD077788886666
[ec2-user@ip-172-31-1-121 raid]$ tail output/part-r-00000
~~~#aiay1X 00000000000000000000000000000025D35EDF 6666AAA5555999977770000222233338888FFFF999922220000
~~~+@p){ 0000000000000000000000000000000045426F4 7777333355P){111111110000000CCC5555999AAA7777DDDDDD
~~~R^?n 000000000000000000000000000000001034E347 11111111999900011118888AAA55554444EEE999933338888
~~~E`Y) 0000000000000000000000000000000016F0E66B CCC6666DDDD2222DD011188889999EEEEEFFFFEEEEE8888B4444
~~~4!kA7X 0000000000000000000000000000001A1E26 EEEEEE777711117777BBB1111EEE88884444DDDDDEEEEBBBB
~~~8!l!@! 0000000000000000000000000000001F05932F 11119999BBB44447777000011114444CCCCAA6666DDDD0000
~~~I-'5>F 00000000000000000000000000000000293 88883333BBB11F1116666999888855588888822228888CCCC
~~~G-)m^) 00000000000000000000000000000013397FT3 DDDDFFFFBBBCCCCFF44446666AAA111133333333AAACCC
~~~c+I&cP 00000000000000000000000000000074BDF64 8888000055550000DDD22227777AAA00003333222AAAADD
~~~hb&SX* 0000000000000000000000000000003C0E06B 7777BBBBBBB9999EEEAAAAAA0000CCCCDD0444BBBB4444
[ec2-user@ip-172-31-1-121 raid]$
```

As observed in above experiment screenshot number 4, the application started running at time 2:33:48 and as seen screenshot 6, the output file was created at 4:22:00, thus we can say the the time required for running th experiment was 5952 secs (1 hour 48 mins 12 secs).

Total time taken = 6492 secs

Spark

3.1. Introduction

Problem Statement:

The goal of this experiment is to sort a large dataset which contains one record per line of 100 bytes and we need to sort them on the basis of ASCII values of first 10 bytes. If we use the traditional methodology of sorting then it would not only require higher space, but will also take higher time.

Methodology:

The spark framework runs on top of hadoop framework and that it runs much faster

Environment Settings:

Type of EC2 instance used	c3.large
Region	N. Virginia
OS and kernel	Red Hat kernel-3.4.37-40.44.amzn1.x86_64
Implementation Language	Scala
JAVA version	Open JDK 1.7.0_99
Hadoop Version	Hadoop 1.0.
Spark Version	Spark 1.6.0
Scala version	Scala 2.10.3

Installation steps:

1. Create any instance on aws.
2. Download Java, Spark, Scala, Ganglia on that instance
3. Upload the pem file to instance and give permissions "chmod 400 hadoop2.pem"
4. Run export commands:

```
export AWS_ACCESS_KEY_ID=<your key ID>
export AWS_SECRET_ACCESS_KEY=<your key>
```
5. Edit security group to add rule for All TCP and ALL traffic
6. run the command to launch spark:
`./spark-ec2 -k hadoop2 -i hadoop2.pem --region=us-east-1 -s 16 --instance-type=c3.large --ebs-vol-size=100 -m c3.large --ebs-vol-size=400 --spot-price=0.023 launch spark2`
7. once cluster is created, login to spark master using command:
`./spark-ec2 -k hadoop2 -i hadoop2.pem --region=us-east-1 login spark2`
8. generate the input file using gensort and transfer it to hdfs
9. run the spark-shell command and then spark shell is opened
10. Here we need to run the scala code by typing commands one after other as given in source code
11. once the output file is generated, it can be transferred to local and verify the result using valsor

3.2 Experiment: 100 GB

EC2 Management Console - Google Chrome

<https://console.aws.amazon.com/ec2/v2/home?region=us-east-1#Instances:sort=instanceType>

AWS Services Edit More Sheetal More N. Virginia Support

EC2 Dashboard Events Tags Reports Limits

INSTANCES Instances Spot Requests Reserved Instances Scheduled Instances Commands Dedicated Hosts

IMAGES AMIs Bundle Tasks

ELASTIC BLOCK STORE Volumes Snapshots

NETWORK & SECURITY Security Groups Elastic IPs Placement Groups

Feedback English

Launch Instance Connect Actions

Filter by tags and attributes or search by keyword

1 to 18 of 18

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS
spark2-slave-i-39b...	i-39b32ea2	c3.large	us-east-1b	running	2/2 checks ...	None	ec2-54-209-248...
spark2-slave-i-13b...	i-13b22f88	c3.large	us-east-1b	running	2/2 checks ...	None	ec2-54-165-44-20
spark2-slave-i-cdad...	i-cdad3056	c3.large	us-east-1b	running	2/2 checks ...	None	ec2-54-175-2-8...
spark2-slave-i-6ccb2...	i-6ccb2ff7	c3.large	us-east-1b	running	2/2 checks ...	None	ec2-54-164-62-21
spark2-master-i-8fb...	i-8fb22f14	c3.large	us-east-1b	running	2/2 checks ...	None	ec2-52-91-139-19
spark2-slave-i-ccad...	i-ccad3057	c3.large	us-east-1b	running	2/2 checks ...	None	ec2-54-152-60-56

Instance: i-8fb22f14 (spark2-master-i-8fb22f14) Public DNS: ec2-52-91-139-197.compute-1.amazonaws.com

Description Status Checks Monitoring Tags

Instance ID	i-8fb22f14	Public DNS	ec2-52-91-139-197.compute-1.amazonaws.com
Instance state	running	Public IP	52.91.139.197
Instance type	c3.large	Elastic IP	-
Private DNS	ip-172-31-17-47.ec2.internal	Availability zone	us-east-1b
Private IPs	172.31.17.47	Security groups	spark2-master, view rules
Secondary private IPs		Scheduled events	No scheduled events
VPC ID	vpc-b65f63d2	AMI ID	spark.ami.pvm.v9 (ami-5bb18832)
Subnet ID	subnet-40e5f010	Platform	

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

```
ec2-user@ip-172-31-13-208:~/spark/ec2
load-spark-env.cmd run-example2.cmd sparkR2.cmd      spark-submit
load-spark-env.sh run-example.cmd sparkR.cmd        spark-submit2.cmd
pyspark   spark-class   spark-shell    spark-submit.cmd
root@ip-172-31-17-47 bin]$ ./spark-shell
16/03/30 20:55:16 INFO spark.SecurityManager: Changing view acls to: root
16/03/30 20:55:16 INFO spark.SecurityManager: Changing modify acls to: root
16/03/30 20:55:16 INFO spark.SecurityManager: SecurityManager: authentication disabled; ui acls disabled; users with view permissions: Set(root); users with modify permissions: Set(root)
16/03/30 20:55:16 INFO spark.HttpServer: Starting HTTP Server
16/03/30 20:55:16 INFO server.Server: jetty-8.y.z-SNAPSHOT
16/03/30 20:55:16 INFO server.AbstractConnector: Started SocketConnector@0.0.0.0:35579
16/03/30 20:55:16 INFO util.Utils: Successfully started service 'HTTP class server' on port 35579.
Welcome to
          _/\_
         /  \_
        /    \_
       /      \_
      /        \_
     /          \_
    /            \_
   /              \_
  /                \_
 /                  \
version 1.6.0

Using Scala version 2.10.5 (OpenJDK 64-Bit Server VM, Java 1.7.0_99)
Type in expressions to have them evaluated.
Type :help for more information.
16/03/30 20:55:21 INFO spark.SparkContext: Running Spark version 1.6.0
16/03/30 20:55:21 WARN spark.SparkConf:
SPARK_WORKER_INSTANCES was detected (set to '1').
This is deprecated in Spark 1.0+.

Please instead use:
- ./spark-submit with --num-executors to specify the number of executors
- Or set SPARK_EXECUTOR_INSTANCES
- spark.executor.instances to configure the number of instances in the spark config.

16/03/30 20:55:21 INFO spark.SecurityManager: Changing view acls to: root
16/03/30 20:55:21 INFO spark.SecurityManager: Changing modify acls to: root
16/03/30 20:55:21 INFO spark.SecurityManager: SecurityManager: authentication disabled; ui acls disabled; users with view permissions: Set(root); users with modify permissions: Set(root)
```

```

ec2-user@ip-172-31-13-208:~/spark/ec2
16/03/30 22:54:48 INFO scheduler.TaskSetManager: Finished task 730.0 in stage 2.0 (TID 2220) in 9778 ms on ip-172-31-24-157.ec2.internal (728/745)
16/03/30 22:54:48 INFO scheduler.TaskSetManager: Finished task 739.0 in stage 2.0 (TID 2229) in 7206 ms on ip-172-31-22-31.ec2.internal (729/745)
16/03/30 22:54:48 INFO scheduler.TaskSetManager: Finished task 732.0 in stage 2.0 (TID 2222) in 9943 ms on ip-172-31-22-195.ec2.internal (730/745)
16/03/30 22:54:48 INFO scheduler.TaskSetManager: Finished task 727.0 in stage 2.0 (TID 2217) in 11738 ms on ip-172-31-29-57.ec2.internal (731/745)
16/03/30 22:54:49 INFO scheduler.TaskSetManager: Finished task 734.0 in stage 2.0 (TID 2224) in 10225 ms on ip-172-31-31-203.ec2.internal (732/745)
16/03/30 22:54:49 INFO scheduler.TaskSetManager: Finished task 733.0 in stage 2.0 (TID 2223) in 10368 ms on ip-172-31-31-183.ec2.internal (733/745)
16/03/30 22:54:49 INFO scheduler.TaskSetManager: Finished task 719.0 in stage 2.0 (TID 2209) in 14098 ms on ip-172-31-21-218.ec2.internal (734/745)
16/03/30 22:54:50 INFO scheduler.TaskSetManager: Finished task 731.0 in stage 2.0 (TID 2221) in 11480 ms on ip-172-31-22-224.ec2.internal (735/745)
16/03/30 22:54:50 INFO scheduler.TaskSetManager: Finished task 737.0 in stage 2.0 (TID 2227) in 9431 ms on ip-172-31-22-224.ec2.internal (736/745)
16/03/30 22:54:50 INFO scheduler.TaskSetManager: Finished task 736.0 in stage 2.0 (TID 2226) in 10818 ms on ip-172-31-23-180.ec2.internal (737/745)
16/03/30 22:54:51 INFO scheduler.TaskSetManager: Finished task 743.0 in stage 2.0 (TID 2233) in 7282 ms on ip-172-31-21-218.ec2.internal (738/745)
16/03/30 22:54:51 INFO scheduler.TaskSetManager: Finished task 740.0 in stage 2.0 (TID 2230) in 9002 ms on ip-172-31-22-195.ec2.internal (739/745)
16/03/30 22:54:51 INFO scheduler.TaskSetManager: Finished task 738.0 in stage 2.0 (TID 2228) in 10972 ms on ip-172-31-20-69.ec2.internal (740/745)
16/03/30 22:54:52 INFO scheduler.TaskSetManager: Finished task 726.0 in stage 2.0 (TID 2216) in 15203 ms on ip-172-31-20-30.ec2.internal (741/745)
16/03/30 22:54:53 INFO scheduler.TaskSetManager: Finished task 735.0 in stage 2.0 (TID 2225) in 13534 ms on ip-172-31-19-74.ec2.internal (742/745)
16/03/30 22:54:54 INFO scheduler.TaskSetManager: Finished task 741.0 in stage 2.0 (TID 2231) in 11501 ms on ip-172-31-19-74.ec2.internal (743/745)
16/03/30 22:54:54 INFO scheduler.TaskSetManager: Finished task 742.0 in stage 2.0 (TID 2232) in 11207 ms on ip-172-31-20-69.ec2.internal (744/745)
16/03/30 22:54:55 INFO scheduler.TaskSetManager: Finished task 744.0 in stage 2.0 (TID 2234) in 10849 ms on ip-172-31-20-30.ec2.internal (745/745)
16/03/30 22:54:55 INFO scheduler.DAGScheduler: ResultStage 2 (saveAsTextFile at <console>:34) finished in 294.561 s
16/03/30 22:54:55 INFO scheduler.TaskSchedulerImpl: Removed TaskSet 2.0, whose tasks have all completed, from pool
16/03/30 22:54:55 INFO scheduler.DAGScheduler: Job 1 finished: saveAsTextFile at <console>:34, took 523.938532 s
scala>

```

Spark Jobs (?)

Total Uptime: 14 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>:34	2016/03/30 22:46:11	8.7 min	2/2	1490/1490
0	sortByKey at <console>:31	2016/03/30 22:45:02	37 s	1/1	745/745

3.3 Experiment: 10GB

EC2 Management Console - Google Chrome

https://console.aws.amazon.com/ec2/v2/home?region=us-east-1#Instances:sort=instanceType

AWS Services Edit More N. Virginia Support

EC2 Dashboard Events Tags Reports Limits

INSTANCES Instances Spot Requests Reserved Instances Scheduled Instances Commands Dedicated Hosts

IMAGES AMIs Bundle Tasks

ELASTIC BLOCK STORE Volumes Snapshots

NETWORK & SECURITY Security Groups Elastic IPs Placement Groups

Launch Instance Connect Actions

Filter by tags and attributes or search by keyword

1 to 4 of 4

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS
spark2-master-i-8fb...	i-8fb22f14	c3.large	us-east-1b	running	2/2 checks ...	None	ec2-52-91-139-19
	i-653635e1	c3.large	us-east-1a	running	2/2 checks ...	None	ec2-54-174-218-1
spark3-slave-i-c437..	i-c4373440	c3.large	us-east-1a	running	2/2 checks ...	None	ec2-52-91-25-39
spark3-master-i-cd...	i-cd373449	c3.large	us-east-1a	running	2/2 checks ...	None	ec2-54-152-239-9

Instance: i-cd373449 (spark3-master-i-cd373449) Public DNS: ec2-54-152-239-9.compute-1.amazonaws.com

Description Status Checks Monitoring Tags

Instance ID	i-cd373449	Public DNS	ec2-54-152-239-9.compute-1.amazonaws.com
Instance state	running	Public IP	54.152.239.9
Instance type	c3.large	Elastic IP	-
Private DNS	ip-172-31-7-51.ec2.internal	Availability zone	us-east-1a
Private IPs	172.31.7.51	Security groups	spark3-master, view rules
Secondary private IPs		Scheduled events	No scheduled events
VPC ID	vpc-b65f63d2	AMI ID	spark.ami.pvm.v9 (ami-5bb18832)
Subnet ID	subnet-1674ee60	Platform	-
Network interfaces	eth0	IAM role	-

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

Spark shell - Spark Jobs - Google Chrome

54.152.239.9:4040/jobs/

Spark 1.6.0 Jobs Stages Storage Environment Executors SQL Spark shell application UI

Spark Jobs (?)

Total Uptime: 13 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>:28	2016/03/31 04:42:29	11 min	2/2	150/150
0	sortByKey at <console>:25	2016/03/31 04:41:29	57 s	1/1	75/75

```

sheetal@sheetal-VPCEH15FD: ~/Desktop/Cloud/Assignment2/SharedMemory/aws
16/03/31 04:52:31 INFO scheduler.TaskSetManager: Finished task 61.0 in stage 2.0 (TID 211) in 12883 ms on ip-172-31-6-182.ec2.internal (62/75)
16/03/31 04:52:33 INFO scheduler.TaskSetManager: Starting task 64.0 in stage 2.0 (TID 214, ip-172-31-6-182.ec2.internal, partition 64,NODE_LOCAL
, 1894 bytes)
16/03/31 04:52:33 INFO scheduler.TaskSetManager: Finished task 62.0 in stage 2.0 (TID 212) in 10131 ms on ip-172-31-6-182.ec2.internal (63/75)
16/03/31 04:52:45 INFO scheduler.TaskSetManager: Starting task 65.0 in stage 2.0 (TID 215, ip-172-31-6-182.ec2.internal, partition 65,NODE_LOCAL
, 1894 bytes)
16/03/31 04:52:45 INFO scheduler.TaskSetManager: Finished task 64.0 in stage 2.0 (TID 214) in 12632 ms on ip-172-31-6-182.ec2.internal (64/75)
16/03/31 04:52:46 INFO scheduler.TaskSetManager: Starting task 66.0 in stage 2.0 (TID 216, ip-172-31-6-182.ec2.internal, partition 66,NODE_LOCAL
, 1894 bytes)
16/03/31 04:52:46 INFO scheduler.TaskSetManager: Finished task 63.0 in stage 2.0 (TID 213) in 15002 ms on ip-172-31-6-182.ec2.internal (65/75)
16/03/31 04:52:46 INFO scheduler.TaskSetManager: Starting task 67.0 in stage 2.0 (TID 217, ip-172-31-6-182.ec2.internal, partition 67,NODE_LOCAL
, 1894 bytes)
16/03/31 04:52:56 INFO scheduler.TaskSetManager: Finished task 65.0 in stage 2.0 (TID 215) in 10353 ms on ip-172-31-6-182.ec2.internal (66/75)
16/03/31 04:52:56 INFO scheduler.TaskSetManager: Starting task 68.0 in stage 2.0 (TID 218, ip-172-31-6-182.ec2.internal, partition 68,NODE_LOCAL
, 1894 bytes)
16/03/31 04:52:56 INFO scheduler.TaskSetManager: Finished task 66.0 in stage 2.0 (TID 216) in 10227 ms on ip-172-31-6-182.ec2.internal (67/75)
16/03/31 04:53:09 INFO scheduler.TaskSetManager: Starting task 69.0 in stage 2.0 (TID 219, ip-172-31-6-182.ec2.internal, partition 69,NODE_LOCAL
, 1894 bytes)
16/03/31 04:53:09 INFO scheduler.TaskSetManager: Finished task 68.0 in stage 2.0 (TID 218) in 12815 ms on ip-172-31-6-182.ec2.internal (68/75)
16/03/31 04:53:09 INFO scheduler.TaskSetManager: Starting task 70.0 in stage 2.0 (TID 220, ip-172-31-6-182.ec2.internal, partition 70,NODE_LOCAL
, 1894 bytes)
16/03/31 04:53:09 INFO scheduler.TaskSetManager: Finished task 67.0 in stage 2.0 (TID 217) in 13696 ms on ip-172-31-6-182.ec2.internal (69/75)
16/03/31 04:53:20 INFO scheduler.TaskSetManager: Starting task 71.0 in stage 2.0 (TID 221, ip-172-31-6-182.ec2.internal, partition 71,NODE_LOCAL
, 1894 bytes)
16/03/31 04:53:20 INFO scheduler.TaskSetManager: Finished task 69.0 in stage 2.0 (TID 219) in 11120 ms on ip-172-31-6-182.ec2.internal (70/75)
16/03/31 04:53:24 INFO scheduler.TaskSetManager: Starting task 72.0 in stage 2.0 (TID 222, ip-172-31-6-182.ec2.internal, partition 72,NODE_LOCAL
, 1894 bytes)
16/03/31 04:53:24 INFO scheduler.TaskSetManager: Finished task 70.0 in stage 2.0 (TID 220) in 15112 ms on ip-172-31-6-182.ec2.internal (71/75)
16/03/31 04:53:36 INFO scheduler.TaskSetManager: Starting task 73.0 in stage 2.0 (TID 223, ip-172-31-6-182.ec2.internal, partition 73,NODE_LOCAL
, 1894 bytes)
16/03/31 04:53:36 INFO scheduler.TaskSetManager: Finished task 71.0 in stage 2.0 (TID 221) in 15406 ms on ip-172-31-6-182.ec2.internal (72/75)
16/03/31 04:53:37 INFO scheduler.TaskSetManager: Starting task 74.0 in stage 2.0 (TID 224, ip-172-31-6-182.ec2.internal, partition 74,NODE_LOCAL
, 1894 bytes)
16/03/31 04:53:37 INFO scheduler.TaskSetManager: Finished task 72.0 in stage 2.0 (TID 222) in 13102 ms on ip-172-31-6-182.ec2.internal (73/75)
16/03/31 04:53:47 INFO scheduler.TaskSetManager: Finished task 73.0 in stage 2.0 (TID 223) in 11494 ms on ip-172-31-6-182.ec2.internal (74/75)
16/03/31 04:53:47 INFO scheduler.TaskSetManager: Finished task 74.0 in stage 2.0 (TID 224) in 10057 ms on ip-172-31-6-182.ec2.internal (75/75)
16/03/31 04:53:47 INFO scheduler.DAGScheduler: ResultStage 2 (saveAsTextFile at <console>:28) finished in 451.156 s
16/03/31 04:53:47 INFO scheduler.TaskSchedulerImpl: Removed TaskSet 2.0, whose tasks have all completed, from pool
16/03/31 04:53:47 INFO scheduler.DAGScheduler: Job 1 finished: saveAsTextFile at <console>:28, took 678.047248 s

```

Spark Master at spark://ec2-54-152-239-9.compute-1.amazonaws.com:7077 - Google Chrome

CS 553 prog2 Micro New Fc Inbox EC2 M Sent M My Dr (no su Spark Spark Upload

Spark 1.6.0

Spark Master at spark://ec2-54-152-239-9.compute-1.amazonaws.com:7077

URL: spark://ec2-54-152-239-9.compute-1.amazonaws.com:7077
 REST URL: spark://ec2-54-152-239-9.compute-1.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, 2 Completed
 Drivers: 0 Running, 0 Completed
 Status: ALIVE

Workers

Worker Id	Address	State	Cores	Memory
worker-20160331040952-172.31.6.182-60627	172.31.6.182:60627	ALIVE	2 (0 Used)	2.7 GB (0.0 B Used)

Running Applications

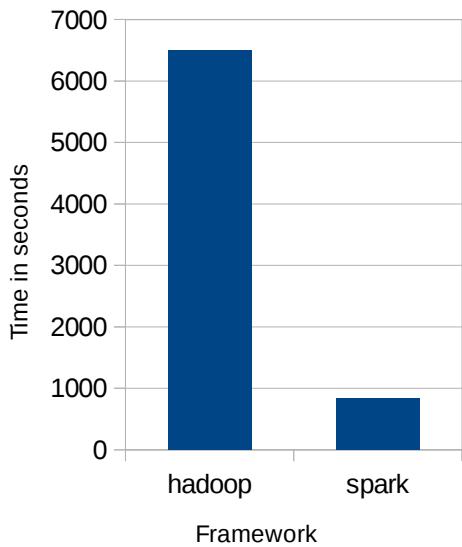
Application ID	Name	Cores	Memory per Node	Submitted Time	User	State	Duration
----------------	------	-------	-----------------	----------------	------	-------	----------

Completed Applications

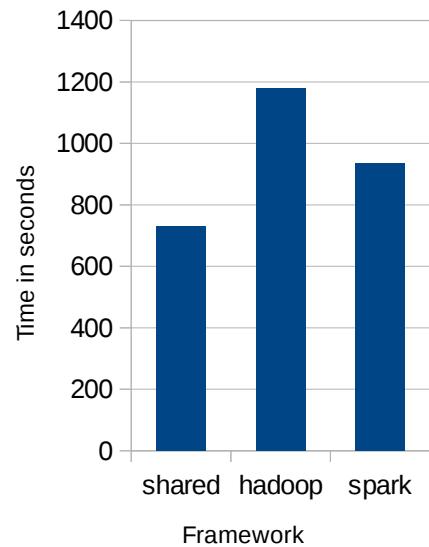
Application ID	Name	Cores	Memory per Node	Submitted Time	User	State	Duration
app-20160331044046-0001	Spark shell	2	2.4 GB	2016/03/31 04:40:46	root	FINISHED	13 min
app-20160331043157-0000	Spark shell	2	2.4 GB	2016/03/31 04:31:57	root	FINISHED	7.2 min

Performance Evaluation

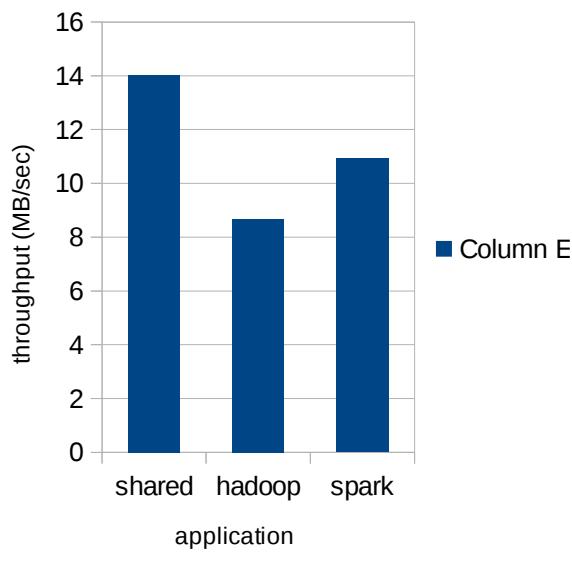
100 GB dataset



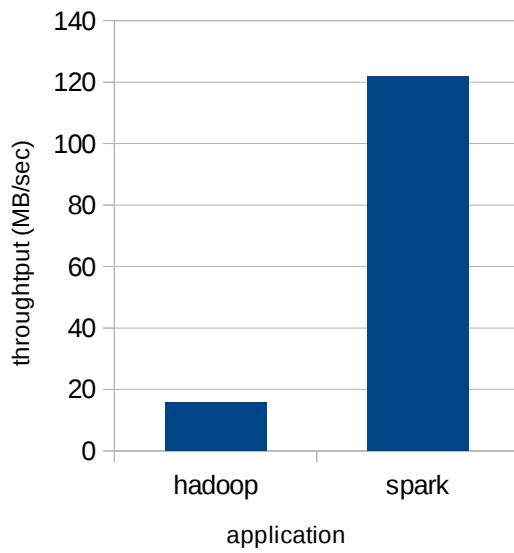
10 GB Dataset

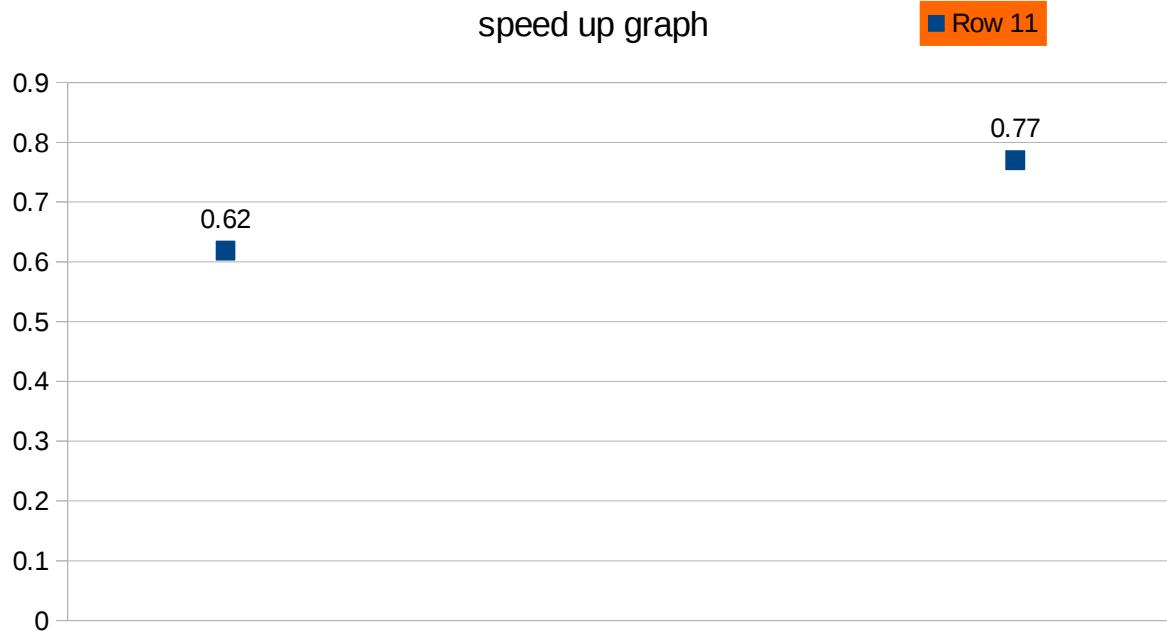


10 GB Throughput



100 GB throughput





0.62 – hadoop, 0.77 -spark with respect to shared memory

Shared Memory, Spark and Hadoop on 10GB dataset:

As seen from above graphs, we can say that for a 10GB dataset, shared memory has taken least time to sort. The reason for this is that the frameworks like hadoop and spark had higher overhead in distributing the data over nodes and that they are made for running on higher datasets. It is one of the limitations of hadoop that it works inefficiently on smaller datasets since the hdfs cannot read the small files in an efficient way.

Spark and Hadoop comparison on 100GB dataset:

- As observed in above graphs, the throughput for spark is much higher. The reason for this is that spark writes data directly to memory. This can be done using the RDDs and it prevents from slowing down of any process. A limited amount of time is spent in moving data in and out of disks.
- It can be seen from graph that the throughput is almost 10 times more than that of hadoop.
- Also for recovering from failed nodes, spark does a much better job than hadoop just by using the RDDs.

Comparison with benchmark:

The benchmark has used a much better performing instance and the number of nodes were set to a higher value in thousands. As a result, they have achieved an outstanding performance. We on the other hand have used a c3.large instance having low memory and have run experiments on a smaller dataset

using only 16 nodes. Hadoop framework works better on a larger dataset with more number of datasets. Thus we can say that the comparison is not the same scale.

We can learn from the CloudSort benchmark that the reason for using external sort is that it is representative of many input output intensive operations. It is of advantage for using public cloud.